Security of an RFID Protocol for Supply Chains

Ton van Deursen
University of Luxembourg
Department of Computer Science
and Communications
ton.vandeursen@uni.lu

Saša Radomirović
University of Luxembourg
Department of Computer Science
and Communications
sasa.radomirovic@uni.lu

Abstract

We analyze the security requirements on RFID protocols to be used in supply chains. We perform a case study on a recently proposed RFID authentication protocol specifically designed for supply chains. We discuss several shortcomings in this protocol related to mutual authentication, unlinkability, and desynchronization and suggest possible improvements. We show how the flaws in the protocol can be used to track products, relate incoming and outgoing products, and extort supply chain partners.

1. Introduction

Radio frequency identification (RFID) systems identify tags to readers in an open environment, requiring neither visual nor physical contact for communication. Due to the low production costs and very small size of the tags, RFID system deployment is steadily increasing, thereby replacing traditional identification methods such as bar codes. This is particularly true in supply chains where RFID tags permit a much more cost-effective and time-efficient tracing and management of a product than bar codes.

While RFID systems make management more efficient, they may also facilitate nefarious activities. Since neither physical nor visual contact is required to communicate with RFID tags, it becomes much easier for legitimate and malicious entities alike to interact with the tags. For instance, the ability to trace a product through a supply chain is vital for the product's manager, but detrimental if it can also be used by a competitor. Therefore, the security of tags, readers, and any other components of an RFID system is as important as the cost-effectiveness and time-efficiency. In this paper, we report on the security of a communication protocol between a tag and a reader proposed for use in supply chains [14]. In the following, we refer to this protocol as *LD* after the last names of its authors.

The intention to keep RFID tags small and cheap in order to facilitate large-scale deployment imposes significant limitations on the communication protocol in terms of the number and type of cryptographic primitives that may be used. Consequently, there is a wide variety of published protocols aiming at achieving strongly secure, untraceable, and efficient communication within today's resource limitations for RFID tags, a recent collection being [4, 8, 9, 12, 20]. The fact that actually all of the protocols in this collection have security flaws together with the significant number of publications containing attacks upon RFID protocols, most recently [1, 5, 6, 10, 13, 18, 7] shows that the design of resource-constrained, secure RFID protocols is still not well-understood.

As observed in [14], protocols for the supply chain setting differ from other RFID protocols in that they need to support transfer of ownership from one supply chain partner to another. Currently there are relatively few protocols that handle ownership transfer [16, 17, 19].

In this paper, we show that in the LD protocol the ownership transfer mechanism introduces security flaws which allow an attacker to trace tags through the supply chain, to break authentication, and to switch tags to and from a non-operational state. The salient part in the last attack is that only the attacker can switch a non-operational tag back on which in the supply chain scenario could be abused for extortion.

Our paper is structured as follows. In Section 2 we explain our terminology and adversary model and we discuss the security requirements for RFID protocols in a supply chain setting. In Section 3 we describe the LD protocol. In Section 4 we exhibit a flaw in the key update procedure that allows for attacks on unlinkability. In Section 5 we show an attack on authentication and discuss its consequences, and in Section 6 we conclude with an outlook on future work.

2. Preliminaries

We begin by explaining our terminology and adversary model and then proceed with the description of security re-

quirements for an RFID system to be used in supply chains.

In this paper, *reader* refers to the actual RFID reader as well as the database communicating with the reader, since this communication takes place over a secure channel. A *partner* refers to the owner and operator of one or more readers, all of which contain the same cryptographic key material. Readers of different partners may contain differing cryptographic key material. An *agent* can be a tag or a reader, while a *role* refers to the protocol steps a tag or reader is expected to carry out. A *run* is the execution of a role by an agent.

