Martínez S, Valls M, Roig C *et al.* A secure elliptic curve-based RFID protocol. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 24(2): 309–318 Mar. 2009

A Secure Elliptic Curve-Based RFID Protocol

Santi Martínez[1], Magda Valls[2], Concepció Roig[1], Josep M. Miret[2], and Francesc Giné[1]

[1] *Computer Science Department, Universitat de Lleida, Lleida, Spain*

[2] *Mathematics Department, Universitat de Lleida, Lleida, Spain*

E-mail: {santi, roig, sisco}@diei.udl.es; {magda, miret}@matematica.udl.es

Received March 15, 2008; revised December 18, 2008.

Abstract Nowadays, the use of Radio Frequency Identification (RFID) systems in industry and stores has increased. Nevertheless, some of these systems present privacy problems that may discourage potential users. Hence, high confidence and efficient privacy protocols are urgently needed. Previous studies in the literature proposed schemes that are proven to be secure, but they have scalability problems. A feasible and scalable protocol to guarantee privacy is presented in this paper. The proposed protocol uses elliptic curve cryptography combined with a zero knowledge-based authentication scheme. An analysis to prove the system secure, and even forward secure is also provided.

Keywords elliptic curve cryptography, forward security, RFID, zero knowledge

1 Introduction

A Radio Frequency Identification (RFID) system allows the remote identification of items that have an RFID tag attached. This is particularly useful in supply chains, stores, etc. It is expected that, in the future, everyday objects will have RFID tags that will enable interesting applications, such as medicines with RFID tags on their package which would be allowed to link a unique identifier for that package to important information of it, like the caducity or contra-indications. Anyway, this kind of services would not be wished by the end user if they entailed serious security problems and, for that reason, several studies are focused on solving the vulnerabilities of these systems, in order to make them secure[1−5].

As can be seen in Fig.1, an RFID system consists of three components.

• Tags, with an integrated circuit and a small antenna, that are placed in each object that should be identified (e.g., the medicines). Each tag will send its identifier (ID) under request.

• Reader(s) that communicates with a centralized database and with the tags. They are responsible for performing the queries to the tags.

• Database with information of the tagged items (e.g., medicine name, chemical components, ...). RFID readers will check the database to identify an object and to obtain its associated information.

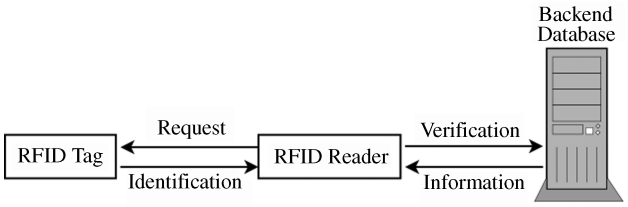

Fig.1. RFID system.

The necessity of having a database in the RFID system is due to the memory limitations of the tags. They are very small, so they cannot store much more information than an identifier. Thus, a backend database is needed for storing the rest of the product information. However, it is important to guarantee that the search for the IDs should be efficient and scalable in a number of tags of the system.

Depending on their power source, tags can be classified as passive tags, semi-passive tags and active tags. The passive tags do not have batteries. This kind of tags derive their power from the signal of the reader. The semi-passive tags have batteries which power their circuitry when they are interrogated. Finally, the active tags have batteries which power their transmissions. The active tags are more expensive, so they are only used in environments where their cost is justified. Only the active tags can initiate a communication.

In this work, we will focus on passive tags since they are cheaper, and are the most broadly used[6]. However,

Regular Paper

This work is supported by the Generalitat de Catalunya under Grant No. FIC 2007FIC 00880, and the projects of the Spanish MCyT MTM2007-66842-C02-02 and TIN2006-15662-C02-02.

their simplicity implies important restrictions on the cost (it should not surpass cents of a dollar), the number of logic gates (about 15 000[1]) and the transmission rate (520 or 640 bps depending on the band used). Considering that a reading operation should take nearly a second, the protocol established between the reader and the tag should transmit less than 500 bits.

Concerning the confidence, there are two main privacy problems related to the RFID systems: (a) leakage of information of the tags should be avoided, otherwise it would allow clandestine inventory and, (b) tracking the behavior of its user should be prevented. The solution to the leakage of information implies some form of encryption of tag IDs. And the tag IDs should be frequently changed in order to avoid the tracking problem. Additionally, the property called forward security is also desirable to provide higher confidence to the system. This will ensure that the revelation of the tag secret information will not unveil the encrypted information that was previously sent.

In this work, we propose a new protocol to guarantee the privacy in the communications between the tags and readers in an RFID system. This protocol is able to solve the privacy problems while taking into account the implementation restrictions associated with current passive tags. The proposed approach is based on elliptic curve cryptography, which allows the use of fewer bits than conventional public key cryptography guaranteeing the same security.

The novelty of the proposed approach is that the protocol requires reader authentication (instead of only tag authentication), by means of an elliptic curve-based zero knowledge protocol. We prove that, in this condition, the system provides forward security and, additionally, it is scalable when the number of tags increases.

The remaining paper is structured as follows. Section 2 outlines some related work and the idea of our solution. Section 3 sketches some preliminaries. In Section 4, the proposed protocol is described. In Section 5, some implementation issues are discussed. A security analysis is given in Section 6. Finally, Section 7 outlines the main conclusions.

2 Related Work

Several approaches have been proposed in the literature during the last years, trying to solve the RFID user privacy problems[1−5,7−9].

