Secure and Scalable RFID Authentication Protocol

Albert Fernàndez-Mir, Jordi Castellà-Roca, and Alexandre Viejo

Dpt. d'Enginyeria Informàtica i Matemàtiques, UNESCO Chair in Data Privacy, Universitat Rovira i Virgili, Av. Països Catalans 26, E-43007 Tarragona, Spain
{albert.fernandez,jordi.castella,alexandre.viejo}@urv.cat

Abstract. The radio frequency identification (RFID) enables identifying an object remotely via radio waves. This feature has been used in a huge number of applications, reducing dramatically the costs in some production processes. Nonetheless, it also poses serious privacy and security risks to them. Thus, researchers have presented secure schemes that prevent attackers from misusing the information which is managed in those environments. These schemes are designed to be very efficient at the client-side, due to the limited resources of the tags. However, they should be efficient at the server-side also, because the server manages a high number of tags, i.e. any proposal must be scalable in the number of tags. The most efficient schemes are based on client-server synchronization. The answer of the tag is previously known by the server. These kind of schemes commonly suffers desynchronization attacks. We present a novel scheme with two main features: (i) it improves the scalability at the sever-side; and (ii) the level of resistance to desynchronization attacks can be configured.

Keywords: rfid, identification, security, privacy, scalability.

1 Introduction

RFID (Radio Frequency IDentification) is a cutting-edge technology that uses radio waves in order to identify remote objects. A basic RFID scheme includes one reader and several small devices which are named *tags*. Each tag is embedded in the object which must be identified. The tag uses the energy of the wave sent by the reader to transmit its unique serial number to the reader. In this way, the reader identifies a certain item.

Technically, any item is capable of incorporating an RFID tag. Nevertheless, it should be done only in the cases where the relationship between the price of the tag and the price of the item is reasonably sufficient. The easy implantation of these devices paves the way to many novel applications of this technology. The use of this technology can reduce dramatically the costs in some production processes (*e.g.* the textile industry or the car industry among others).

Even though this technology offers important advantages to the users, it also poses serious privacy and security risks to them. According to that, it is necessary

J. Garcia-Alfaro et al. (Eds.): DPM 2010 and SETOP 2010, LNCS 6514, pp. 231–243, 2011.
© Springer-Verlag Berlin Heidelberg 2011

to develop secure schemes that prevent attackers from misusing the information which is managed in those environments. This point is earnestly suggested by the European Union in [1].

The most vulnerable part of the system is the communication between the reader and the tag. During this transmission, an attacker can eavesdrop the messages (*e.g.* the identifier of the tag) and she can identify the item or monitor its position.

In the next section we analyze the different requirements that must be fulfilled by a secure RFID protocol. In Section 3, the current literature in this field is described. Section 4 presents our new proposal. Section 5 analyzes the security of the new protocol. Finally, some conclusions are given in Section 6.

2 Requirements for RFID Protocols

Before designing a new RFID identification protocol, it is important to take into account a number of requirements that must be achieved. These requirements are related to privacy, security and performance issues.

2.1 Privacy

Privacy is one of the main concerns in RFID systems. The use of radio waves to communicate the devices between them enable any attacker to eavesdrop the transmissions and obtain the identification of the tags. Song et al. [2] consider two different issues related to privacy:

- *Profile.* In a traditional RFID system, when the reader asks the tag, it responds with its unique identifier. An unauthorized reader that obtains a certain number of identifiers associated to a single user can create a user profile. Also, RFIDs which contain personal information (*e.g.* passport, medical card...) can disclose this sensible data in an unauthorized way.
- *Monitoring and location.* If the answers of a certain tag can be distinguished from the ones coming from other tags, an attacker can get its location using various readers located in strategic points.

2.2 Security

Below are the most common attacks which can be launched on RFID schemes:

- *Denial of service.* An attacker can jam messages (and eliminate) the messages which are exchanged between the different tags and the reader in order to disrupt the communications.
- *Tracking.* If a certain tag is always identified with the same identifier, an attacker can use various readers in order to know the approximate location of the tag and/or the person who is wearing it.
- *Forward security.* If an attacker is able to compromise a tag, then she can track that tag knowing its previous identifiers and knowing its location.

