RESEARCH ON SOLUTIONS FOR PREVENTING ALGEBRAIC ATTACKS AGAINST BIOMETRIC AND RFID PROTOCOLS

Mihailescu Marius Iulian

Abstract. In this paper we will go through different algebraic problems. Here are presented authentication, secrecy and untraceability attacks. Many vulnerability of different protocols have not been published, starting from this aspect, the paper will demonstrate some attacks making references to other wrong protocols. Malicious persons, executing different attacks, are abusing the characteristics of operators which are engaged by the protocols.

2000 *Mathematics Subject Classification*: 68Q65, 68M14./Subject Classification for Computer Science.

1. Introduction

Radio Frequency Identification (RFID) is expected to become a valuable key technology in supply chain management, because it has a large potential to save costs. When we concentrate to prove the security of cryptographic protocols, we need to take in considerations the fact that is based on formal languages, considering that protocol messages will be on a high level of abstraction, therefore missing the implementation details. This paper presents different algebraic verification methods which from my personal point of view represents a combination of two aspects presented above. Here we see the evaluation process of the security of protocols by taking into consideration the free term algebra which is generated by the messages that were exchanged between the most important protocols and acted on by the standard proposed by DolevYao adversary [1]. This article takes into considerations cryptographic primitives, such as hash functions and encryptions to be perfectly. When we are referring to the computational aspect, we focus on how much information is compromised by the security flaw through terms in which operators with algebraic properties are applied.

This study wont focus on proving protocols that are secure; the primary goal is to study and understand how the properties of algebraic operators and functions can be used in communication protocols to identify the reason that can make these protocols go wrong, and also to achieve the security targets. Following this goal, the paper presents three categories of vulnerabilities discovered by the state of the art and analysing recently published RFID protocols. Investigating the algebraic characteristics can be a very useful tool in discovering vulnerabilities in RFID protocols. The resource constraints imposed on RFID tags have led to a congestion of proposals for protocols engage XOR, cyclic redundancy check functions, modular addition, and custom-made hash like functions. Trying to demonstrate that all such protocols are secure using a computational security model is boring and cannot be justified, because a significant and important number of protocols that were proposed turn out to be wrong. Automated tools, based on formal methods boarding currently, fail to verify the security of the most protocols, because they cannot verify some of the desired security characteristics, such as untraceability of tags, or don‘t consider flaws related to partial leakage of keys. While our boarding is not automatic in general, the automatic detection process of attacks will need to become necessary in the anticipated future. The types of attacks presented in this paper are what we call algebraic continuation attacks. The targets of the challenge-response mechanism in authentication protocols are mentioned in Section 3, attribute acquisition attacks on untraceability of tags are mentioned in Section 4, and cryptanalytic attacks on secrecy of keys and tag identities presented in Section 5.

2.State of the art

2.1. Notations and conventions

A reader will take into consideration the actual RFID reader as the same as the potential database or server communicating with the reader, because in all protocols that we consider, the communication is done over a secure channel. An agent can have two identities, a tag or a reader, while a role refers to the protocol steps a tag or reader is expected to carry out. A run represent the execution of a role by an agent. For our intuition and convenience, we‘ll make a reference to different specific attacks which take place on protocols as quality-time attacks. These are attacks in which the adversary interacts with a tag in absence of a sincerely or trusted RFID reader. The point of such attack is to send very carefully challenges to the tag with the scope to obtain different types of vital information and which later will play the role of a reader or the tag, trace the tag or to attack any other type of security requirement of a protocol. The quality-time attacks are very common in the mobile and wireless natural structure of RFID tags. The attacks can be realized on tags that happen to be in the neighbourhood of an adversary for a short period of time or on tags where the attacker is able to isolate them from their nature for an extended period of time. In this study the things are simplified for presentation of protocols whenever is possible by leaving out the non-important steps, terms, and communication. The description gives enough to reconstruct the attacks on the original protocols. When we are discussing about the non-possibility of traceable property of a protocol, we understand the fact that the tags cannot be traceable. For the readers comfort, is very important how the describing protocol is done. For example, we frequently use the following notations:

1. k for a shared secret key;
2. h for hash functions;
3. $r_1, \ . \ . \ . \ , r_n$ for nonce;
4. *ID* when we refer to tags ID.