In *Tag Killing* scheme[8], the tags have a PIN protected *kill* command that allows deactivation of the tags to protect user's privacy. When a tag receives the killing command with the corresponding PIN it renders itself permanently inoperative. However, this approach is not suitable for systems that wish to use the benefits of RFID.

Some systems such as *Hash Lock*[7], *Anonymous ID*[4] or *External Encryption*[3] solve the privacy problem because the tags do not store their real identifier, but they do not solve the problem of tracking.

And some others, such as the *Random Hash Lock*[7], solve the privacy and tracking problems, because the tags send identifiers pseudo-randomly generated. Unfortunately, they do not provide forward security.

These previous schemes have scalability problems. Some of them, such as the *Hash Lock*[7], require that the reader checks all the possible tag keys, which will be infeasible in very large systems.

More efficient schemes are the tree-based approaches, such as Molnar and Wagner[9], since the key search that the reader must perform to identify a tag is logarithmic in the maximum number of tags (it depends on the depth of the key tree). In these systems, each tree node has a key and each tag is associated with a leaf of this key tree, thus receiving all the keys corresponding to the path from the root to its leaf. These systems are vulnerable to the leakage problem by the so-called compromising attack: if an attacker captures a group of the tags, he gets access to all the keys from the root to their corresponding leaves, thus compromising some of the keys of the non captured tags.

In [5], Lu *et al.* proposed the *SPA* protocol which is able to defend against this attack by greatly reducing the key exposing probability when the tags are captured. This protocol considers that each tree node stores two (or, in general, a fixed number of) keys, of which a tag only stores one for each node of its path from the root. When a tag T is identified, the keys may be updated, but since the database maintains the old ones, other tags sharing nodes with T can still be correctly identified.

Ohkubo, Suzuki and Kinoshita in [1] proposed a scheme in which tags output the result of applying a hash function to its secret identifier and then change the identifier using a second hash function each time a tag is read. This approach solves the problems of information leakage, tracking and provides forward security. However, it presents a high computational complexity for the database during the identification phase, since it must compute all the possible hashes until it finds a coincidence.

This problem is partially solved in Avoine and Oechslin's scheme[2] using a time-memory trade-off, in which tag identifiers are precomputed and stored. The main problem of this approach lies in its scalability,

since the number of stored identifiers grows exponentially when the number of tags increases.

The reason that creates the need of extra resources in these two previous protocols is that any reader is allowed to read the tags. These extra, and unexpected, readings (from probably malicious readers) will cause the change of the internal secret of the tag in such a manner that the database will not know what is the expected output value for each tag at a certain moment. This uncertainty is responsible for the extra computations in Ohkubo *et al.*'s scheme and the extra memory in Avoine and Oechslin's scheme (they have to check all the possibilities until finding the correct one). This makes these two previous protocols confident but not scalable.

To solve these problems we propose a new protocol in which readers must be authenticated and hence, only valid readers will be allowed to read the tags. Our solution is based on elliptic curve cryptography, because it allows the use of fewer bits than conventional public key cryptography, making its implementation more feasible. Due to the intrinsic insecurity of the communication channel between the readers and the tags, a zero knowledge protocol is used to carry out readers authentication. Such a protocol allows the reader to prove the knowledge of a secret without revealing any information related to it. Thus, the previous authentications between the readers and tags cannot be reused by malicious readers who try to impersonate a valid one.

3 Preliminaries

As we mentioned earlier, our approach is based on elliptic curve cryptography and zero knowledge authentication, so, in this section we will review some preliminaries concerning these techniques.

(a) *Elliptic Curve Cryptography*[10,11]. An elliptic curve E over a finite field $\mathbb{F}_q$ consists of all the points $(x, y) \in \mathbb{F}_q \times \mathbb{F}_q$ that satisfy an equation of the form

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6,$$

with $a_i \in \mathbb{F}_q$, whose discriminant is non null, along with the point at infinity.

There is a point addition operation whose neutral element is the point at infinity. This set of points under this operation is an Abelian group. Therefore, a point $Q \in E(\mathbb{F}_q)$ can be multiplied by a scalar:

$$eQ = \underbrace{Q + \ldots + Q}_{e \text{ times}} = P.$$

The inverse problem (i.e., given P and Q, find an e such that $P = eQ$), called the Elliptic Curve Discrete Logarithm Problem (ECDLP), turns out to be computationally hard to solve. There are several cryptosystems[10,11], whose security is based on the intractability of the ECDLP problem.

The main interest of the ECDLP, compared to the ordinary DLP for multiplicative groups, is that there exist subexponential algorithms such as the index-calculus to solve the DLP on multiplicative groups, but they cannot be used to solve the ECDLP. Hence, it turns to be a harder problem. Under a practical point of view, it turns out that shorter keys can be used in the ECDLP while offering the same security as DLP.

Table 1 shows the number of bits needed for achieving the same security levels in cryptosystems that base its security on the DLP compared with those that are based on the ECDLP.

Table 1. Bits Needed for the Same Security Levels

DLP	ECDLP
512 bits	112 bits
1 024 bits	160 bits
2 048 bits	224 bits
3 072 bits	256 bits
7 680 bits	384 bits
15 360 bits	512 bits

(b) *Zero Knowledge Authentication*[12,13]. The purpose of zero knowledge protocols is to prove the knowledge of a secret without revealing any information about it.

In a zero knowledge authentication protocol an entity A (prover) has some secret s whose knowledge has to be proven to an entity B (verifier). Given the fact that only A is able to do this demonstration, B will accept this as a proof of the identity of A. Additionally, the protocol does not reveal any useful information, other than that A has knowledge of s, so neither B nor an eavesdropper will obtain any additional information of s and, because of that, they will not be able to impersonate A.