We use a standard Dolev-Yao intruder model in which the adversary controls the network. Since there might be attacks in this model which are not feasible in a real-world RFID system, we discuss, when necessary, the circumstances under which a presented attack becomes feasible. An RFID supply chain protocol should be considered a multi-party protocol, since every tag is supposed to successively interact with a predefined series of readers. Thus the intruder model should take malicious collaboration of partners into account. This is typically modeled by allowing the intruder to corrupt readers thus learning the reader's cryptographic keys. However, the attacks presented in the following sections can be carried out even under assumption that partners are trusted (and thus their readers incorruptible). We discuss malicious collaboration of partners in the concluding section.

We consider the following security properties to be relevant for an RFID protocol in the supply chain setting.

1. Mutual authentication through recent aliveness and agreement.

   - Tag-to-reader authentication allows the partner to verify that an incoming product is authentic. For this purpose the *recent aliveness* security property [15] suffices. Recent aliveness captures the fact that the tag needs to have generated a message as a consequence of a reader's query. More formally, a protocol guarantees to an agent a in role A that any corresponding agent b in role B has been recently alive, if and only if, whenever a completes a run, there has been an event of b during that run.
   - Reader-to-tag authentication allows the tag to verify that it is communicating with an authorized reader. This is important in the present scenario since a tag will need to be updated with new cryptographic key material which the reader sends to the tag. It is clear that the integrity of the key-update message needs to be preserved. Thus recent aliveness alone is not sufficient for this purpose. The tag needs a guarantee that the message containing key update information sent by the reader is equal to the message received by the tag. For this, the *agreement* [15] property is necessary. Formally, a role A satisfies the agreement property on a set of data items d if, whenever an agent a in role A completes a run of the protocol with an agent b in role B, then b (in role B) has been running the protocol with a in role A and the two agents agreed on the data values in d. Furthermore, each such run of a corresponds to a unique run of b.

2. Untraceability and unlinkability

   As observed in [14], it is important for a supply chain partner to prevent outsiders from being able to correlate incoming and outgoing products. Thus an adversary should not be able to trace products through the supply chain, or at the least, he should not be able to link incoming and outgoing products.

   A protocol satisfies the *untraceability* property for tags, if an adversary is not able to recognize a tag he previously observed or interacted with. Formal definitions of untraceability have been given in [2, 11, 7]. The weaker notion of *unlinkability* requires that an adversary cannot recognize a previously observed tag if between the two observations the tag has communicated with an authorized reader and updated its cryptographic key material.

The mutual authentication and unlinkability properties identified above correspond to the security requirements for the LD protocol stated in [14].

It is worth noting that the LD protocol due to its key-update procedure is a *stateful* protocol. Therefore the possibility of desynchronization attacks needs to be considered. In a *desynchronization attack* the adversary aims to disrupt the key update leaving the tag and reader in a desynchronized state and rendering future authentication impossible. It can be argued that the existence of desynchronization attacks implies the need for a desynchronization resistance property. Currently, there is no formal definition of desynchronization resistance for RFID protocols. In this paper, we have put extra conditions on the reader-to-tag authentication property by requiring agreement on sent messages to ensure some degree of desynchronization resistance. However, it is easy to see that in general, this extra requirement is insufficient. For instance, even with an agreement property the reader still has no guarantee that a tag has successfully updated its keys.

3. Protocol description

The LD protocol [14] was designed to be a mutual authentication protocol for re-writable RFID tags guaranteeing the unlinkability of tags in a supply chain. A supply