There will be some special cases when additional and variables are needed, and then we use the notation proposed by the authors of the protocol. When an attack is composed of several runs, the terms used in a second run are primed. In this article I have decided to represent the protocols in a graphical way using UML sequence charts, such as in Figure 1. Each message that is shown in the diagram illustrates the role names, framed near the top of the chart. Above the role names, the roles secret terms are shown. A box represents actions, such as nonce generation, computation, and assignments. Arrows which connect roles represent the messages that are send and are expected to be received are specified above. An agent will continue its execution process only if it receives a message according to the specifications. Other conditions that need to be accomplished are illustrated using a diamond box. For an example, in Figure 1, the role names are identified by R and T, both are known as k and *ID*, which are the secret terms. R will generate the nonce r_1 before sending the first message. When the message is received, T will generate a nonce and computes the response. The reader will accept the response only if the condition of finding the pair k, ID is satisfied; the pair will generate the same term when the computation process shown is applied to it. The reader will continue by computing and sending the last message in the case if the response is accepted.

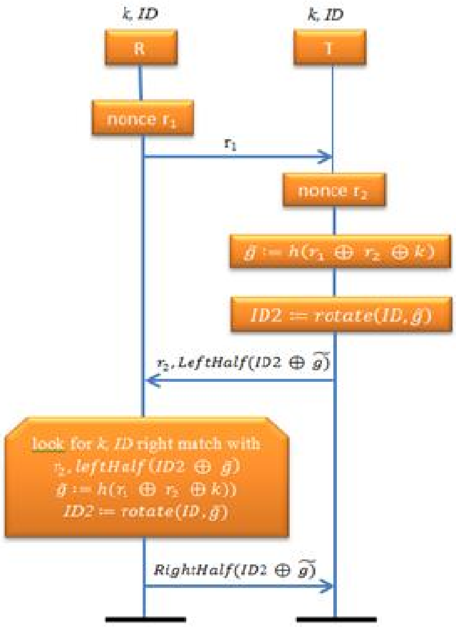
k, ID
k, ID
R
T
nonce r_1
r_1
nonce r_2
$\tilde{g} := h(r_1 \oplus r_2 \oplus k)$
$ID2 := rotate(ID, \tilde{g})$
$r_2, LeftHalf(ID2 \oplus \tilde{g})$
look for k, ID right match with
$r_2, leftHalf(ID2 \oplus \tilde{g})$
$\tilde{g} := h(r_1 \oplus r_2 \oplus k))$
$ID2 := rotate(ID, \tilde{g})$
$RightHalf(ID2 \oplus \tilde{g})$

Fig. 1 Wrong (flawed) authentication protocol

2.2. Security Characteristics and Properties. Adversary Models

This section starts with the Gavin Lowe‘s study [2], in which he suggest that ,, *an appropriate authentication requirement will depend upon the use to which the protocol is put, and identify several possible definitions of authentication Gary Lowe.* In the article of Lowes authentication hierarchy [2], he takes into consideration recent aliveness to be the most appropriate authentication requirement for RFID protocols. Recent aliveness catches the fact that the tag needs to have generated a message as a result of a readers query. We take into consideration the notion of untraceability which is defined by Van Deursen et al. [3] in which they put the accent on the fact that a tag is untraceable if, for any two protocol that are running, a person cannot tell whether the same tag was executing both runs or two different tags were executing the runs. In the end, terms that cannot be found in the adversarys knowledge are said to be secret.

3. Attacks on Authentication based on Algebraic Replay

Many studies put the accent on the common way to authenticate RFID tags by means of the following challenge response mechanism. The RFID reader sends challenges to the tag using a nonce r_1 on which the tag gives a reply with a term derived from the nonce r_1; some information is identifying the tag, and potentially a nonce r_2 is generated by the tag. If present, the nonce r_2 serves as the tags challenge to the reader in mutual authentication protocols or as a blinding term to obtain tag untraceability. We can represent the tags reply to the readers challenge as the pair r_2, *g(*r_1*,* r_2*, s)* with the understanding that r_2 may be constant or empty. The reader verifies the authenticity by applying the inverse of the function g to the term and checking whether the response contains r_1 and a valid s. If g is a one-way function then the reader verifies the authenticity of the tag by computing the function *g(*r_1*,* r_2*, s)* and comparing it to the received value. The reader can calculate this function, since it‘s generated by value r_1 itself; the value r_2 is supplied by the tag, and the reader has a database with values of s for every tag it may authenticate. We now argue that the following two properties are necessary in order for the challenge-response mechanism to guarantee recent aliveness of the tag.[12]

Property For fixed r_2 and s the range of the function $r_1 \rightarrow$ *g(*r_1*,* r_2*, s)* must be large. More precisely, given r_2, s, the adversarys advantage in guessing *g(*r_1*,* r_2*, s)* correctly for an unknown, randomly chosen r_1 must be negligible.