The idea is that A sends B a value, called witness, which defines a range of questions (or challenges) related to s that only the owner of s (A) can respond, but in such a manner that the answer does not reveal the secret or even part of it. B sends one of these challenges to A, who will send the response to B. The exchanged messages are typically dependant on random numbers. After these messages B either accepts, with certain probability of being correct, or rejects the proof of knowledge of s. This probability comes from the fact that an attacker can impersonate A if he is able to guess the challenge that B will send to him.

The protocol presented in this paper takes benefit of the usage of the elliptic curve version of Schnorr's zero knowledge protocol[13]. Notice that, since security is based on the ECDLP, small keys can be used, which fit the implementation restrictions of the passive RFID tags.

4 Protocol Description

The proposed protocol, sketched in Fig.2, uses the elliptic curve of Schnorr's protocol combined with a mechanism for updating the tag's secret. The idea is that a reader authenticates itself before receiving the tag's identifier, which is computed from a secret that is changed after each reading operation. The protocol consists of the following phases: (a) setup phase, (b) reader authentication phase, (c) tag identification phase, and (d) tag verification phase.

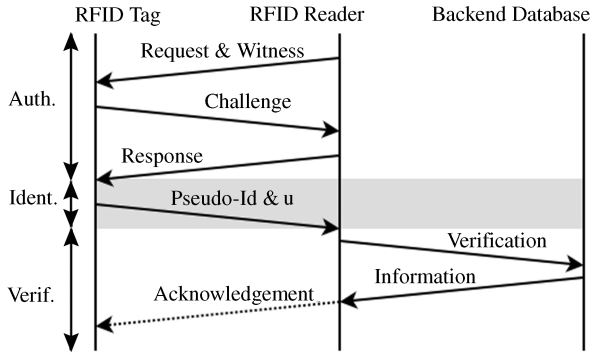

Fig.2. Protocol diagram.

(a) *Setup Phase.* In the setup of the system the following public and secret parameters are generated:

• a finite field $\mathbb{F}_q$ and an elliptic curve $E/\mathbb{F}_q$ defined over this finite field[14,15];

• a generator Q of a cyclic subgroup of points of the elliptic curve, of order g, where the ECDLP is infeasible;

• a secret $s \in [2, g-1]$, chosen as readers' secret, the knowledge of which must be proven to the tags for authentication;

• the public key $P \in E(\mathbb{F}_q)$ of valid readers, computed as $P = sQ$;

• the initial secret point $K_1^i \in E(\mathbb{F}_q)$ for each tag i, from which the pseudo-identifiers will be computed. This point will be changed at each reading.

(b) *Reader Authentication Phase.* For the purpose of avoiding impersonation of a valid reader, an authentication phase must take place before tag identification. This phase consists of these three steps:

1) the reader chooses a random value r (the commitment), with $2 \leqslant r \leqslant (g-1)$, computes $W = rQ$ (the witness), and sends W to the tag;

2) the tag chooses a random l-bit-length value $c \geqslant 1$ (the challenge) and sends it to the reader;

3) the reader computes $a = r + cs$ (the response) and sends it to the tag.

At this point the tag will accept that the reader is valid if $aQ - cP = W$ (notice that indeed $aQ - cP = (r + cs)Q - c(sQ) = rQ = W$).

(c) *Tag Identification Phase.* This process will take place every time the reader wants to read a tag, and it has been successfully authenticated. Each tag T_i keeps a secret elliptic curve point K_j^i, belonging to $E(\mathbb{F}_q)$, which will vary over time. This point will be changed each time the tag is successfully identified to prevent that two readings of the same tag can be related by an adversary. Every time a tag is read it will send its current pseudo-id, id_j^i, corresponding to tag T_i and the j-th reading.

The identification process of tag i at the j-th reading consists of these three steps:

1) The tag computes its pseudo-id as $id_j^i = LB(x(K_j^i)) * LB(y(K_{j-1}^i))$, where $x(K_j^i)$ and $y(K_{j-1}^i)$ are the abscissa and the ordinate of the current and previous secret points, respectively. $LB(\cdot)$ outputs some of the last bits of its input, and $*$ is an operation that is not algebraic over $\mathbb{F}_q$. This operation could be a modular addition if the field is binary or a bitwise *xor* if the field is prime. In the first reading, a seed is needed to act as $y(K_0^i)$. Subsection 5.1 will discuss the appropriate number of bits for id_j^i.

2) The tag computes its next secret point $K_{j+1}^i = zQ$, where z is the result of applying a certain function f to the abscissa of K_j^i ($z = f(x(K_j^i))$). Subsection 5.1 will discuss on the appropriate selection for function f.

3) The tag stores its new secret point and finally sends its pseudo-id, id_j^i, to the reader.

(d) *Tag Verification Phase.* At this point, the reader has already received id_j^i, so it has to access the database in order to verify the identity of the tag, as well as to obtain the information associated with it. For a successful identification the backend database has to store the outputs for all the tags, i.e., all the id_j^i for $i \in \{1, \ldots, n\}$ where n is the number of tags in the system. It also needs to store the corresponding secret points K_j^i. These values should be stored in a hash table for easy access. So, when a valid reader obtains a pseudo-id, it sends it to the backend database, which searches for it in the hash table, changes the corresponding secret point in the same manner that the tag does, removes the old pseudo-id from the hash table and inserts the one that will be obtained in the next reading. Finally, it sends product information to

the reader.

