Reconsidering Ryu-Takagi RFID Authentication Protocol

Dang Nguyen Duc, Kwangjo Kim
Department of Computer Science
KAIST
Daejeon, Republic of Korea
{nguyenduc, kkj}@kaist.ac.kr

Chan Yeob Yeun
Khalifa University of Science Technology and Research
Sharjah Campus
PO Box 537, UAE
cyeun@kustar.ac.ae

Abstract

Authentication and privacy-preserving are two important security properties for a cryptographic protocol for RFID tags. However, providing privacy-preserving often increases the cost of looking up a tag in the database. In response to this issue, Ryu and Takagi proposed an RFID authentication protocol which provides a way to trade-off security and memory for performance. In this paper, we first point out that Ryu and Takagi's protocol does not provide any privacy-preserving at all. We then propose a solution to achieve privacy-preserving in Ryu-Takagi protocol.

1. Introduction

RFID technology is expected to deliver many powerful applications in supply chain management, smart home appliances and ubiquitous computing, *etc*. At its core, the technology is about giving each and every object of interest a unique identifier that can be read wirelessly by so-called RFID readers. Identity information is embedded into a small device called RFID tag which should be inexpensive to manufacture. A basic RFID system is completed with a back-end server which stores all detailed information about tagged objects. Typically, the back-end database is indexed with object identifiers so that once given an identifier, the detailed information about the object can be quickly looked up.

Unfortunately, RFID technology also faces several security threats, most notably tag cloning and violation of consumer privacy [11]. To deal with tag cloning, we can enforce tag-to-reader and reader-to-tag authentication so that cloned tags are rejected and malicious readers cannot harvest useful information from tags. Many lightweight authentication protocols designed for RFID tags were proposed [4, 5, 6, 7, 10]. Dealing with privacy issue is more subtle. It is not as simple as hiding the true identity of a tag (*e.g.,* by encrypting the tag identifier) because as long as a tag always backscatters a unique number, it can be traced by malicious parties. Therefore, in order to achieve privacy-preserving, it is required for tags to emit different and random-looking information for every time they are queried. A common approach to provide privacy-preserving is to use pseudonym for RFID tags. That is, for each protocol session, each RFID tag uses a different temporary identifier (hence, the name pseudonym) so that malicious parties cannot track RFID tags. However, using different identifiers for every session makes it difficult for the back-end server to identify RFID tags being queried. Some privacy-preserving RFID authentication protocols such as [4] have $O(N)$ worst-case complexity of looking up a tag in the back-end server's database where N is the number of tags in the database. Indeed, many cryptographic protocols differentiate each other by improving this complexity [7, 8, 9, 12, 14]. Among these protocols, a protocol by Ryu and Takagi [14] is unique in its approach. In [14], the authors proposed a privacy-preserving authentication protocol whose goal is to balance the trade-off between security (in terms of privacy-preserving) and performance (in terms of the cost of looking up a tag). Ryu and Takagi addressed the performance-cost-of-privacy problem by encrypting a tag identifier to create tag pseudonyms which can be only decrypted by the back-end server. The authors of [14] also suggested a possible alternative protocol in which the tag has to encrypt its identifier itself using a memory-efficient variant Rabin's encryption scheme [2, 13]. In this paper, we just focus on the main protocol by Ryu and Takagi as it received security analysis by the authors. In addition, we will propose our own alternative to the original Ryu-Takagi protocol without using any encryption.

In this paper, we first point out that Ryu-Takagi protocol does not provide privacy-preserving. In the light of our attack, we suggest a variant of Ryu-Takagi protocol which does not require any encryption nor decryption by the back-end server. Based on this variant, we suggest a new protocol which achieves privacy-preserving and a number of other

advantages as follows:

- *True memory-security trade-off*: Our proposed protocol achieves what Ryu-Takagi protocol failed to do. That is, security can be tweaked by using more or less memory.