– *Impersonation of devices.* An attacker may impersonate a tag even without knowing its data. For example, in a replay attack, an attacker can listen to the identification of a certain tag and can forward the same message to the reader behaving like the original tag. Similarly, if an attacker knows the internal state of the tag, it is able to impersonate the reader. The combination of both situations results in a *man-in-the-middle* attack [3].

2.3 Efficiency

Due to the technological limitations of the RFID tags, designers must take into account the following requirements in order to develop an RFID identification protocol:

– *Minimizing the storage capacity.* The volume of data that is stored in the tag should be minimal due to the limited memory of these devices.
– *Minimizing the computational cost.* The computational cost required in the tag side should be minimal due to the lack of resources (*e.g.* energy, computational power) of the tags.
– *Minimizing the communication cost.* The volume of data that each tag can transmit per second is limited by the bandwidth which is available for RFID tags [4, 5].
– *Scalability.* The server must be able to identify a large number of different tags using the same radio station. The use of exhaustive search in its database can slow down the identification process.

3 State of the Art

There are different types of secure authentication protocols for RFID systems. This section describes two classes of protocols. The first one is based on asymmetric cryptography. This kind of cryptography provides more security to the

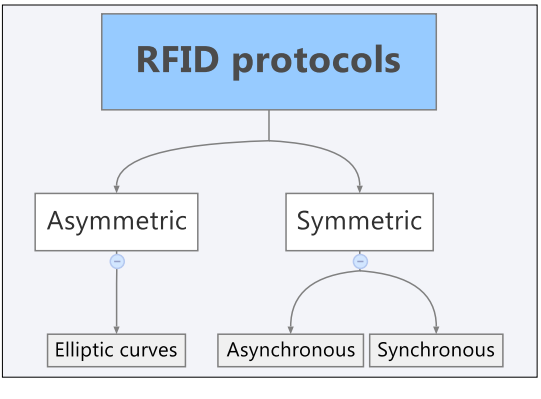


Fig. 1. Clasification of RFID authentication protocols

system. The second class is based on symmetric cryptography. Currently, this approach is the most significant and used because it can be implemented using a minor number of logic gates.

3.1 Asymmetric Protocols

The use of asymmetric cryptography is possible in RFID schemes. In this way, the work presented in [6] uses elliptic curves. Nevertheless, the limited resources of the tags limit the implementation of this kind of protocols: these schemes require a high number of logic gates to be executed and a high number of logic gates implies a higher cost for the tag. Therefore, in the present day their implementation is unfeasible.

3.2 Symmetric Protocols

Nowadays, this kind of protocols are the most used. Symmetric cryptography fits better in low-cost tags. Nevertheless, it is more difficult to achieve a good privacy level. These protocols can be divided into two categories: asynchronous and synchronous.

Asynchronous protocols. All the protocols found into this category base their security on one-way hash functions. From knowledge of the hash function h and $y = h(x)$, an intruder is not able to determine x. Hash-based protocols are very efficient in the tag side because the operations which are performed are minimal. According to that, they are really suitable for lightweight RFID schemes like the ones which are currently being used.

The first protocol based on hash functions was proposed by Rivest et al. [7] in 2003. In this work, the use of deterministic hash locks was presented. The OSK protocol [5] works in a similar way but here the IDs are updated after each identification using hash functions. This behaviour prevents intruders from tracing the IDs of the tags. This protocol also suffers from scalability issues since it is necessary to make a thorough search for each identifier until the right one is found. Avoine and Oechslin addressed this shortcoming in [8] that reduces the scalability problem but is vulnerable to replay attacks.

Juels [9], in 2006, proposed one of the most popular protocols in this field: the random hash locks. In this protocol, both the reader and the tag generate a random number r. The tag concatenates r with its unique ID and obtains a hash value $y = h(r||ID)$. The tag sends y to the reader and this device searches its database for the ID that matches y. Note that the reader has to compute $y_i = h(r||ID_i)$ for each ID_i in the database until the correct one is found. In the worst case, this process is done as many times as tags are controlled by the system. The main problem of this method is again the scalability of the system [10].

Despite the efforts made in this field, current protocols still have serious scalability problems at the server side when the number of tags is very high.