In the environments where it may be possible that readers do not receive the pseudo-id (because of noise or attackers blocking the signal), the tags may have been updated without the corresponding update of the database. The tag should wait for an *Acknowledgement* message before storing its new secret point to avoid this. Obviously, this message is also susceptible to not arriving, but this is a less dangerous problem, since in that case it would be the tag that needs to be updated. The only consequence of this fact is that the next reading of a tag would return an old pseudo-id, so the reading operation would have to be repeated until the tag reaches the pseudo-id that is stored in the database.

Nevertheless, an attacker could intercept and block the pseudo-id message and send himself the *Acknowledgement* to the tag. This would cause an update of the tag's secret without the corresponding update of the database, which is a serious problem. For avoiding this, the *Acknowledgement* message will contain the value $ack_j^i = MB(x(K_j^i)) * MB(y(K_{j+1}^i))$. $MB(\cdot)$ will output some of the middle bits, avoiding the bits previously used for the computation of the pseudo-id. This value will be computed by the database and it will be sent to the reader as part of the information message. The reader will send it to the tag, which can also easily compute it, so the tag will only accept an acknowledgement originated by a valid reader.

This protocol also admits the extension of sending data coming from a sensor, which makes it suitable for use in several scenarios. The protocol can be tuned to provide the tag with the ability to securely send small values apart from the pseudo-id (e.g., data generated by a sensor) to the reader. For example, Michelin will use RFID tags on its tires for monitoring tire pressure. This may be achieved with a very simple form of encryption, using the secret of the tag K_j^i as a key. For example, the additional data to be sent, v_j^i, can be encrypted using a bitwise *xor* with the first bits of the ordinate of the secret point: $u_j^i = v_j^i \ xor \ FB(y(K_j^i))$. Since these first bits will not be used in the computation of the pseudo-id (nor the acknowledgement), this may be considered a one-time pad, and, hence, secure. The encrypted data u_j^i can be sent in the same message with the pseudo-id as it is shown in Fig.2.

5 Implementation Issues

As mentioned in Section 1, strong restrictions on implementation have to be considered as we are dealing with small and cheap devices, as is the case of the passive RFID tags. In this section, we provide a specific selection for the length of the parameters to be used in the proposed protocol towards efficient and feasible implementation.

5.1 Selection of Parameters

There are some parameters that need to be accurately selected in order to achieve a good balance between the desired degree of security and the best performance.

(a) *The Finite Field, the Elliptic Curve and the Generator.* For the finite field $\mathbb{F}_q$ where $q = p^m$ with p prime, the usage of a binary finite field $\mathbb{F}_{2^m}$ is recommended. In fact, the expected security is similar for prime and binary finite fields but binary field arithmetic can be implemented more easily than the prime one[16].

Now, concerning the degree of the field, m, prime values should be selected for security reasons. The lastly solved ECC challenge is ECC2-109, which is defined over $\mathbb{F}_{2^{109}}$ (the Certicom ECC Challenge was introduced by Certicom in 1997 in order to evaluate the difficulty of the elliptic curve discrete logarithm problem). So, we propose the use of the field $\mathbb{F}_{2^{137}}$.

This implies that the keys used in the system will have 137 bits. Therefore, it is currently an acceptable level of security for this kind of applications, although bigger fields can be used in sensitive or long term systems.

Finally, according to Certicom[17], the elliptic curve should be selected with a cardinal of the form $\#E(\mathbb{F}_{2^{137}}) = h \cdot p$, where p is a big prime and $h \leqslant 4$ is the cofactor[14,15]. A point of prime order p should be chosen as generator.

(b) *Length of the Challenge*: l (*the Number of Bits of* c). For the challenge length l there are two considerations: firstly, l must be large enough to make negligible the probability of correctly guessing the challenge; secondly it is proved that the protocol is not zero knowledge for large l values[18]. These two considerations lead to $2^{-l} \approx 0$ and $q > 2^{2l}$. Thus, for $q = 2^{137}$ we recommend the usage of 40 bits, which gives a negligible probability of forgery of about 10^{-12}, while maintaining the zero knowledge property.

(c) *Length of the Pseudo-Id* id_j^i. For the pseudo-id id_j^i, computed by the tags, the goal is to minimize the probability that two tags have the same pseudo-id. Because of that, the number of bits must depend on n, the number of tags in the system. It should be greater than $2\log_2(n)$ to avoid birthday paradox collisions[18]. Note that if the length of the pseudo-id is very close to this value then the probability of having two tags with the same pseudo-id is about 50%. So, for instance, in a system with one million tags ($\approx 2^{20}$), the length of the

pseudo-id should be 48 bits, that would give a probability of having a collision below 1%.

Anyway, even in the rare case that a reading operation returned a pseudo-id that is the same for two (or more) tags, one can simply re-read the tags. Then, the values should be different.

(d) *The Function f to Obtain the z Value.* As mentioned in the tag identification phase in Section 4, a tag computes its next secret point K_{j+1}^i multiplying the generator Q by a factor $z = f(x(K_j^i))$. So, obtaining the previous secret from the current one turns out to be difficult, since it implies the computation of an elliptic discrete logarithm. This fact gives the property of forward security to our system (as will be seen in Subsection 6.2).

For the sake of efficiency, the function f should be selected in a manner that avoids large Hamming weights for z, assuring that the computation of zQ will be fast without compromising security.

A good function f for the generation of these values is the one that ensures the generation of a z with some certain maximum Hamming weight, for example, splitting the abscissa of K_j^i into many overlapped 7-bit fragments and using each of these fragments to select the activated bits of z.