ARR Let O_s(x) be an oracle which upon input x randomly chooses y and returns y and *g(x, y, s)*. If s is unknown, then given access to a polynomial number of queries $O_s(x_1)$,...,$O_s(x_l)$ to the oracle is not feasible. If the property is satisfied, then, as stated, the probability of the adversary guessing *g(*r_1*,* r_2*, s)* is negligible. Thus with overwhelming probability, a response r_2, *g(*r_1*,* r_2*, s)*, to the readers challenge r_1 must have been generated after the challenge was sent. This property is obviously necessary for recent aliveness and, in particular, excludes classic replay attacks. The **ARR** (algebraic replay resistance) property guarantees that there is no efficient algorithm to compute a response r_2, *g(*r_1*,* r_2*, s)* to the challenge r_1 even after having observed previous challenge-response pairs. Clearly, an attackers ability to compute such a response violates recent aliveness and this property is thus necessary

for it. Such an attack generalizes replay attacks instead of merely replaying previously observed information; the attacker combines previously obtained challenge-response pairs to compute the response to a fresh challenge. Hence, we refer to attacks on challenge-response authentication protocols exploiting the lack of the ARR property as algebraic replay attacks.
It is obvious that for a function $g(r_1,r_2,s)$ to have the ARR property, it must preserve the secrecy of *s*. Indeed, cryptographic hash functions are frequently used for the type of challenge-response mechanism considered here. Since the collision resistance property of cryptographic hash functions does not seem necessary for the challenge-response mechanism, the question arises whether all one-way functions satisfy the ARR property and the answer is negative. It is certainly false for all homomorphic one-way functions. Consider, for instance, the Rabin function, defined by $x \rightarrow x^2$ *mod N* for certain composite integers *N*. If $(r_1,r_2,\ s) \rightarrow g(r_1,r_2,\ s) = (r_1\ r_2\ \mathrm{s})$ *mod N* is a Rabin function, then given only one challenge-response pair, r_1, $g(r_1,r_2,\ s)$ it is easy to compute responses for any challenge $r^{`}_1$, since $\mathrm{g}(r^{`}_1,\ r_2,\ \mathrm{s}) = \mathrm{g}(r_1,\ r_2,\ \mathrm{s}) \cdot (r^{`}_1/r_1)^2$. Furthermore, even non-homomorphic one-way functions will, in general, not have the ARR property if their argument has algebraic properties. As demonstrated in the examples below, there are several protocols that fail to achieve recent aliveness for this very reason. In these protocols the challenge-response construction can typically be represented as $g(r_1,r_2,\ s) = f(n\circ\ r_2,s)$, where *f* is a (non-homomorphic) cryptographic hash function and $\circ$ denotes an operator with the following algebraic property. Given *a*, *b*, and *c*, it is easy to find *d* with $\mathrm{a} \circ \mathrm{b} = \mathrm{c} \circ \mathrm{d}$. This construction clearly does not have the ARR property, regardless of the properties of *f*. The algebraic replay attack on such a protocol works as follows. An adversary observing one execution of the protocol learns r_1, r_2, and $f(r_1\ f(r_1 r_2, s)\ r_2,s)$. When challenged with $r^{`}_1$, the adversary finds $r^{`}_2$ such that $r_1 \circ r_2 = r^{`}_1 \circ r^{`}_2$ and replies with $r^{`}_2$, $f(r_1 \circ r_2,s)$. The attack succeeds because $f(r_1 \circ r_2,\ s) = f(r^{`}_1 \circ r^{`}_2\ ,s)$. Examples of operators $\circ$ for which this type of attack succeeds are *xor*, *modular addition*, and any *associative operator* for which it is easy to compute left inverses.

3.1. Examples of ∘ operators. New attacks

In this section I have introduced the most recent examples of algebraic replay attacks and also where to be founded. Another important aspect which is presented in this section is the new attacks.