- *No public-key encryption required*: Our proposed protocol does not encrypt tag identifiers. Instead, tag pseudonyms are simply random numbers.

- *Early filter of illegitimate tags*: Our proposed protocol allows RFID readers to partially authenticate tags before actually passing legitimate tags to the back-end server for final identification and authentication. In other words, only tags that are verified to be actually in the back-end database will be processed by the back-end server. This potentially reduces the load on the server as well as prevent illegitimate tags from abusing the server's computational resources.

2 Ryu-Takagi RFID Authentication Protocol

In this Section, we briefly review Ryu-Takagi protocol using the same notation described in Table 1. In the description of Ryu-Takagi (and other RFID protocols in this paper), we omit the RFID reader and suppose that the protocol happens between the back-end server and RFID tags. When the presence of a reader makes sense, we will mention it specifically.

Table 1. Notations

Notation	Description
$\mathbf{S}$	Back-end server
D	Tag database
N	Number of tags in D
$\mathbf{auth_S}$	Authentication token of server
$\mathbf{T}$	RFID tag
ID	Tag identifier
$\mathbf{auth_T}$	Authentication token of tag
K	Secret key shared between server and a tag
f	A pseudorandom function
Δ	A set of tag pseudonyms
α	A tag pseudonym
$\mathbf{auth}_\alpha$	Authentication token of tag pseudonym
$\|\|$	Bit string concatenation

In Ryu-Takagi protocol, a trusted party setups tags and populate the tag database as follows: Each tag is assigned a unique identifier ID and a secret key K; The tag identifier is then encrypted m times to produce a set of encrypted tag pseudonyms $\Delta = \{\alpha_1, \alpha_2, \cdots, \alpha_m\}$ where α_i is a randomized ciphertext of ID[1]; Lastly, the pair (Δ, K) is stored in the tag's memory and the tag database D is populated with N pairs (ID, K) for N tags. Tags are assumed to be capable of evaluating a keyed pseudorandom function $f(.)$, *e.g.,* HMAC scheme by Bellare *et al.* $f(.)$ will be used to implement the classical challenge-response authentication between the server and a tag. A detailed description of Ryu-Takagi protocol is illustrated in Fig. 1.

The key idea is Ryu-Takagi protocol is to limit number of pseudonyms per tag to m in a way that the server can always compute the true identifier of each tag from the tag's pseudonyms and easily look up the tag in the database. In other words, depending on m, some security is sacrificed for the sake of performance. If the number m can be as many as the number of times a tag is queried in its lifetime, it is hoped that Ryu-Takagi protocol provides strong privacy-preserving because each encrypted identifier will be used only once. On the other hand, if m is small, storage requirement on a tag is reduced but the tag will have to reuse some of encrypted identifiers in Δ, making it possible for malicious parties to trace the tag.

3. Tracking a tag in Ryu-Takagi protocol

Ryu-Takagi protocol does not provide any security in terms of privacy-preserving. The reason is that a tag gives away one of its encrypted identifier without authenticating the server (or RFID readers in practice). Therefore, a malicious party equipped with a compatible reader can repeatedly query a tag to harvest the whole set of encrypted identifiers Δ of the tag. Because, with high probability, the set Δ stored in each tag is unique, the malicious party can trace the victim tag.

As we can see, encrypting tag identifiers does not help protecting privacy at all. Therefore, it should be possible to remove this entirely without scarifying any security. We propose here a variant of Ryu-Takagi protocol which is as secure as the original protocol (at least in terms of authentication). This variant will be the basis for our improved protocol which achieves true memory-privacy trade-off. The setup procedure is the same as in Ryu-Takagi protocol except the following:

- Instead of encrypting tag identifiers, tag pseudonyms are chosen at random. Note that, choosing tag pseudonyms at random may have storage advantage comparing to that of encrypting tag identifier. For

[1] ID can be encrypted using a randomized encryption algorithm or concatenated with a random number before encrypting with a deterministic cipher.