If one of the fragments corresponds to a previously activated bit it may be deactivated, so the number of fragments is a maximum for the Hamming weight of z. This helps to decrease the computations of the tag, without affecting too much the security of the system.

To illustrate the computation of z, consider the example of the small finite field $\mathbb{F}_{2^{16}}$, and an hypothetic elliptic curve point $(54321, 9876)$, in binary $(1101010000110001_2, 10011010010100_2)$; if a factor z of maximum Hamming weight of 7 is needed it can be accomplished in the following manner: generating seven 4-bit fragments (shifting two bits each time) of the abscissa and activating the corresponding bits of z. This is shown in Table 2. In this case the final z value is $12347 = 8192 + 32 + 16 + 1 + 8 + 4096 + 2$ or in binary 11000000111011_2.

Table 2. Generation of z for a Small Example

Fragment	Bit	2^{bit}
1101_2	13	8 192
0101_2	5	32
0100_2	4	16
0000_2	0	1
0011_2	3	8
1100_2	12	4 096
0001_2	1	2

In a similar manner, for the field $\mathbb{F}_{2^{137}}$, choosing z with 7-bit fragments shifting two bits each time, gives factors of maximum Hamming weight 66 which should be considered secure.

5.2 Implementation Feasibility

Now, we will consider the implementation feasibility of our solution under current standard of implementation for the RFID tags. There are six main aspects to consider: the number of logic gates, the computation to be performed by the tag, the amount of data transferred between tags and readers, the time complexity to perform the protocol, the efficiency of the backend database, and the scalability of the system. The analysis of these aspects will be done using the parameters proposed in Subsection 5.1 for the protocol and its extension for sending data captured by some sensor.

(a) *Area Complexity.* According to the implementation requirements for elliptic curve processors that are reported in the literature[16,19], the implementation for the field $\mathbb{F}_{2^{137}}$ can be achieved using less than 10 000 logic gates. Taking into account that the current passive tags may have about 15 000 logic gates[1], we can state that the proposed protocol is feasible in terms of area complexity.

(b) *Tag Computations.* The most costly operations to be performed by the tag are the verification of the identity of the reader and the generation of the next secret point, involving three point multiplications: the first is aQ, the second is cP, whose factor is restricted to the range $1 \leqslant c < 2^{40}$, and the third is zQ, where z has a maximum Hamming weight of 66.

These computations may be faster if the points Q^{2^e} for $1 \leqslant e < 137$ (for the first and third multiplications) and the points P^{2^t} for $1 \leqslant t < 40$ (for the second multiplication) are stored in tag memory (if there is enough space).

Considering the factor space restrictions, but assuming that the mentioned points are not stored in tag's memory (since it would require additional gates), the expected time needed for the three multiplications is nearly 1.53 seconds[16]. The rest of tag computations: comparisons, bitwise xors, etc. are cheaper.

(c) *Communications.* The evaluation of communication concerns the length of the messages exchanged between the reader and tag during the protocol (see Fig.2).

The first message contains the witness W. It is sufficient to include the last 100 bits of the abscissa of the witness, since the protocol does not need that the entire witness is transmitted (for the verification of $aQ - cP = W$ it is sufficient to compare the last 100 bits of the abscissa of both parts of the equality). The

second message contains the challenge c of 40 bits. The third message has the response a of 137 bits. So, the reader authentication phase will need the transmission of $100 + 40 + 137 = 277$ bits.

The next message sent by the tag contains (id_j^i, u_j^i). Considering 64-bit-length sensor data to be encrypted (the size of a double precision floating point number), leads to a length of $48 + 64 = 112$ bits. Finally, the acknowledgement message ack_j^i must contain no more than 25 bits (note that, using 25 or less bits for this message allows the guarantee that the functions FB, MB and LB will not overlap).

Considering all the messages, the total amount of communication between the reader and the tag will be $277 + 112 + 25 = 414$ bits.

In this condition, the RFID tags with the minimum transmission rate of 520bps will need 0.8 seconds for communication, which fits the bandwidth restrictions on RFID systems, that should not take more than one second.

(d) *Time Complexity.* Table 3 shows the worst case times of the time-consuming computations of the protocol (those computations whose time is considered negligible are not shown) and the communications. These times are calculated according to the EC processors of [16, 19], considering a transmission rate of 520bps, and the points Q^{2^e} for $1 \leqslant e < 137$ and P^{2^t} for $1 \leqslant t < 40$ are not stored in memory.

Table 3. Protocol Worst-Case Times

Time-Consuming Operation	Time (s)
Receive W (100 bits)	0.19
Send c (40 bits)	0.08
Receive a (137 bits)	0.26
Compute aQ	0.77
Compute cP	0.22
Compute zQ	0.54
Send id_j^i, u_j^i (112 bits)	0.22
Receive ack_j^i (25 bits)	0.05

The global time of the protocol could be reduced by means of two feasible considerations. On the one hand, if the tag hardware admitted computing while receiving data, the computation of cP could be performed while receiving a (reducing the total time by 0.22s). On the other hand, the double-and-add algorithm for point multiplication could be modified, so that aQ and zQ are computed at the same moment (since they multiply scalars by a common point Q). This would reduce the time for these multiplications to approximately 0.4 second.