1. In article [33], Lee et al. describes with many details in Section 4 his

protocol, which is very vulnerable to an algebraic replay attack in which an adversary needs to observe three protocol executions or perform a quality-time attack composed of three queries. The execution of algebraic replay attack can be solved by a small system equations yielding. This type of attack has been described first by Bringer et al. [32].

2. The mechanism challenge-response proposed by Chien and Chen [9] is composing the *xor* with cyclic redundancy check (CRC). When we have a challenge r_1, the tag will respond with r_2,CRC(EPC,r_1,r_2)⊕k, where *EPC* represent a constant which identifies the tag. In the work of Peris-Lopez et al. [10] is presented the attack on the protocol. We can see *CRC* is homomorphism, i.e. CRC(a)⊕CRC(b)= CRC(a⊕b).

The article presents a complete attack on the protocol which is proposed by Chien and Huang [28], shown in Figure 1 above. As recommendation, the reader should read the full version of the paper [31] for detailed attacks on the other protocols. The reader *R* and tag *T* share secrets *k* and *ID*. The reader starts by sending a random bit string r_1. The tag generates a random string r_2 and hashes the *xor* of r_1, r_2, and the secret *k*. This hash and *ID* are used as input for a function in which the *ID* is rotated by a value depending on the hash. The tag computes the *xor* of the rotated *ID* and the hash, before sending the left half of the resulting bits and r_2 to the reader. The reader performs the same operations on every pair of *ID* and *k* until it finds the corresponding tag. It then sends the right half of the corresponding bits to the tag. To play the role of a tag, it is enough to observe that the tag response to the readers challenge depends only on $r_1 \oplus r_2$ and a shared secret. The composition process of functions applied to the *xor* and shared secret can be represented by the function *f*, which we‘ve defined above. So, the adversary can outcome with a quality-time attack by sending a challenge to a tag with any r_1 to have a valid combination of r_1, r_2 and *Left(ID2* $\oplus \tilde{g}$*)*. These types of information are enough for the adversary to be able to respond to any future challenge $r_1^{‘}$ received from a reader.

4. The LD Protocol Description

The LD protocol [14] has been developed as a mutual authentication protocol for re-writable RFID tags guaranteeing the unlinkability of tags in supply chain and not only. Each supply chain consists of a chain of partners, each being represented by a reader. Each reader R_i contains a secret k_i, as well as the secret of the next reader $k_{(i+1)}$. In addition, every reader stores the identity c of each tag that could authenticate. Each tag T contains a pseudonym α that represents the identity which is temporary. The value of is equal to c $\oplus$ k_i where k_i represents the secret of one of many readers R_i which currently allow to identify and authenticate the tag.

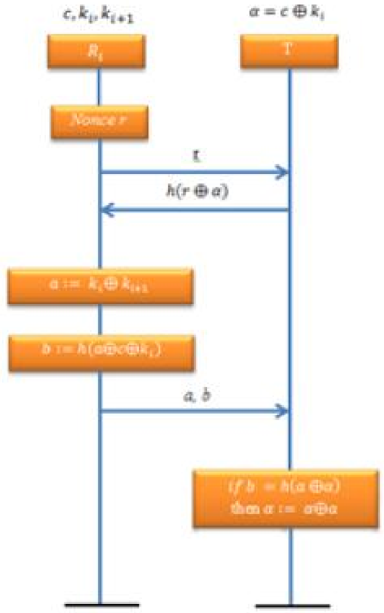
c, k_i, k_{i+1}
$\alpha = c \oplus k_i$
R_i
T
Nonce r
r
$h(r \oplus \alpha)$
$a := k_i \oplus k_{i+1}$
$b := h(a \oplus c \oplus k_i)$
a, b
if $b = h(a \oplus \alpha)$
then $\alpha := \alpha \oplus a$

Fig. 2 RFID protocol for supply chains

As we have figured out the *LD* protocol is a challenge-response based protocol. The reader R_i sends a challenge tag T with a nonce r. The next step consists in calculation of the *xor* for its current secret α and will challenge r and respond with the digest of this value. The reader will consider the tag authentic if he'll find a secret c for which *h(*$c \oplus r \oplus k_i$*)* is equal with the received response. At this moment, the reader can stop the protocol execution giving the possibility to authenticate the tag again, in a future communication process session. We have an alternatively aspect, the reader also can send the update $a = k_i \oplus k_{(i+1)}$, accompanied by b=*h(*$a \oplus c \oplus k_i$*)* to the tag. The tag will then verify if b=*h(*$a\oplus \alpha$*)* and update its secret α by applying *xor* with a. By doing this, the owner of tag T will be transferred from the reader R_i to the reader $R_{(i+1)}$. This protocol is illustrated in Figure 2.