Synchronous protocols. This kind of protocols appear because of the scalability issues related to the hash-based protocols. Several synchronous protocols are also based on hash functions. Nevertheless, their main particularity is the use of state synchronism between the reader and the tag. In this way, if both devices are synchronized, these protocols enable quick and secure identification because the server knows the next tag response. Protocols like [11, 12] update the identifiers of the tags but still remain vulnerable to traceability attacks.

Regarding scalability, these protocols are scalable while both devices are synchronized but they have problems when the synchronism is lost. The YA-TRAP protocol [13] uses a series of time states $[T_0, ..., T_{MAX}]$ that protects the identity of the tag.

Dimitriou [14] in 2005 and Lee et al. [15] in 2006, proposed two new protocols based on synchronization between server and tag. Even so, their proposals had did not prevent that an attacker could track a tag. To solve this problem Ha et al. [16] use the synchronization state for fast identification. They use hash locks to identify the tags when they are desynchronized. This is a serious problem in terms of scalability because if an attacker disrupts the update messages, the system does not scale correctly.

The problems arise when an attacker desynchronize the two devices. Other problems in these types of protocols appear when an attacker tries to track a tag. However, several protocols based on synchronism [17–20], have different security issues that are described in [21] as for example: tracking, impersonation of devices or denial of service.

4 Our Proposal

As seen in Section 3, the protocols that are efficient at both server and client sides, are those based on synchronization. However, as detailed in [21] these protocols are vulnerable to desyncronization attacks or when they are attacked, they present scalability problems. We present a protocol that is efficient for both the client and the server in the synchronization state. Besides, if the protocol is not synchronized, the level of resistance against denial of service attacks can be selected and the protocol will still be robust against privacy threats. The proposed scheme has been designed considering that an attacker must be unable to distinguish whether a tag is synchronized or not. This fact adds one more degree of difficulty to an attacker who wants to desynchronize the system.

The proposed protocol consists of four phases: initialization phase, synchronized identification phase, desynchronized identification phase and update phase. The last one is executed after each successful identification. If the tag cannot be updated and the system is desynchronized, it can still be identified with the desynchronized identification phase preserving the initial security and privacy properties.

Table 1 shows the notation used to describe the new proposal.

Table 1. Notation used in the protocol

id	Tag identifier
R	Reader
T	Tag
r_i	Random number i
$h()$	Unidirectional Hash Function
$h_k()$	Keyed Hash Function (HMAC)
ks	Secret key of reader
$PRNG$	Pseudo-Random Number Generator
$SYNC$	Synchronization State
C_i	Pseudo-Random bit sequence
S	Pseudo-Random bit sequence
m_i	Update message i
k_δ	$h_{id}(r_1)$ done δ times
$\|\|$	Concatenation Operator
$\oplus$	XOR Operator

Table 2. BD values

$Table\,k$	$h'_{id}(r_1)$	r'_1	$h_{id}(r_1)$	r_1	$h_{ks}(id)$	id
r_1 r'_1						
k_1						
k_2						
k_3	...	...	...	...	...	...
...						
k_{MAX}						

4.1 System Environment

The proposed protocol requires a server that hosts its own database and a reader that transmits information from the various tags which are present in the scenario.

Each tag has a record in the server as shown in Table 2, and the server stores the following data for each tag:

- id: The identity of the tag.
- $h_{ks}(id)$: Hash value, where ks is a secret key that is only known by authorized readers and it used to generate the hash value.
- r_1 *and* r'_1: These random values and their associated hash values $h_{id}(r_1)$. Current random value (r_1) and next random value (r'_1) are kept in order to distinguish the state of synchronization of the system.
- $Table\,k$: This table is used to identify the tag in case of the system is not synchronized. We keep two columns of data corresponding to the values of the key r_1 previous and present used in secondary identification phase.

Another value that must be taken into account is MAX. This value is used to set the level of resistance to denial of service attacks. MAX is the is the length of the $Table\,k$ and the possible values that the server knows to identify a tag in a desynchronized identification phase.

4.2 Protocol Phases

In this section we describe the different phases of our protocol. Table 3 details the four phases and their steps.