(e) *Database Efficiency.* The backend database has to store a record for each tag, with the next expected pseudo-id (48 bits) and the current secret point, which has $137 + 1$ bits (abscissa and one bit for selecting the ordinate), with some additional data, such as product type, caducity or similar. These records will be stored in a hash table indexed by the pseudo-id for easy accessing. As it is known, hash tables[20] are broadly used, since they permit constant time accessing and insertions, not depending on the number of items stored (tag IDs in that case).

Considering only the point and the pseudo-id, but not any additional data not related with the protocol itself, the memory needed by our solution for n tags is of order $O(n \log n)$ bits, because it must store n pseudo-ids each of them with approximately $2 \log_2 n$ bits, and the n fixed length secret points. This spatial cost is polynomial on the number of tags. A system with one million tags will need $1000000 \cdot (48+137+1)$ bits that is only about 23MB (perfectly reasonable in current systems).

(f) *Scalability.* In those protocols in which the number of changes of tags' IDs is not limited in order not to compromise forward security[1,2], the database lost control over the current identifier of each tag (since malicious readers could force the tag to refresh its values unnecessarily). So, the database had to keep information of long chains of possible identifiers. The advantage of our proposal is precisely that we do not need to compute or to keep long chains of identifiers for each tag, but only the next identifier and the current secret. Additionally, it does not have any restriction of tag reading operations, and we need only 23MB, for a one million tags system. All this implies that our system is highly scalable.

6 Security Analysis

In this section we present the security analysis of the system. For doing this, it is useful to recall the public and the secret information involved in the development of the protocol.

Public Information:

- $\mathbb{F}_q$: finite field;
- $E/\mathbb{F}_q$: elliptic curve over $\mathbb{F}_q$;
- $Q \in E(\mathbb{F}_q)$: generator of a subgroup;
- $P \in \langle Q \rangle$: public key of valid readers.

Secret Information:

- $s \in [2, g-1]$: secret of valid readers;
- $K_j^i \in \langle Q \rangle$: secret of tag T_i, at moment j;
- $LB(y(K_{j-1}^i))$: needed for the computation of the pseudo-id.

Secret Information per Round:

- $r \in [2, g-1]$: the random commitment (chosen by the reader);

- $z = f(x(K_j^i))$: tag's secret changing factor;
- v_j^i: tag's sensor value.

Public Information per Round:

- $W = rQ$: the computed witness (sent from reader to tag);
- c: the random challenge (sent from tag to reader);
- $a = r + cs$: the response (sent from reader to tag);
- $id_j^i = LB(x(K_j^i)) * LB(y(K_{j-1}^i))$: the pseudo-id (sent from tag to reader);
- $u_j^i = v_j^i$ xor $FB(y(K_j^i))$: the encrypted sensor value (sent from tag to reader);
- $ack_j^i = MB(x(K_j^i)) * MB(y(K_{j+1}^i))$: the acknowledgement (sent from reader to tag).

Concerning security, the typical scenario that is assumed in the literature[6] for RFID systems, considers two zones: (a) the secure zone conformed by the back-end database and the RFID readers, and (b) the insecure zone where there are RFID tags and their communication channels with the readers.

In the following subsections, we will define the basic types of attacks against this protocol in the communications that are established in the insecure zone, that must be analyzed[21]. Then, we will also evaluate the forward security property.

6.1 Types of Attacks

According to the literature there are five basic types of attacks that we must consider.

(a) *Sniffing.* In an sniffing attack the attacker eavesdrops the communications between a reader and a tag, trying to obtain useful information.

The only public information is that needed for the protocol setup, the reader's public key, the information sent during the reader authentication phase (the witness, the challenge and the response), the information sent in the identification phase (the pseudo-id and, optionally, the encrypted sensor data) and the acknowledgement.

With these data, if the attacker tries to guess the tag's secret K_j^i, the only information related to it is the pseudo-id (and the acknowledgement). But the bits of the pseudo-id are only the result of an operation, that is not algebraic, performed on the last bits of the abscissa and the ordinate of two different secret points, so it would be computationally unfeasible to obtain the secret from the pseudo-id (a similar reasoning applies to the acknowledgement).

If the attacker wants the reader's secret s, the only information related to it is the response. But, in that case, the secret will be multiplied by the challenge and added to a random commitment (r), which can only be obtained computing the discrete logarithm of the witness (which is computationally unfeasible).

So, an sniffing attack is useless due to the use of a zero knowledge protocol in combination with the use of a pseudo-id which cannot be related to the tag's secret and the encryption of the sensor data.

(b) *Tracking of the Tags.* The tag's tracking attack consists of the tracking of the behavior of the owner of a tag (for instance, if someone has an RFID tag in his mobile phone, the tracking of this tag allows the tracking of his behavior).

In this case, the only information to be considered is the pseudo-id (and the optional sensor data), since the challenge is random and the rest of the data is sent by the reader. A pseudo-id sniffed at a certain moment cannot be related to the information obtained before (or after), because it is generated using a tag's secret, that varies at each reading operation, in a non-invertible manner (the same reasoning applies to the encrypted sensor data).

(c) *Spoofing.* A spoofing attack is the impersonation of one of the entities of the system. We have to consider two different types of this attack.

- *Impersonation of a Reader*

Due to the use of a zero knowledge authentication protocol, the probability of impersonation of a reader is negligible.

- *Impersonation of a Tag*

An attacker willing to impersonate a tag needs the current secret K_j^i of the tag. Such a value cannot be obtained from the public information or the communications established during the execution of the protocol.