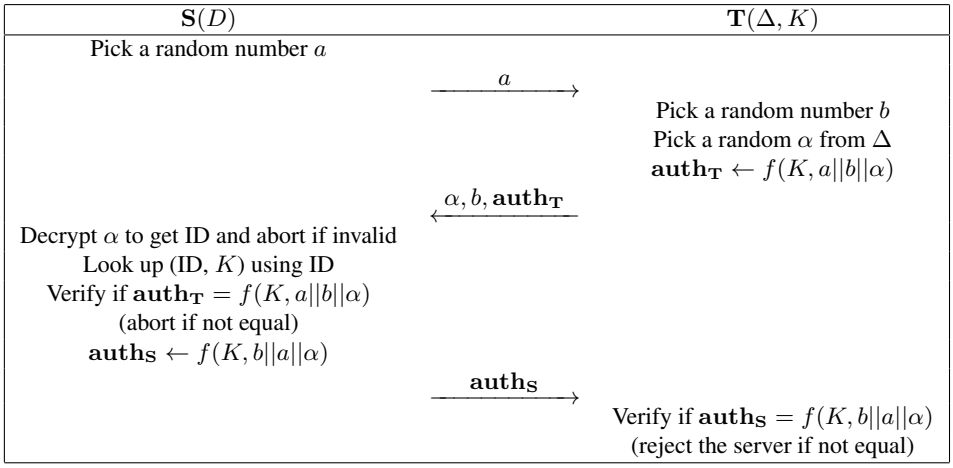

Figure 1. Ryu-Takagi Protocol

example, as suggested in [14], RSA encryption algorithm can be used to create tag pseudonyms. Therefore, each tag pseudonym would require 1024 bits of storage for adequate security. On the other hand, a 128-bit randomly chosen pseudonym should be sufficient for the same level of security.

- The tag database is indexed with tag pseudonyms so that the server can quickly look up a tag given its pseudonyms. Alternatively, the database is indexed with tag identifiers and the server also maintains another table to map tag pseudonyms to tag identifiers.

A detailed description of the modified Ryu-Takagi protocol is depicted in Fig. 2.

4. Our Proposed Protocol

We now present our privacy-preserving protocol based on our variant of Ryu-Takagi protocol described in the last section. As we can see in our attack, in order to prevent malicious parties from harvesting tag pseudonyms, a query request must be authorized. In other words, server-to-tag authentication must happen before tag-to-server authentication. However, before the server identifies and authenticates a tag, the server does not know which secret key the tag being queried has and therefore computes its authentication token. To tackle this problem, we propose another secret key shared between the server and all of the tags. We call this secret key K_S. Note that, even though security against key exposure (*e.g.,* forward security) is not considered in Ryu-Takagi and our improved protocol, it is still necessary to address the issue of having a common K_S for all tags because . In practice, we can have multiple K_S, each for a group of tags. We give the following examples:

- The tag database can be partitioned such as each partition belongs one branch of an organization. One K_S can be assigned for each partition.

- A group of tags that are physically near each other (*e.g.*, tags on items packed in a box or a container) can be assigned one common K_S. However, the server still has to identify which group of tags that a tag being queried belongs to. We can do so by having another tag to act as representative of a group of tags. In order to scan the tags belong a group, the server first queries the presentative tag of the group to identify the group and then looks up the corresponding K_S in its database.

Having server-to-tag authentication before tag-to-server authentication also has other security and performance advantages as follows:

- Authenticating the server (or the readers in practice) first has positive performance implication. The reason is that by having the key K_S, the server can identify and authenticate in two steps: the first step using the secret key K_S and the second step using the secret key K (after the tag has been identified). In practice, the first authentication step can be done by RFID readers so that only tags passing this authentication step will be processed by the server. In other words, the first authentication step verifies that the tags being queried are actually in the database.