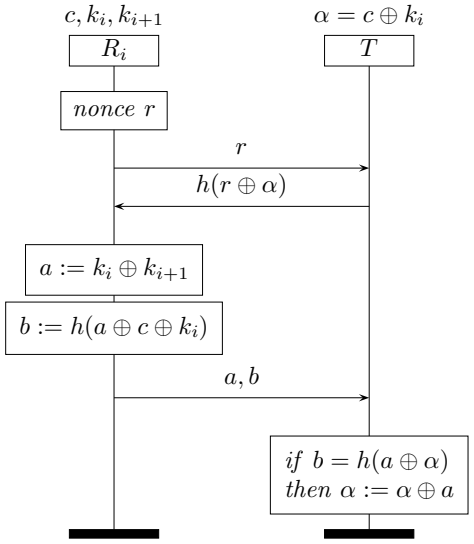

Figure 1. RFID protocol for supply chains

chain consists of a chain of partners, each of which is represented by a reader. Every reader R_i contains a secret k_i, as well as the secret of the next reader k_{i+1}. Additionally, every reader stores the identity c of every tag it may authenticate. Every tag T contains a pseudonym α representing its current temporary identity. The value of α is equal to $c \oplus k_i$ where k_i is the secret of some reader R_i currently allowed to identify and authenticate the tag.

The LD protocol is a challenge-response based protocol. The reader R_i challenges tag T with a nonce r. The tag calculates the *xor* of its current secret α and challenge r and responds with the hash of this value. The reader considers the tag authentic if it finds a secret c for which $h(c \oplus r \oplus k_i)$ is equal to the received response. The reader may then stop the protocol execution giving it the possibility to authenticate the tag again in a future communication session. Alternatively, the reader may 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 then verifies that $b = h(a \oplus \alpha)$ and updates its secret α by *xor*ing it with a. In doing this, ownership of tag T is transferred from reader R_i to reader R_{i+1}. The protocol is depicted as a message sequence chart in Figure 1.

The message sequence chart shows the role names, framed, near the top of the chart. Above the role names, the role's secret terms are shown. Actions, such as nonce generation, computation, verification of terms, and assignments are shown in boxes. Messages to be sent and expected to be received are specified above arrows connecting the roles. It is assumed that an agent continues the execution of its run only if it receives a message conforming to the specification.

4. Unlinkability

The LD protocol has not been designed to be untraceable, but merely unlinkable. Indeed, the fact that a tag does not introduce any randomness in its response to a reader's query implies that the tag is traceable between key updates. In the following we show, however, that the protocol does not provide unlinkability either by exhibiting an attack on this property. We then discuss the flaw in the security analysis of LD in [14].

4.1. The attack

To show that the protocol does not satisfy unlinkability, it suffices to exhibit a scenario in which the adversary recognizes a previously observed tag after the tag has updated its secret α.

By eavesdropping on a valid authentication session between a tag and a reader, the adversary learns $r, h(r \oplus \alpha), a$ and b. At the end of its execution, the tag updates its secret α by replacing it with $\alpha \oplus a$. The adversary can now challenge the tag with $r' = r \oplus a$, to which the tag will respond with $h(r' \oplus \alpha')$. By a simple algebraic property of *xor*, the response is equal to the previously observed one:

$$h(r' \oplus \alpha') = h(r \oplus a \oplus \alpha \oplus a) = h(r \oplus \alpha). \qquad (1)$$

In this context, we refer to the property of *xor* in equation (1) as the *cancellation property*, for obvious reasons. The attack is depicted in Figure 2.

Even if we assume that an adversary is not able to get close enough to a tag while it is being updated on a partner's premises, unlinkability can be plausibly broken. While a tag's sent messages can only be received within short ranges, the messages sent from reader to tag can be captured from a long distance. Thus we may assume that an adversary is able to eavesdrop on the reader's messages. We may further assume that the adversary can query incoming and outgoing products while they are outside the restricted area of a partner. Thus the following small extension of the above attack then becomes very plausible in the supply chain scenario. The attacker generates a nonce r and queries all incoming products with this nonce, observes the reader's key-update messages a, b, and queries all outgoing tags with $a \oplus r$. It now follows from equation (1) that eavesdropping on messages from reader to tag suffices to be able to match the incoming products' responses to the outgoing products' 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] contains the actual value with which the tag should update its key. This message can be observed and used by the adversary to break unlinkability