It should be noted that, if an attacker physically obtains the secret of a tag, but returns the tag to the system without altering it, and impersonates the tag before a valid reader reads the real one, then the real tag's secret will be obsoleted, since the database will change the expected pseudo-id. This allows a denial of service for this tag, but considering that the attacker needs physical access to the tag for doing this, he could also have destroyed the stolen tag directly.

In both cases, for spoofing attacks, the adversary must obtain the secret of the element to impersonate.

(d) *Replay Attacks.* A replay attack consists of the attacker's resending information that he had captured before, and eavesdropping a previous session.

On one hand, it is useless to reuse the witness, the challenge or the response, since it is a zero knowledge authentication protocol. On the other hand, a pseudo-id cannot be reused, because the database will wait for the next pseudo-id of any tag. So, if an attacker reuses a pseudo-id the reader will not recognize the value as a

valid one.

(e) *Denial of Service.* Finally, a denial of service consists of a temporal (or permanent) incapacity of the system or a part of it.

Since a tag only changes its secret K_j^i if a reader has successfully authenticated, there is no danger that an attacker performs a denial of service attack of multiple reading requests.

As said in Subsection 4(d) the additional acknowledgement message may be needed in some environment, to avoid the problem of having the database obsoleted.

Obviously, as in any wireless system, a malicious reader generating a bulk of requests could provoke the result that the requests from valid readers would be denied. In general, wireless systems cannot be protected against this.

6.2 Forward Security

The property of forward security ensures that the revelation of tag secret information will not put in danger the security of previously sent information.

We will assume that the tag has some kind of sensor, and that the sensor data $v_1^i, v_2^i, \ldots, v_{j-1}^i$ has been securely sent in previous reading operations. We also assume that the attacker knows all public parameters as before, and has eavesdropped all the $u_1^i, u_2^i, \ldots, u_{j-1}^i$, the pseudo-ids and all the other parameters sent during the execution of the protocol.

Let us assume that the adversary is able to physically attack tag T_i and obtain its current secret point K_j^i. But in order to decrypt the u values, he needs the previous $K_{j-1}^i, K_{j-2}^i, \ldots, K_1^i$, and they cannot be easily obtained because it implies solving one elliptic curve discrete logarithm for each secret point he wants to obtain (and inverting the non-bijective function f).

As a proof of the forward security, consider the problem of obtaining the previous secret K_{j-1}^i from the current one.

Let us consider an attacker with knowledge of all the public parameters and the current secret parameters of a stolen tag, in particular: the field $\mathbb{F}_q$, the elliptic curve $E/\mathbb{F}_q$, the generator $Q \in E(\mathbb{F}_q)$, and the tag's secret $K_j^i \in E(\mathbb{F}_q)$. Recall that $K_j^i = zQ = f(x(K_{j-1}^i))Q$. We will prove that if this attacker has access to an oracle which returns K_{j-1}^i from these data, he can easily compute elliptic curve discrete logarithms, so obtaining the previous secret from the current one is not easier than computing elliptic curve discrete logarithms.

Suppose the attacker wants to compute $z = \log_Q T$. For obtaining it, he first uses the oracle, which returns a point K, corresponding to a hypothetic previous secret point of T, that is, $T = zQ = f(x(K))Q$. Then, he easily computes, from K, $z = f(x(K))$, which is the desired discrete logarithm.

So, this communication has the property of forward security, i.e., for an attacker trying to decrypt some of the values that he might have eavesdropped, it is useless to obtain the current secret of the tag, because obtaining the previous secret keys will provide a solution to the ECDLP, which is infeasible.

7 Conclusions

This paper proposes a confident protocol to preserve privacy in RFID tag interfacing. The protocol uses cryptographic techniques, namely elliptic curves and the Schnorr identification to construct an interactive protocol whereby:

1) authorized readers are securely identified;
2) the tag identifiers are securely updated to prevent tracking;
3) additional data coming from some sensor can be securely transferred from the tag to the reader.

To meet the implementation restrictions of RFID tags, we propose an accurate selection of the parameters involved in the protocol.

Finally, our approach is shown to be scalable on the number of tags in the system; and a security analysis is provided that proves the confidence of the system and that it provides forward security.

References

[1] Ohkubo M, Suzuki K, Kinoshita S. Cryptographic approach to "Privacy-Friendly" tags. In *RFID Privacy Workshop*, MIT, MA, USA, November 2003.

[2] Avoine G, Oechslin P. A scalable and provably secure hash based RFID protocol. In *Proc. International Workshop on Pervasive Computing and Communication Security* (*PerSec 2005*), Kauai Island, Hawaii, USA, IEEE, IEEE Computer Society Press, March 2005, pp.110–114.

[3] Juels A, Pappu R. Squealing euros: Privacy protection in RFID-enabled banknotes. In *Proc. Financial Cryptography* (*FC'03*), Rebecca N Wright (ed.), Le Gosier, Guadeloupe, French West Indies, IFCA, *LNCS* 2742, Springer-Verlag, January 2003, pp.103–121.

[4] Kinosita S, Hoshino F, Komuro T, Fujimura A, Ohkubo M. Nonidentifiable anonymous-ID scheme for RFID privacy protection. *Joho Shori Gakkai Shinpojiumu Ronbunshu*, 2003, (15): 497–502.

[5] Lu L, Liu Y, Hu L, Han J, Ni L M. A dynamic key-updating private authentication protocol for RFID systems. In *Proc. International Conference on Pervasive Computing and Communications* (*PerCom 2007*), New York, USA, IEEE, IEEE Computer Society Press, March 2007, pp.13–22.

[6] Juels A. RFID security and privacy: A research survey. Manuscript, September 2005.