Initialization phase. In this first phase, the server sends to the reader the following information: $h_{ks}(id)$, r_1 and $h_{id}(r1)$. This information will be used later by the tag to perform the authentication phase. The data is transmitted using a secure channel that guarantees their confidentiality.

Synchronized identification phase. When the tag and reader are synchronized, they run this phase to identify the tag. The system follows the next steps: the reader sends a random number r_0 to the tag. Then, the tag answers with the following information: $h_{id}(r_1)$, C_0 and r_2, where $C_0 = PRNG(r_1||r_0)$ and r_2 is a random value that is generated by the tag.

Upon reception of this data, the reader forwards it to the server. Then, the server computes r_1 and id. Besides, the server must be able to verify the tag sequence C_0 to ensure that it is authentic. This is done to avoid a phishing attack or a replay attack.

After sending all the data, the tag goes to a desynchronized state ($SYNC = 0$) until the update phase ends.

Update phase. This phase is used to update the values of the tag and avoid several attacks in order to protect the privacy. This phase is executed after synchronized an desynchronized phases.

First of all, the server composes m_0 concatenating a new $r_1^{'}$ and new $h_{id}(r_1^{'})$, these elements will be updated by the tag after the update phase. Then, the server generates m_1 that will be used by the tag to verify m_0. Later, the server concatenates m_0 and m_1 into m_2, generates the pseudorandom sequence S where $h_{ks}(id)||r_1$ is the seed and it finally computes $C_1 = m_2 \oplus S$. Finally, the server sends C_1 to the tag and the columns r_1 and r_1' of the $Table\,k$ are updated using the new key r_1'.

When the tag receives the message C_1, obtains the pseudorandom sequence $S^{'}$ with the same seed as the server $(h_{ks}(id)||r_1)$ and decrypts the message C_1 by a XOR operation. Then checks whether m_1 is correct or not. If it is correct, then the tag updates the new values $r_1^{'}$ and $h_{id}(r_1^{'})$ and sets $SYNC = 1$.

Desynchronized identification phase. This phase is executed when both devices are desynchronized. If the message C_1 corresponding to the update phase is incorrect or it has been received incorrectly (and assuming that $SYNC = 0$), the tag will follow the next steps: First of all the reader sends r_0 to the tag like

Table 3. Proposed authentication protocol

Server	**Reader**	**Tag**
Initialization phase (with secure channel)		
	$h_{ks}(id), r_1, h_{id}(r_1)$ →	
Synchronized identification phase		
	$r_0 \in R\{0,1\}^*$ r_0 →	$r_2 \in R\{0,1\}^*$ $C_0 = PRNG(r_1 \| r_0)$ $SYNC = 0$
$h_{id}(r_1), C_0, r_2, r_0$ ←	$h_{id}(r_1), C_0, r_2$ ←	
Search $h_{id}(r_1)$ Check C_0		
Update phase		
$m_0 = r_1^{'} \| h_{id}(r_1^{'})$ $\alpha = h_{ks}(id) \| r_2$ $m_1 = h_\alpha(m_0)$ $m_2 = m_0 \| m_1$ $PRNG(h_{ks}(id) \| r_1) = S$ $C_1 = m_2 \oplus S$		
C_1 →	C_1 →	$S^{'} = PRNG(h_{ks}(id) \| r_1)$ $m_3 = C_1 \oplus S^{'}$ $\alpha^{'} = h_{ks}(id) \| r_2$ $m_1 \stackrel{?}{=} h_{\alpha^{'}}(m_0)$ $r_1 = r_1^{'}, h_{id}(r_1) = h_{id}(r_1^{'})$ $SYNC = 1$
Desynchronized identification phase		
	$r_0 \in R\{0,1\}^*$ r_0 →	$r_2 \in \mathbb{R}\{0,1\}^*$ $\delta = \delta + 1$ $k_\delta = h_{r_1}(k_{\delta-1})$ $C_0 = PRNG(h_{r_1}(r_2 \| r_0))$
k_δ, C_0, r_2 ←	k_δ, C_0, r_2 ←	
If k_δ is not in BD Calculation of id by k_δ Check C_0 Start Update phase		

in the synchronized identification phase. Then the tag generates a new r_2 and, for $k_0 = h_{id}(r_1)$, performs the following computations:

$$\delta = \delta + 1$$

$$k_\delta = h_{r_1}(k_{\delta-1})$$

Next, the reader forwards the data to the server that will search for the value k_δ in the database like in the synchronized identification phase. If this value is not found, the server will search k_δ in the $Table\,k$ of each tag. If it finds the value, the identifier of the tag will be checked using C_0. If the server has two or

more records of k_δ in the database, it will obtain the correct identifier through checking the C_0 value. After identifying the tag, the reader starts the Update phase.