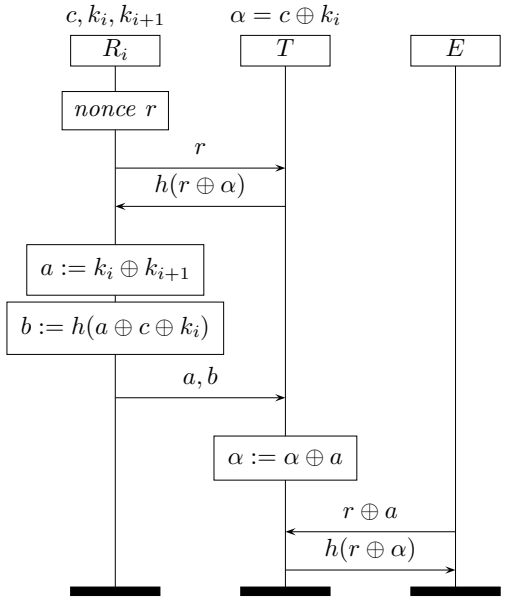

Figure 2. Attack on unlinkability

in a manner similar to the one described above. The attacks on these protocols are described in [21].

4.2. The flaw in the security proof

In the proof for the unlinkability claim [14, Statement 4.4] it is noted that the adversary may choose the same nonce $r = r'$ for the challenge before a tag is updated and after the tag is updated. It is then shown that in this case the adversary is not able to link the two responses $t = h(r \oplus c \oplus k)$ and $t' = h(r \oplus c' \oplus k')$ without knowledge of the keys. However, setting $r = r'$ is not the best strategy for the adversary. Since

$$\begin{aligned} & h(r \oplus c \oplus k) = h(r' \oplus c' \oplus k') \\ \Leftarrow \quad & r \oplus c \oplus k = r' \oplus c' \oplus k' \\ \Leftarrow \quad & r' = r \oplus k \oplus k' \wedge c = c', \end{aligned}$$

setting $r' = r \oplus k \oplus k'$ is a better choice for the adversary. Note that k and k' do not have to be keys of successive readers. By merely observing the key updates a chain of readers $R, \ldots, R'$ send to tags, the adversary can always compute $k \oplus \cdots \oplus k' = k \oplus k'$.

4.3. Recommendation

The flaw in the LD protocol affecting unlinkability is due to the algebraic property of the *xor* operator shown in equation (1).

This flaw can be avoided if concatenation of terms inside the hash functions is used instead of the *xor* operator, thus if the reader sends $h(a, (c \oplus k_i))$ instead of $b = h(a \oplus c \oplus k_i)$, where the comma denotes concatenation. While this makes the computation of the hash functions more expensive for the tag, it is a much safer alternative. Replacing the *xor* operator with any other operator with algebraic properties is dangerous in this setting, since the alternative operator's algebraic properties may be used to obtain a relation similar to equation (1).

Note that this improvement mitigates solely the *xor* cancellation attack on unlinkability. Further improvement are suggested at the end of the following section.

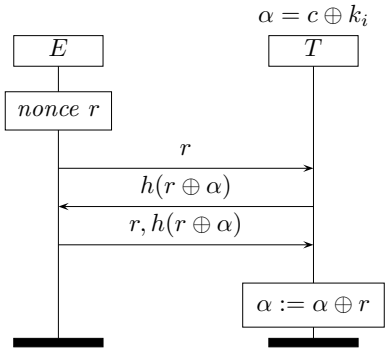

Figure 3. Attack on authentication

5. Authentication

As indicated in Section 2, the LD protocol needs to provide tag-to-reader authentication in order to ensure that only authentic products are accepted by a partner's reader and reader-to-tag authentication in order to guarantee that tags only update their keys after communicating with a legitimate reader. In this section, we take advantage of the *xor* operator to impersonate a reader and thus to break reader-to-tag authentication.

5.1. The attack

Recall that in the third protocol message the reader sends a, b to the tag, where $a = k_i \oplus k_{i+1}$ and $b = h(a \oplus \alpha)$. We observe that the tag cannot verify the value of a, since k_i and k_{i+1} are not known to the tag. The tag only verifies that the value of b is indeed equal to $h(a \oplus \alpha)$ and then updates its secret α by *xor*ing it with a.