5.Unlinkability

One negative aspect of the LD protocol is the fact that wasnt designed to be untraceable, but only unlinkable. It‘s real the fact that a tag will not introduce any randomness in its response to a query that comes from a reader which implies that a tag is traceable between key updates. We‘ll see in the next sections that the protocol will not provide unlinkability either by exhibiting on this property. The discussion on this section is based on the analysis of LD in [14].

5.1. Attack analysis

The first step is to demonstrate that the protocol doesn‘t satisfy unlinkability. For this is enough to present a scenario in which the adversary recognizes a previously observed tag after the tag that has updated its secret. The problem is difficult in the case of eavesdropping on a valid authentication session. Between a tag and a reader, the adversary learns r,h(r$\oplus$ α),a and b. In the end of the execution, the tag will update its secret by replacing with $r^{\text{‘}}$=r$\oplus$a, to which the tag will respond with $h(r^{\text{‘}} \oplus \alpha^{\text{‘}})$. Using a simple algebraic property of *xor*, the response will be equal with the previous that we have observed:

$$h(r^{\text{‘}} \oplus \alpha^{\text{‘}}\ ) = h(r\oplus \alpha\oplus\ a) = h(r \oplus\alpha) \ (1)$$

In this context, we refer to the property of xor in equation (1) named as give up property, because of the evident reasons. In Figure 3 we can see how the attack is realised. We can assume that a malicious person is not able to get close enough to a tag while it‘s being updated on a good person‘s premises, in conclusion unlinkability can be plausibly broken. This way, we may assume that an adversary can and it is able to eavesdrop on the readers messages. Furthermore we can assume that the malicious persons can query incoming and outgoing products while they are outside the restricted area of a good person. So, the following small extension of the above attack becomes then very real and plausible in the supply chain scenario. The attacker will generate a nonce r and will query all incoming products with this value, observing the readers key-update messages a, b, and query all outgoing tags with a$\oplus$r. Lets stop again on equation (1) where the eavesdropping on messages from reader to tag it‘s enough to be able to match the incoming product‘s responses with (to) the outgoing product‘s responses and thus link the products. We have found the same type of flaw in several other protocols. The last message from the reader to the tag in the protocols in [22, 17, 12] are containing the actual value in which the tag should update its key. The message can be observed and used by the adversary to break unlinkability.

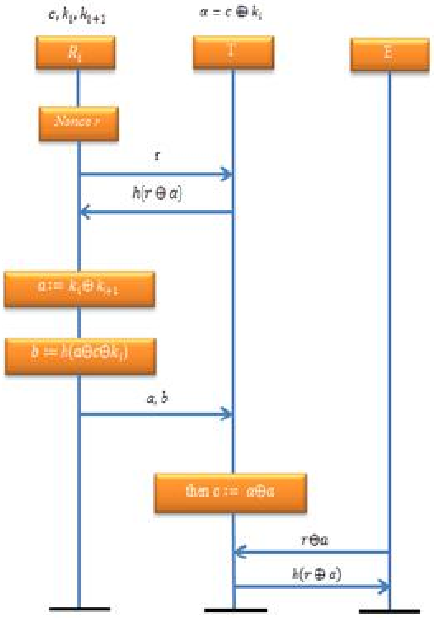

Fig. 3 Illustration of Unlinkability Attack

5.2. The weak point in the security proof process

As we can see in [14] section 4.4, we have to take in consideration that the adversary can choose the same nonce $r = r^{‘}$ for the challenge before a tag is updated and after the tag is updated. It is demonstrated that in this case the adversary cannot link the two responses $t{=}h(r{\oplus}c{\oplus}k)$ and $t^{‘} = h(r{\oplus}\ c^{‘} \oplus k^{‘})$ without having any knowledge of the keys. So, setting $r = r^{‘}$ is not the best tactic and solution for the adversary. Lets see why:

$$\begin{gathered} h(r{\oplus}c{\oplus}k){=}h(r^{‘} \oplus c^{‘} \oplus k^{‘}) \\ \Leftarrow r{\oplus}c{\oplus}\ k = r^{‘} \oplus c^{‘} \oplus k^{‘} \\ \Leftarrow r^{‘} = r{\oplus}\ k \oplus k^{‘} \wedge c = c^{‘} \end{gathered}$$

setting $r^{‘}$=r $\oplus$ k $\oplus$ $k^{‘}$ represents a better choice for the adversary. Remember that k and $k^{‘}$ don‘t have keys of successive readers. Only observing the updating process of keys for a chain of readers R,. . .,$R^{‘}$ that are sending to tags, the malicious persons can calculate k$\oplus$ . . . $\oplus$ $k^{‘}$ = k $\oplus$ $k^{‘}$.

5.3. Presenting the Solution

The mistake in the LD protocol which is affecting unlinkability owes itself to the algebraic property of the xor operator, shown in equation (1). This mistake may be avoided if the concatenation of the terms that can be found inside the hash functions is used, instead of the *xor* operator, if the reader sends $h(a,(c\oplus\ k_i))$ instead of $b=h(a\oplus c\oplus\ k_i)$, where the comma represent the concatenation process. This concatenation makes the calculation of the hash function more expensive for the tag, represent much stronger alternative. Take into consideration the fact that this improvement reduces the *xor* give up attack on unlinkability.

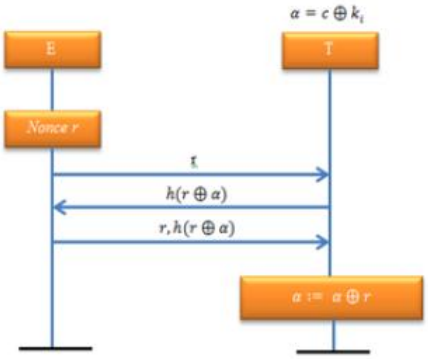

Fig. 4 Illustration of attack on authentication

6. Authentication

As we have presented in the above section, the LD protocol needs to give the opportunity to have an tag-to-reader authentication, with the purpose of ensuring that only authentic products are accepted by a good persons reader and reader-to-tag authentication, with the scope to guarantee that the tags can only update their keys after the communication with a real and legitimate reader is done.

6.1. Attack Analysis

As we have seen in the protocol message the reader sends a, b to the tag, where $a = k_i \oplus k_{(i+1)}$ and *b=h(a*$\oplus$ α*)*. We can see that the tag cannot check the value of *a*, since k_i and $k_{(i+1)}$ are stranger for the tag. The tag verifies only the fact that the value of *b* is for real equal with *h(a*$\oplus$*)* and then the possibility of updating its secret α by applying it together with *a*. We have to remind the fact that a malicious person can produce a valid combination of a and b and successfully play the role of a reader. The malicious person sends a challenge to the tag with the *r* to obtain *h(r*$\oplus$*)* and after this to send $a = r$ and *b=h(r*$\oplus$ α*)* to the tag. After this process is done, the tag will accept a and b, given the fact that *b=h(a* $\oplus$ α*)*= b = *h(r* $\oplus$ α*)*, so the authentication is broken. We can see the attack states chart in the Figure 4.

6.2. The weak point in the security proof process

The weak point in the LD protocol affect reader-to-tag authentication and leading to de desynchronization and traceability of tags is that the last message is compound and contains information whose integrity the tag cannot check but which is used to actualize the tags secret. In the demonstration from section 4.1, from [14] it is argued that in the LD protocol the correct third message *a,h(a* $\oplus$ c $\oplus$ k_i*)* can only be calculated with knowing of the tags serial number *c* and readers key k_i.

6.3. Presenting the solution

If we want to break reader-to-tag authentication in LD, the malicious persons uses the advantage of xors give up property, which is presented in the equation (1), and the fact that the tag cannot check the integrity of the last message. In section 5.3 we have already gave the solution against the use of xor. To authenticate a reader, the tag must send a challenge to the reader in the second message with a nonce value tag created. For this goal there are several standard mechanisms. If we take a look at the last message of the protocol, the reader must place the nonce together with all terms whose integrity needs to be protected inside of the digest function only an authorized person can create. In the end, the structure of the last message is then m,h(n,m,k), where m contains the terms on which integrity needs to be protected, n is the tag nonce and k represent the common secret.