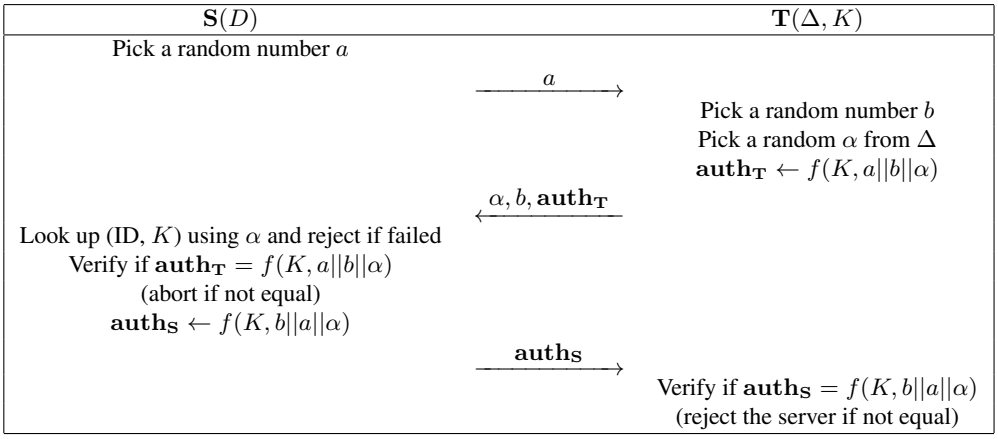

Figure 2. Ryu-Takagi Protocol without Encryption

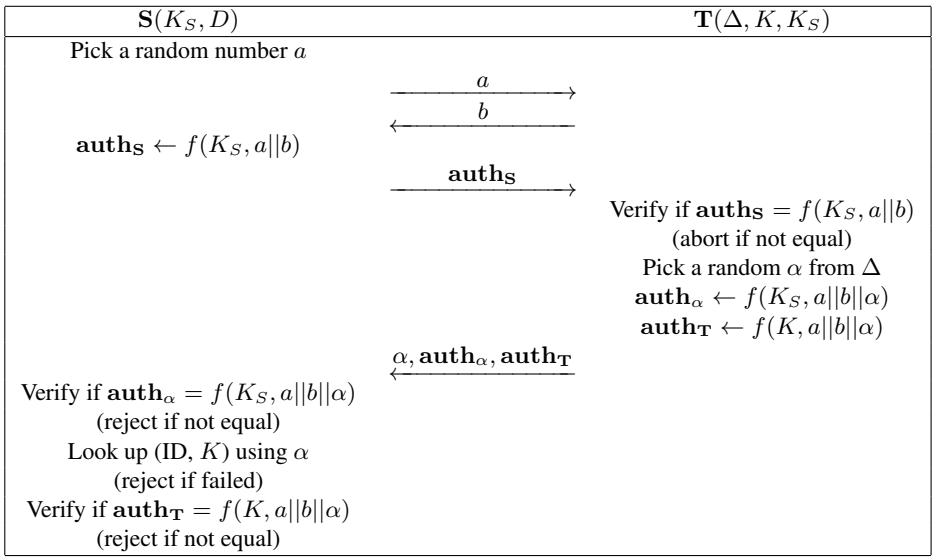

Figure 3. Our Proposed Protocol

- Authenticating the server (or the readers in practice) first is arguably a better security practice than authenticating the tags first. It is because malicious parties cannot extract much information from tags and thus have difficulties in analyzing the protocol signature and detect the existence of tags. This is a security benefit in, for example, e-passport application. As very few countries have deployed e-passports, detecting the existence of tags embedded in e-passports would also imply revealing the nationalities of e-passport holders.