Therefore, an adversary that is able to produce a valid combination of a and b can successfully impersonate a reader. To this end, the adversary challenges the tag with r to obtain $h(r \oplus \alpha)$ and then sends $a = r$ and $b = h(r \oplus \alpha)$ to the tag. The tag accepts a and b, because $b = h(a \oplus \alpha) = h(r \oplus \alpha)$, thus authentication is broken. The attack is depicted in Figure 3.

Note that in order to mount this attack, the adversary

does not even need to have observed a communication between a tag and legitimate reader.

5.2. Consequences

When the adversary impersonates a reader with the attack presented above, the tag updates its secret α to $\alpha \oplus r$. This results in *desynchronization* of the tag's state and the supply chain partner's database, since α is equal to $c \oplus r \oplus k_i$ instead of $c \oplus k_j$ for some j. The adversary can *re-synchronize* the tag with the database by repeating the attack with the same value for r so that the tag's secret becomes $\alpha \oplus r \oplus r = \alpha$, that is, equal to the original value.

Note that only the attacker may re-synchronize the tag with the database, since the nonce r was chosen by him. Thus, the attacker can *extort* supply chain partners by desynchronizing tags, virtually holding the products hostage, and re-synchronizing them when his demands are met.

Desynchronization attacks can also be used to trace tags. Since desynchronized tags will lead to problems at a supply chain partner's location the attacker is able to trace the tags through space and time.

5.3. The flaw in the security proof

The flaw in the LD protocol affecting reader-to-tag authentication and leading to desynchronization and traceability of tags is that the last message contains information whose integrity the tag cannot verify but which is used to update the tag's secret.

In the proof of statement 4.1 in [14], it is argued that in the LD protocol the correct third message $a, h(a \oplus c \oplus k_i)$ can only be computed with knowledge of the tag's serial number c and the reader's key k_i. Since the adversary does not know k_i, the conclusion is that he cannot produce a message which will be accepted by the tag. This reasoning does not take into account the test the tag performs in order to authenticate a reader. The tag does not verify whether the reader can compute $h(a \oplus c \oplus k_i)$, but merely whether the message a, b satisfies the equation $b = h(a \oplus \alpha)$. As shown in the attack, it is easy to produce such a message.

5.4. Recommendation

In order to break reader-to-tag authentication in LD, the adversary takes advantage of *xor*'s cancellation property, shown in equation (1), and the fact that the tag cannot verify the integrity of the last message.

We have already advised against the use of *xor* in Section 4.3. Furthermore, to authenticate a reader, the tag must challenge the reader in the second message with a nonce the tag created. There are several standard mechanisms for this purpose. In the last message of the protocol, the reader must place this nonce together with all terms whose integrity needs to be protected inside a hash function only an authorized reader can create. The format of the last message is then $m, h(n, m, k)$, where m contains the terms whose integrity needs to be protected, n is the tag's nonce, and k is a common secret. Such a construction proves to the tag that a legitimate reader was recently alive and that it recently created the integrity-protected parts in the last message. This is, in principle, achievable with hash functions.

6. Conclusion

We have analyzed the security of an RFID protocol aimed at supply chain structures. 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. We have described the consequences these attacks can have for supply chain partners and we have given recommendations for improvements of the protocol. We do not suggest, however, that merely applying the proposed improvements will suffice to obtain a secure protocol. The design and verification of such a protocol are beyond the scope of this paper and will be considered in future work.

More generally, we believe that the attacks exhibited in this paper highlight three interesting areas with open problems affecting the security of RFID protocols.

1. Operators with algebraic properties

   The resource constraints imposed on RFID tags have enticed protocol designers to take advantage of operators with algebraic properties due to their simplicity and efficiency. As the attacks in this paper indicate, the introduction of even the simplest operator, *xor*, into a security protocol can lead to serious security problems.