7. Untraceability Tag Condition

A necessary condition for tag untraceability is that a malicious person, which has observed a particular tag once, must not be capable to face and recognize the tag as being the same tag in the future. To be more specific, we call a term, in which the malicious person can gain from one or more runs of a tag and which can identify to the malicious person, a unique attribute of the tag. There is a necessary condition for a tag to be untraceable so that a malicious person couldn‘t be capable to gain a unique attribute for the tag. The malicious person should be able to calculate a unique attribute, then we make a reference to the adversarys steps to arrive at such a term as the attribute acquisition attack. A very simple and concrete attribute can be found in the protocols where the tag receives a challenge and answers to that challenge c, challenge that has been sent by a reader and it‘s only a function f(c, k) which represents the challenge and a secret *k* which doesn‘t imply any nonce created by the tag. In this case, *c* is under the malicious person‘s control, *k* represents uniqueness to the tag, and the malicious person learns *f(c, k)* after one round of communication process with the tag. So, for the constant *c*, chosen by the malicious person, *f(c, k)* represents a unique attribute of the tag whose secret is *k*.

8. Conclusions

Together we have analyzed the simple necessary conditions for authentication and untraceability and we have studied the information misleading of the secret terms, discovering the two categories of attacks that have been published about RFID protocols. The attack methods presented in this study are very appropriate for RFID protocols. They take the big advantage of algebraic properties which give us enough weaknesses for operators and functions which typically are used in these protocols. The methods that are used in this study for finding algebraic replay and attribute acquisition attacks are without any kind of complications and also easily implementable, creating a tool that will be supported as a verification framework. The tool-supported verification of secrecy and authentication properties in presence of associative and commutative operators is already a very active research area. The verification process, automatically realized for untraceability, will be considered in future work following the procedure outlined in Section 3. An indication for how some of the cryptanalytic attacks may be automated can be obtained from the attack presented in Section 5 and 6. By representing all atomic terms as bit vectors, the system of equations of atomic terms can be expanded to a larger system over the finite field of two elements involving the bits of the vectors as variables. We have identified flaws in the protocol caused by the use of the xor operator and lack of message integrity verification. We have shown how these flaws can lead to attacks on authentication, untraceability, unlinkability, and synchronization of cryptographic key material. This article also described the consequences that these attacks can have for supply chain partners and we have given recommendations for improvements of the protocol. This article doesn‘t suggest, however, that only applying the proposed improvements will be enough to gain and to have a secure protocol. The design and verification process of such protocol does not make the goal of this paper but it will be taken into consideration for future work. In the end, working on this study we can draw a final conclusions that the attacks presented here underline three interesting areas with open problems affecting the security of RFID protocols.

References

[1] B. Alomair, L. Lazos, and R. Poovendran. Passive attacks on a class of authentication protocols for RFID. In ICISC, pages 102115, 2007

[2] G. Avoine. Adversary model for radio frequency identification. Technical Report LASEC-REPORT-2005-001, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland, September 2005.

[3] M. Backes and B. Pfitzmann. Limits of the cryptographic realization of DolevYao-style XOR. In ESORICS, pages 178196, 2005.

[4] H.-Y. Chien and C.-H. Chen. Mutual authentication protocol for RFID conforming to EPC class 1 generation 2 standards. Computer Standars and Interfaces, Elsevier Science Publishers, 29(2):254259, February 2007.

[5] H.-Y. Chien and C.-W. Huang. A lightweight RFID protocol using substring. In EUC, pages 422431, 2007.

[6] B. Defend, K. Fu, and A. Juels. Cryptanalysis of two lightweight RFID authentication schemes. In PerComWorkshops, pages 211216, 2007.

[7] T. v. Deursen, S. Mauw, and S. Radomirovic. Untraceability of RFID protocols. In Information Security Theory and Practices. Smart Devices,

Convergence and Next Generation Networks, volume 5019 of Lecture Notes in Computer Science, pages 115, Seville, Spain, 2008. Springer.

[8] R. Di Pietro and R. Molva. Information confinement, privacy, and security in RFID systems. In ESORICS, pages 187202, 2007.

[9] J. Ha, S.-J. Moon, J. M. G. Nieto, and C. Boyd. Low-cost and strong-security RFID authentication protocol. In EUC Workshops, pages 795807, 2007.