Note that, we haven not addressed the issue of leaking tag pseudonyms by having tags to send them in cleartext. This is a valid concern since Ryu-Takagi protocol and our variant use only m pseudonyms for each tag. An adversary might not be able to collect tag pseudonyms via repeatedly querying a tag, it can still eavesdrop the communication channel between the victim tag and a legitimate reader to capture tag pseudonyms sent in cleartext. However, if the adversary is able to eavesdrop the communication channel between the same tag and legitimate readers multiple times, it implies that the adversary is already capable of tracing the tag. In other words, there is no point in eavesdropping for tag pseudonyms (to trace the tag) when the tag is already traceable.

We now describe our proposed protocol based on our variant of Ryu-Takagi protocol without encrypted tag pseudonyms. The setup procedure proceeds as follows: Firstly, a secret key K_S which will be shared between the server and all tags (or group of tags as we discussed earlier) is chosen. Each tag is assigned a unique identifier ID, a secret key K and a set of m tag pseudonyms

$\Delta = \{\alpha_1, \alpha_2, \cdots, \alpha_m\}$ where α_i is chosen at random; Finally, (Δ, K, K_S) is stored in each tag's memory and the back-end database D is initialized with N triples of (ID, Δ, K) for N tags; The database is indexed with tag pseudonyms (or tag identifiers with an additional table to map tag pseudonyms to tag identifiers). A detailed description of our proposed protocol is illustrated in Fig. 3.

Note that, our proposed protocol is a 4-round protocol instead of a 3-round one like Ryu-Takagi protocols. However, in practice, tags are usually not the one to initiate a protocol session. Therefore, even though the tag can first send its challenge b to the server (and the server's challenge a can be sent together with its response $\mathbf{auth_S}$ thus the protocol becomes a 3-round one without any loss of security), we specified that the server should start a protocol session.

5. Analysis of Our Proposed Protocol

Mutual Authentication. We can see that our proposed protocol is as secure as Ryu-Takagi protocol in terms of providing mutual authentication between the server and tags. The reason is that both protocols employ the same challenge-response mechanism. In particular, the challenge is a random number and the corresponding response is computed as the output of the function $f(.)$ which is fed with the random challenge and a secret key. This authentication mechanism is secure under the assumption that $f(.)$ is a pseudorandom function.

Privacy-preserving. Our proposed protocol provides privacy-preserving by assigning m pseudonyms for each tag. As m increases, we can get to the point where a tag uses a unique pseudonym for each authentication session during its lifespan. Furthermore, malicious parties cannot harvest these tag pseudonyms by repeatedly querying a victim tag because an RFID reader must be authenticated before a tag emits one of its pseudonym.

Comparison. To conclude this section, we compare the computation cost (per one protocol session) and storage requirement between Ryu-Takagi protocol and our proposed protocol in Table. 2 where t, t', k and l are the bit length of a tag pseudonym in Ryu-Takagi, a tag pseudonym in our proposed protocol, secret keys K and K_S, and output of $f(.)$, respectively. Note that, depending on the choice of encryption algorithm in Ryu-Takagi protocol, t' can be much smaller than t.

6. Conclusion

In this paper, we first pointed out that a privacy-preserving RFID authentication by Ryu and Takagi does not provide any privacy-preserving. More specifically, in Ryu-Takagi protocol, the number of pre-determined pseudonyms per tag is fixed for the sake of performance but there is no protection against harvesting of tag pseudonyms by malicious parties.

Table 2. Comparison

	Ryu-Takagi Protocol	**Our Protocol**
Tag storage	$mt + k$	$mt' + 2k$
Tag computation	$2f(.)$	$3f(.)$
Server storage	$N(k + \log_2(\text{ID}))$	$N(ml' + k + \log_2(\text{ID}))$
Server computation	$2f(.)+$ 1 Decryption	$3f(.)$
Communication cost	$t + 2l + \log_2(ab)$	$t' + 3l + \log_2(ab)$

To prevent the attack, we proposed a two-phase authentication approach in which the server/RFID readers should be authenticated first so that RFID tags do not backscatter its pseudonyms to malicious parties. Our proposed protocol achieves the so-called the memory-privacy-trade-off RFID authentication protocol that motivates Ryu-Takagi protocol. That is, the more tag pseudonyms is stored in each tag's memory, the stronger privacy for the tag is guaranteed. Our proposed protocol also exhibits a number of advantages including less memory storage at the tag side and computational overhead at the server side.