In case of desynchronization, the server has MAX opportunities to identify the tag before this becomes useless. The value of this parameter is discussed in the subsection 4.3.

4.3 Selection of Parameter MAX

The parameter MAX which is used in the protocol influences the system in the following three concepts:

- *Memory Space.* Considering n tags, in the BD there is a table of MAX values associated to each tag. Each of these values is equivalent to the output of a hash function. The output of the hash function SHA-1 [22] has a length of 160 bits. As a result, the memory requirements for this concept are $n \cdot MAX \cdot 160$ bits.
- *Computational cost.* In the secondary identification phase, k_δ must be found in the table of MAX positions which is associated to each tag. When k_δ is located, the reader knows the *id* of the corresponding tag. The tag associated to k_δ is located with a cost of $O(1)$ because it is only necessary to know whether this element is missing or not. As a result of that, the cost to locate the tag that contains k_δ in its table is O (1) because the server only need to know if this element is missing or not in our database. Therefore, the system scales correctly for large values of n.
- *Security.* The system can be configured to resist MAX attacks of desynchronization, *i.e.* a Denial of Service (DoS). This is justified in Section 5.

As a conclusion, the value of MAX depends on the available memory space and time β in which an attacker is able to get the MAX values of table k. For the calculation of this value, we consider an interval of 200 ms for each identification.

Table 4. Table of values MAX

MAX	$Tags$	$Disk\ space$	$Time\ \beta$
100	100	0,024 Gb	20 s
1.000	1.000	2,4 Gb	200 s
10.000	10.000	240 Gb	2000 s
100.000	100.000	24 Tb	5,5 h
1.000.000	1.000.000	2400 Tb	55,5 h

Although the memory requirements are high, we consider that this resource is cheap and therefore it is acceptable. In terms of time, it depends on the level of security which is needed by the system. Due to the current power of the processors, we believe that the user response time is not a major drawback for the system.

Table 4 presents different results depending on the MAX value in use and the number of tags in the system. Considering parameter β, it can be observed that for large values of MAX, the time which is necessary to obtain all the records in table k is high enough to avoid any attack. Nevertheless, the shortcoming of this security is the memory space which is needed.

5 Security Analysis

The protocol has the following properties of security and privacy:

- *Private identity.* A tag does not know its own *id*. It only knows $h_{ks}(id)$. Thus, an attacker who is able to get the contents of the tag will not know its true identifier.
- *Locating and tracking a certain tag.* Location privacy is guaranteed because, in each identification, the data sent is always different. In this way, values $h_{id}(r_1)$, C_0 and r_2 are updated in each identification. If the reader and the tag are not synchronized, then the updated elements are: k_δ, C_0 and r_2. If both devices are synchronized, $h_{id}(r_1)$ can appear twice with a probability of $1/2^{160}$. The probability for k_δ is the same as $h_{id}(r_1)$ in consecutive secondary identifications.
- *Impersonating a certain tag.* An attacker does not know values $h_{ks}(id)$, r_1, $h_{id}(r_1)$. These elements are sent in the initialization phase using a secure communication channel. An attacker trying to impersonate a certain tag without knowing these values will be detected by the reader.
- *Impersonating the server.* This case is similar to the last property. The server can only be impersonated by an attacker who knows the entire database. We assume that the database is hosted in a secure environment.
- *Replay attacks.* This type of attack is not possible in this protocol because in each identification, each one of the sides (the tag and the server) provides a new value which is computed at random. In this way, a replay attack at the server side will be successful only if the message which has been captured previously contains the expected r_0 value. The same happens at the tag side but with the value r_2. Random values r_0 and r_2 are generated in each device and they are unlikely to be repeated in a short period of time. In this way, value $h_{id}(r_1)$ is different in each Synchronized identification phase. In the Desynchronized identification phase case, k_δ is always different.
- *Forward security.* Our protocol is partially resistant to this kind of attacks. An attacker can obtain the following data: $h_{ks}(id), r_1, h_{id}(r_1)$. In order to link this data with the previous authentication (Synchronized identification phase), she should be able to compute r_1^* from the PRNG using sequence S,