   It is therefore important to understand how to use operators with algebraic properties safely and how to verify the security of protocols employing such operators. Research into the latter question has been ongoing for several years under the question of how to incorporate new cryptographic primitives into existing formal models for verification of security protocols. It has been shown [3] that the inclusion of the *xor* operator into a Dolev–Yao-style formal model is not straightforward due to soundness problems with respect to its cryptographic realization.

2. Definition and verification of new security properties

   Formal definitions of unlinkability and untraceability have only relatively recently been formulated and there

is still much room for improvement in the verification of these properties.

Furthermore, the introduction of stateful RFID protocols and the immediate proliferation of desynchronization attacks, such as the one shown in Section 5.2, indicate a strong need for a formal definition and methods for the verification of desynchronization resistance.

3. Adversary model

   The adversary model for supply chains presented in Section 2 and also proposed in [14] requires that the RFID protocol has to be secure even under the assumption that partners collude in order to attack each other. As noted in [14] the protocol considered in this paper does not meet this requirement for the following reasons. Since the knowledge of one key k_i allows the successive computation of all keys in the chain, it is possible for one partner to attack all other partners. For a merely curious partner, knowing the keys of other partners allows for the linking of incoming and outgoing products at those partner's premises. More malicious partners can even change product codes, introduce new codes, and place the blame for faulty products on any other partner by using the other partner's key.

   The possibility of such insider attacks poses a severe security risk, since it implies that the negligence or malicious intent of a single partner leads to security problems in the entire supply chain. As a consequence, a potentially challenging open problem is the design and verification of a practical light-weight RFID protocol for supply chains secure under malicious collusion of partners.

References

[1] B. Alomair, L. Lazos, and R. Poovendran. Passive attacks on a class of authentication protocols for RFID. In *ICISC*, pages 102–115, 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 Dolev–Yao-style XOR. In *ESORICS*, pages 178–196, 2005.

[4] H.-Y. Chien and C.-H. Chen. Mutual authentication protocol for RFID conforming to EPC class 1 generation 2 standards. *Computer Standars & Interfaces, Elsevier Science Publishers*, 29(2):254–259, February 2007.

[5] H.-Y. Chien and C.-W. Huang. A lightweight RFID protocol using substring. In *EUC*, pages 422–431, 2007.

[6] B. Defend, K. Fu, and A. Juels. Cryptanalysis of two lightweight RFID authentication schemes. In *PerCom Workshops*, pages 211–216, 2007.

[7] T. v. Deursen, S. Mauw, and S. Radomirović. 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 1–15, Seville, Spain, 2008. Springer.

[8] R. Di Pietro and R. Molva. Information confinement, privacy, and security in RFID systems. In *ESORICS*, pages 187–202, 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 795–807, 2007.

[10] J. Ha, S.-J. Moon, J. M. G. Nieto, and C. Boyd. Security analysis and enhancement of one-way hash based low-cost authentication protocol (OHLCAP). In *PAKDD Workshops*, pages 574–583, 2007.

[11] A. Juels and S. A. Weis. Defining strong privacy for RFID. In *PerCom Workshops*, pages 342–347, 2007.

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

[13] T. Li and G. Wang. Security analysis of two ultra-lightweight RFID authentication protocols. In *IFIP SEC 2007*, Sandton, Gauteng, South Africa, May 2007. IFIP.

[14] Y. Li and X. Ding. Protecting RFID communications in supply chains. In *ASIACCS*, pages 234–241, 2007.

[15] G. Lowe. A hierarchy of authentication specifications. In *CSFW*, pages 31–44, 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 276–290, 2005.

[17] K. Osaka, T. Takagi, K. Yamazaki, and O. Takahashi. An efficient and secure RFID security method with ownership transfer. In *CIS*, pages 778–787, 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 tag's key for owner transfer. In *EUC Workshops*, pages 1303–1312, 2005.

[20] B. Song and C. J. Mitchell. RFID authentication protocol for low-cost tags. In *WISEC*, pages 140–147, 2008.

[21] T. van Deursen and S. Radomirović. 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.