7. Acknowledgement

This research was supported by the ICT Standardization program of MKE (The Ministry of Knowledge Economy).

References

[1] EPCglobal Inc, *Class 1 Generation 2 UHF Air Interface Protocol Standard*, Version 1.2.0, Available at `http://www.epcglobalinc.org/standards/uhfc1g2`.

[2] Adi Shamir, *Memory Efficient Variants of Public-key Schemes for Smartcard Applications*, In the Proceedings of EUROCRYPT'94, Springer-Verlag, LNCS 950, pp. 445-449, 1995.

[3] Ari Juels, Ronald Rivest, and Michael Szydlo, *The Blocker Tag: Selective Blocking of RFID Tags for Consumer Privacy*, In V. Atluri (ed.) 8th ACM Conference on Computer and Communications Security, ACM Press, pp. 103-111, 2003.

[4] Miyako Ohkubo, Koutarou Suzuki, and Shingo Kinoshita, *Efficient Hash-Chain Based RFID Privacy Protection Scheme*, In the Proceedings of International Conference on Ubiquitous Computing, Workshop Privacy, September 2004.

[5] Stephen Weis, S. Sarma, Ronald Rivest, and D. Engels, *Security and Privacy Aspects of Low-Cost Radio*

Frequency Identification Systems, In the Proceedings of the 1st Security in Pervasive Computing, Spriner-Verlag, LNCS 2802, pp. 201-212, 2004.

[6] Ari Juels and Stephen Weis, *Authenticating Pervasive Devices with Human Protocols*, In the Proceedings of CRYPTO'05, Victor Shoup (Eds.), Springer-Verlag, LNCS 3261, pp. 293-308, 2005.

[7] Gildas Avoine and Philippe Oechslin, *A Scalable and Provably Secure Hash-Based RFID Protocol*, In the Proceedings of Workshop on Pervasive Computing and Communications Security - PerSec'05, March 2005.

[8] Gildas Avoine, Etienne Dysli, and Philippe Oechslin, *Reducing Time Complexity in RFID System*, In the Proceedings of Selected Areas in Cryptography (SAC)'05, Bart Preneel and Stafford Tavares (Ed.), Springer-Verlag, LNCS 3897, pp. 291–306, 2005.

[9] David Molnar, Andrea Soppera, and David Wagner, *A Scalable, Delegatable Pseudonym Protocol Enabling Ownership Transfer of RFID Tags*, In the Proceedings of Selected Areas in Cryptography'05, Springer-Verlag, LNCS 3897, 276-290.

[10] Tassos Dimitriou, *A Lightweight RFID Protocol to Protect against Traceability and Cloning Attacks*, In the Proceedings of SecureComm'05, September 2005.

[11] Ari Juels, *RFID Security and Privacy: A Research Survey*, In the Journal of Selected Areas in Communication (J-SAC), 24(2): pp. 381-395, February 2006.

[12] Tri Van Le, Mike Burnmester and Breno de Medeiros, *Universally Composable and Forward Secure RFID Authentication and Authenticated Key Exchange*, In the Proceedings of the 2nd ACM Symposium on Information, Computer and Communications Security, pp. 242-252, March 2007.

[13] Adi Shamir, *SQUASH - A New MAC with Provable Security Properties for Highly Constrained Devices Such as RFID Tags*, In the Proceedings of Fast Software Encryption 2008, Springer-Verlag, LNCS 5086, pp. 144-157, 2008.

[14] Eun-Kyung Ryu and Tsuyoshi Takagi, *A Hybrid Approach for Privacy-preserving RFID Tags*, Computer Standards and Interfaces 31 (2009), 812-815.