where r_1' is the previous r_1^*. This operation is computationally hard since a secure PRNG is being used. Next, the attacker should be able to find the id used in the HMAC $h_{id}(r_1^*)$. Assuming that a secure HMAC function is being used, this attack is computationally unfeasible. However, the attacker could still identify the authentications done with the secondary method which have been computed using r_1.

– *Denial of service.* An attacker can query the tag or interrupt the update phase $MAX + 1$ times. So, the reader will not find k_δ in the database, *i.e.* the server cannot identify the tag. Since, MAX is a security parameter of our system, we can set this value in order to increase the resistance of the proposed protocol against this attack. If the table is large enough, the time which is needed to consume all the values will be high enough (this was discussed in Section 4.3).

Table 5, shows a comparison of security among several protocols of secure authentification on RFID technology.

Table 5. Comparison of security

Protocol	LCRP [14]	Juels et al. [9]	Lee et al. [15]	Ha et al. [16]	Our proposal
Information leakage	◯	◯	◯	◯	◯
Spoofing attack	◯	◯	◯	◯	◯
Replay attack	◯	◯	◯	◯	◯
Indistinguishability	×	◯	◯	◯	◯
Forward security	△	×	△	△	△
Resyncrhonization	×	◯	◯	◯	◯

◯: secure or support △: partially secure ×: insecure or not support

6 Conclusions and Future Work

We have presented a new protocol that is more scalable than the hash-based protocols or the synchronous proposals. We have used both technologies in order to achieve a cost of $O(1)$ for each identification. Besides, attackers are not allowed to know whether the system is synchronized or desynchronized.

As a secondary feature, tags do not know their own ids so an attacker who compromises a tag is not able to know its identifier.

Finally, the proposed scheme can be configured in order to resist a DoS attack. This is done by setting parameter MAX. For larger values of this parameter, the proposed system can stand better against DoS attacks but the memory space which is required is also higher.

In this way, the main shortcoming of the proposed scheme is the memory requirements. In scenarios with millions of tags, our database requires Terabytes of space. Fortunately, the market offer cheap solutions to address this point.

Future work will focus on implementing the proposed protocol in a real scenario. Current RFID low-cost tags have a very limited number of logic gates, hence they cannot be used to deploy our scheme. Fortunately, it is possible to use smart cards instead. This technology can offer a practical environment based on standards such as ISO 14443 and ISO 15693. In the future, low-cost tags with improved capabilities are expected to be available.

Disclaimer and Acknowledgments

The authors are solely responsible for the views expressed in this paper, which do not necessarily reflect the position of UNESCO nor commit that organization. This work was partly supported by the Spanish Ministry of Education through projects TSI2007-65406-C03-01 "E-AEGIS" and CONSOLIDER CSD2007-00004 "ARES", by the Spanish Ministry of Industry, Commerce and Tourism through project "eVerification" TSI-020100-2009-720 and by the Government of Catalonia under grant 2009 SGR 1135.