[7] Weis S A, Sarma S E, Rivest R L, Engels D W. Security and privacy aspects of low-cost radio frequency identification systems. In *Proc. International Conference on Security*

in Pervasive Computing (*SPC 2003*), Hutter D, Müller G, Stephan W, Ullmann M (eds.), *LNCS* 2802, Boppard, Germany, Springer-Verlag, March 2003, pp.454–469.
[8] 860MHz–960MHz class I radio frequency identification tag radio frequency and logical communication interface specification proposed recommendation, version 1.0.0. Technical Report MIT-AUTOID-TR-007, Auto-ID Center, November 2002.
[9] Molnar D, Wagner D. Privacy and security in library RFID: Issues, practices, and architectures. In *Proc. Conference on Computer and Communications Security* (*ACM CCS*), Pfitzmann B, Liu P (eds.), Washington DC, USA, ACM, ACM Press, October 2004, pp.210–219.
[10] Koblitz N. Elliptic curve cryptosystems. *Mathematics of Computation*, 1987, 48: 203–209.
[11] Miller V S. Use of elliptic curves in cryptography. In *Proc. Advances in Cryptology* (*CRYPTO'85*), *LNCS* 218, Springer, 1986, pp.417–426.
[12] David Chaum, Jan-Hendrik Evertse, Jeroen van de Graaf, René Peralta. Demonstrating possession of a discrete logarithm without revealing it. In *Proc. Advances in Cryptology* (*CRYPTO'86*), Santa Barbara, USA, August 1986, pp.200–212.
[13] Schnorr C P. Efficient signature generation by smart cards. *Journal of Cryptology*, January 1991, 4(3): 161–174.
[14] Martínez S, Tomàs R, Roig C, Valls M, Moreno R. Parallel calculation of volcanoes for cryptographic uses. In *Proc. the 20th IEEE International Parallel & Distributed Processing Symposium* (*IPDPS*), *Workshop on Parallel and Distributed Scientific and Engineering Computing* (*PDSEC*), Rhodes Island, Greece, April 25–29, 2006, p.8.
[15] Miret J, Moreno R, Sadornil D, Tena J, Valls M. An algorithm to compute volcanoes of 2-isogenies of elliptic curves over finite fields. *Applied Mathematics and Computation*, 2006, 176(2): 739–750.
[16] Batina L, Guajardo J, Kerins T, Mentens N, Tuyls P, Verbauwhede I. An elliptic curve processor suitable for RFID-tags. Cryptology ePrint Archive, Report 2006/227, 2006.
[17] SECG. SEC 2: Recommended elliptic curve domain parameters. Standards for Efficient Cryptography Group, Certicom Corp., September 2000.
[18] Menezes A J, van Oorschot P C, Vanstone S A. Handbook of Applied Cryptography. CRC Press, 1996.
[19] Batina L, Guajardo J, Kerins T, Mentens N, Tuyls P, Verbauwhede I. Public-key cryptography for RFID-tags. In *Proc. International Workshop on Pervasive Computing and Communication Security* (*PerSec 2007*), New York, USA, IEEE Computer Society Press, March 2007, pp.217–222.
[20] Ward Douglas Maurer, Theodore Gyle Lewis. Hash table methods. *ACM Comput. Surv.*, 1975, 7(1): 5–19.
[21] Rieback M R, Crispo B, Tanenbaum A S. The evolution of RFID security. *IEEE Pervasive Computing*, January–March 2006, 5(1): 62–69.

Santi Martínez received the B.S. and M.S. degrees in computer science from the Universitat Rovira i Virgili (URV), Spain, in 2002 and 2004, respectively. He is currently a Ph.D. candidate of computer science at the Universitat de Lleida (UdL), Spain. His research interests include RFID systems, cryptography and parallel processing.

Magda Valls received the B.S. in Mathematics from the Universitat Autònoma de Barcelona (UAB), Spain, in 1994, and the M.S. and Ph.D. degrees in applied mathematics from the Universitat Politècnica de Catalunya (UPC), in 1999 and 2001 respectively. She is currently an associate professor of mathematics at the Universitat de Lleida (UdL), Spain. Her research interests include cryptography, computational security and elliptic curve cryptosystems.

Concepció Roig received the M.S. and Ph.D. degrees in computer science from the Universitat Autònoma de Barcelona (UAB), Spain, in 1996 and 2002 respectively. She has been an associate professor in the Department of Computer Science at the Universitat de Lleida (UdL), Spain, since 1992. Her current research interests include parallel and distributed systems, modelling of parallel applications, task graph models, task mapping algorithms, heterogeneous computing, security and privacy in distributed systems.

Josep M. Miret received the M.S. degree in mathematics from the Universitat de Barcelona (UB), Spain, in 1983 and the Ph.D. degree in applied mathematics from the Universitat Politècnica de Catalunya (UPC), Spain, in 1999. He is currently an associate professor of mathematics at the Universitat de Lleida (UdL), Spain. His research interests include cryptography, enumerative geometry, elliptic and hyperelliptic curve cryptosystems and rational projective plane cubics.

Francesc Giné received the B.S. degree in telecommunication engineering from the Universitat Politècnica de Catalunya (UPC), Spain, in 1993 and the M.S. and Ph.D. degrees in computer science from the Universitat Autònoma de Barcelona (UAB), Spain, in 1999 and 2004, respectively. He is currently an associate professor of computer architecture at the Universitat de Lleida (UdL), Spain. His research interests include cluster, multicluster and peer-to-peer computing and scheduling-mapping for parallel processing.