[10] Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J., Ribagorda, A.: Cryptanalysis of a novel authentication protocol conforming to EPC-C1G2 standard. (2007)

[11] A. Juels and S. A. Weis. Defining strong privacy for RFID. In PerCom Workshops, pages 342347, 2007.

[12] I. J. Kim, E. Y. Choi, and D. H. Lee. Secure mobile RFID system against privacy and security problems. In SecPerU 2007.

[13] T. Li and G. Wang. Security analysis of two ultralightweight RFID authentication protocols. In IFIP SEC 2007, Sandton, auteng, South Africa, May 2007. IFIP.

[14] Y. Li and X. Ding. Protecting RFID communications in supply chains. In ASIACCS, pages 234241, 2007.

[15] G. Lowe. A hierarchy of authentication specifications. In CSFW, pages 3144, 1997.

[16] D. Molnar, A. Soppera, and D. Wagner. A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In Selected Areas in Cryptography, pages 276290, 2005.

[17] K. Osaka, T. Takagi, K. Yamazaki, and O. Takahashi. An efficient and secure RFID security method with ownership transfer. In CIS, pages 778787, 2006.

[18] P. Peris-Lopez, J. C. Hernandez-Castro, J. Estevez-Tapiador, and A. Ribagorda. Cryptanalysis of a novel authentication protocol conforming to EPC-C1G2 standard., 2007.

[19] J. Saito, K. Imamoto, and K. Sakurai. Reassignment scheme of an RFID tags key for owner transfer. In EUC Workshops, pages 13031312, 2005.

[20] B. Song and C. J. Mitchell. RFID authentication protocol for low-cost tags. In WISEC, pages 140147, 2008.

[21] T. van Deursen and S. Radomirovic. RFID protocol attacks (version 0). Technical report, July 2008. http://satoss.uni.lu/projects/rfid/

[22] J. Yang, J. Park, H. Lee, K. Ren, and K. Kim. Mutual authentication

protocol for low-cost RFID. Handout of the Ecrypt Workshop on RFID and Lightweight Crypto, July 2005.

[23] Dolev, D., Yao, A.: On the security of public key protocols. IEEE Transactions on Information Theory IT-29(2) (March 1983) 198208

[24] Lowe, G.: A hierarchy of authentication specifications. In: CSFW. (1997) 3144

[25] Deursen, T.v., Mauw, S., Radomirovic, S.: Untraceability of RFID protocols. In: Information Security Theory and Practices. Smart Devices, Convergence and Next Generation Networks. Volume 5019 of Lecture Notes in Computer Science., Seville, Spain, Springer (2008) 115

[26] Avoine, G.: Adversary model for radio frequency identification. Technical Report LASEC-REPORT-2005-001, Swiss Federal Institute of Technology (EPFL), Security and Cryptography Laboratory (LASEC), Lausanne, Switzerland (September 2005)

[27] Juels, A., Weis, S.: Defining strong privacy for RFID. In: International Conference on Pervasive Computing and Communications PerCom 2007, New York, USA, IEEE, IEEE Computer Society Press (March 2007) 342347

[28] Chien, H.Y., Huang, C.W.: A lightweight RFID protocol using substring. In: Embedded and Ubiquitous Computing (EUC). (2007) 422431

[29] Damgard, I., Pedersen, M..: RFID security: Tradeoffs between security and efficiency. In: CT-RSA. (2008) 318332

[30] Paise, R.I., Vaudenay, S.: Mutual authentication in RFID: Security and privacy. In: ACM Symposium on Information, Computer and Communications Security (ASIACCS08), ACM Press (2008) 292299

[31] Deursen, T.v., Radomirovic, S.: Attacks on RFID protocols (version 1.0). Cryptology ePrint Archive,
Report 2008/310 (July 2008) http://eprint.iacr.org/2008/

[32] Lee, Y.K., Batina, L., Verbauwhede, I.: EC-RAC (ECDLP based randomized access control): Provably secure RFID authentication protocol. In: Proceedings of the 2008 IEEE International Conference on RFID. (2008) 97104

Mihailescu Marius Iulian
Department of Computer Science
University of Bucharest
Address: Str. Academiei nr.14, sector 1, C.P. 010014, Bucuresti, Romania
email:*mihmariusiulian@gmail.com*