References

1. European Union, Commission recomendation of 12 May 2009 on the implementation of privacy and data protection principles in applications supported by radio-frequency identification. In Official Journal of the European Union (2009), http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2009:122:0047:0051:EN:PDF
2. Song, B., Mitchell, C.J.: RFID authentication protocol for low-cost tags. In: Proceedings of the First ACM Conference on Wireless Network Security, pp. 140–147 (2008)
3. Sarma, S., Weis, S., Engels, D.: RFID systems and security and privacy implications. In: Kaliski Jr., B.S., Koç, Ç.K., Paar, C. (eds.) CHES 2002. LNCS, vol. 2523, pp. 1–19. Springer, Heidelberg (2003)
4. Avoine, G.: Cryptography in radio frequency identification and fair exchange protocols. In: Faculté Informatique et Communications, pp. 2007–06 (2005)
5. Ohkubo, M., Suzuki, K., Kinoshita, S., et al.: Cryptographic approach to privacy-friendly tags. In: RFID Privacy Workshop, vol. 82 (2003)
6. Martínez, S., Valls, M., Roig, C., Miret, J.M., Giné, F.: A secure Elliptic Curve-Based RFID Protocol. J. Comput. Sci. Tech. 24(2), 308–318 (2009)
7. Rivest, R., Weis, S., Sarma, S., Engels, D.: Security and privacy aspects of low-cost radio frequency identification systems. In: Hutter, D., Müller, G., Stephan, W., Ullmann, M. (eds.) PC 2003. LNCS, vol. 2802, pp. 454–469. Springer, Heidelberg (2003)
8. Avoine, G., Oechslin, P.: A scalable and provably secure hash-based RFID protocol. In: PERCOMW, pp. 110–114 (2005)
9. Juels, A.: Rfid security and privacy: A research survey. IEEE Journal on Selected Areas in Communications 24(2), 381–394 (2006)
10. Solanas, A., Domingo-Ferrer, J., Martínez-Ballesté, A., Daza, V.: A distributed architecture for scalable private RFID tag identification. Computer Networks 51(9), 2268–2279 (2007)

11. Kanf, J., Nyang, D.: RFID authentication protocol with strong resistance against traceability and denial of service attacks. In: Molva, R., Tsudik, G., Westhoff, D. (eds.) ESAS 2005. LNCS, vol. 3813, pp. 164–175. Springer, Heidelberg (2005)
12. Yang, J., Park, J., Lee, H., Ren, K., Kim, K.: Mutual authentication protocol for low-cost RFID. In: Handout of the Ecrypt Workshop on RFID and Lightweight Crypto (2005)
13. Tsudik, G.: YA-TRAP: Yet another trivial RFID authentication protocol. In: Fourth Annual IEEE International Conference on Pervasive Computing and Communications Workshops, PerCom Workshops 2006 (2006)
14. Dimitriou, T.: A lightweight RFID protocol to protect against traceability and cloning attacks. In: Security and Privacy for Emerging Areas in Comunications Networks, SecureComm 2005, pp. 59–66 (2005)
15. Lee, S., Asano, T., Kim, K.: RFID: Mutual Authentication Scheme based on Synchronized Secret Information. In: Proceedings of the SCIS 2006 (2006)
16. Ha, J.C., Moon, S.J., Nieto, J., Boyd, C.: Low-cost and strong-security RFID authentication protocol. In: Denko, M.K., Shih, C.-s., Li, K.-C., Tsao, S.-L., Zeng, Q.-A., Park, S.H., Ko, Y.-B., Hung, S.-H., Park, J.-H. (eds.) EUC-WS 2007. LNCS, vol. 4809, pp. 795–807. Springer, Heidelberg (2007)
17. Lee, S., Asano, T., Kim, K.: RFID mutual authentication scheme based on synchronized secret information. In: Symposium on Cryptography and Information Security (2006)
18. Osaka, K., Takagi, T., Yamazaki, K., Takahashi, O.: An efficient and secure RFID security method with ownership transfer. In: Wang, Y., Cheung, Y.-m., Liu, H. (eds.) CIS 2006. LNCS (LNAI), vol. 4456, pp. 778–787. Springer, Heidelberg (2007)
19. Seo, Y., Lee, H., Kim, K.: A scalable and untraceable authentication protocol for RFID. In: Zhou, X., Sokolsky, O., Yan, L., Jung, E.-S., Shao, Z., Mu, Y., Lee, D.C., Kim, D.Y., Jeong, Y.-S., Xu, C.-Z. (eds.) EUC Workshops 2006. LNCS, vol. 4097, pp. 252–261. Springer, Heidelberg (2006)
20. Song, B., Mitchell, C.J.: A scalable and untraceable authentication protocol for RFID. In: RFID Authentication Protocol for Low-Cost Tags. In Wireless Network Security (WISEC), pp. 140–147 (2008)
21. Van Deursen, T., Radomirovic, S.: Attacks on RFID protocols. In: IACR eprint Archive 2008, vol. 310 (2008)
22. Eastlake, D., Jones, P.: US secure hash algorithm 1 (SHA1). In: RFC 3174 (2001)