Jaap-Henk Hoepman
Ingrid Verbauwhede (Eds.)

Radio Frequency Identification

Security and Privacy Issues

LNCS 7739

8th International Workshop, RFIDSec 2012
Nijmegen, The Netherlands, July 2012
Revised Selected Papers

Springer

Lecture Notes in Computer Science 7739

Commenced Publication in 1973
Founding and Former Series Editors:
Gerhard Goos, Juris Hartmanis, and Jan van Leeuwen

Jaap-Henk Hoepman Ingrid Verbauwhede (Eds.)

Radio Frequency Identification

Security and Privacy Issues

8th International Workshop, RFIDSec 2012
Nijmegen, The Netherlands, July 2-3, 2012
Revised Selected Papers

Springer

Volume Editors

Jaap-Henk Hoepman
Radboud University Nijmegen
Institute for Computing and Information Sciences
Department of Digital Security
Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands
E-mail: jhh@cs.ru.nl

Ingrid Verbauwhede
K.U. Leuven, ESAT - SCD/COSIC
Electrical Engineering Department
Kasteelpark 10, 3001 Heverlee, Belgium
E-mail: ingrid.verbauwhede@esat.kuleuven.be

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-36139-5 e-ISBN 978-3-642-36140-1
DOI 10.1007/978-3-642-36140-1
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012955559

CR Subject Classification (1998): K.6.5, E.3, J.1, K.4.4, C.2.4, C.3, F.2.2

LNCS Sublibrary: SL 4 – Security and Cryptology

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

RFIDsec was the first international workshop to focus on security and privacy in Radio Frequency Identification (RFID). Starting in 2005, RFIDsec is today the reference workshop in the RFID field with participants from all over the world. The 8th workshop took place in Nijmegen, The Netherlands, July 1–3, 2012.

The workshop received 29 submissions. Each submission was reviewed by at least 2, and on average 3, program committee members. The committee decided to accept 12 papers, of which you can find an updated version in these proceedings.

The program also included 3 invited talks by experts in the field. The first invited talk was given by Florian Michahelles on the topic of "When Will RFID Embrace Our Everyday Lives?" Joan Daeman presented the second invited presentation on "Permutation-Based Symmetric Cryptography." The third presentation was given by Konstantinos Markantonakis on the "Interplay of Business Objectives and Academic Research Holders of NFC Mobile Service Destiny."

New this year was the organization of 4 tutorials on July 1, 2012. These tutorials covered the following topics: "Proxmark, the Swiss Army Knife for RFID Security Research," given by F. Garcia, G. de Koning Gans, and R. Verdult; an "RFID hands-on," given by Philippe Teuwen; "Some Physical Aspects of RFID Security," by Boris Škorić; and "Low-Power Hardware Design for Lightweight Cryptography," given by Ingrid Verbauwhede.

The workshop organizers gratefully acknowledge the sponsorship from NWO (Nederlandse Organizatie voor Wetenschappelijk Onderzoek), and from Riscure, Nedap, NXP, and Oridao. Their support allowed us to provide stipends to students attending the conference. We would also like to thank Lejla Batina, the general chair of RFIDSec 2012, for making sure that every aspect of the conference ran smoothly. The local organizing committee saw to it that all attendees felt welcome. Our thanks go to Gergely Alpár Fabian van den Broek, Flavio Garcia, Irma Haerkens, Gerhard de Koning Gans, Joeri de Ruiter, and Roel Verdult. Gergely Alpár maintained the website. Part of conference logistics is a software system to manage the submissions and the proceedings. The ease of use of Easychair was much appreciated. Baris Ege and Amitabh Das made all of the necessary preparations for these proceedings.

Finally, we would like to thank the presenters, the attendees, and now the readers of these proceedings for their interest and their contribution to the research knowledge in this field.

November 2012

Ingrid Verbauwhede
Jaap-Henk Hoepman

Organization

Program Committee

Gildas Avoine	UCL, Louvain-la-Neuve
Lejla Batina	Katholieke Universiteit Leuven
Mike Burmester	Florida State University
Srdjan Capkun	ETH Zurich
Flavio Garcia	Radboud University Nijmegen
Jorge Guajardo	Bosch Research and Technology Center, Greater Pittsburgh Area
Jaap-Henk Hoepman	Radboud University Nijmegen
Michael Hutter	University of Technology Graz
Farinaz Koushanfar	Rice University Houston
Gregor Leander	DTU, Lyngby
Kerstin Lemke-Rust	Hochschule Bonn-Rhein-Sieg, Sankt Augustin
Kostas Markantonakis	ISG-Smart Card Centre, Royal Holloway University of London
Florian Michahelles	ETH Zurich
Karsten Nohl	University of Virginia
Berna Ors	Istanbul Technical University
Christof Paar	Ruhr University Bochum
Svetla Petkova-Nikova	Katholieke Universiteit Leuven and University of Twente
Axel Poschmann	Nanyang Technological University
Matt Robshaw	Orange Labs, Caen
Ahmad Sadeghi	TU Darmstadt
Kazuo Sakiyama	The University of Electro-Communications, Chofu Tokyo
Nitesh Saxena	Polytechnic University, New York and University of Alabama at Birmingham
Dave Singelee	Katholieke Universiteit Leuven
Francois-Xavier Standaert	UCL, Louvain-la-Neuve
Ingrid Verbauwhede	Katholieke Universiteit Leuven
Marc Witteman	Riscure, Delft
Avishai Wool	Tel Aviv University

Additional Reviewers

Balasch, Josep
Carpent, Xavier
de Koning Gans, Gerhard
Erguler, Imran
Halevi, Tzipora
Hinterwaelder, Gesine
Kiltz, Eike
Knospe, Heiko
Koeune, François
Lyubashevsky, Vadim
Martin, Tania
Meurer, Alex
Nikov, Ventzislav
Ozkaya, Ozen
Peris-Lopez, Pedro
Plos, Thomas
Poschmann, Axel
Rozic, Vladimir
Seys, Stefaan
Verdult, Roel
Voris, Jonathan
Vullers, Pim
Wachsmann, Christian
Watanabe, Dai
Wenger, Erich
Zanetti, Davide

Table of Contents

On the Security of Tan *et al.* Serverless RFID Authentication and Search Protocols

Masoumeh Safkhani[1], Pedro Peris-Lopez[2], Nasour Bagheri[3], Majid Naderi[1], and Julio Cesar Hernandez-Castro[4]

[1] Department of Electrical Engineering, Iran University of Science and Technology (IUST), Tehran, Iran
[2] Computer Security Lab (COSEC), Carlos III University of Madrid, Spain
[3] Department of Electrical Engineering, Shahid Rajaee Teachers Training University, Tehran, Iran
[4] School of Computing, University of Portsmouth, UK

Abstract. In this paper, we analyze the security of the mutual authentication and search protocols recently proposed by Tan *et al.* [20]. Our security analysis clearly highlights important security pitfalls in these. More precisely, privacy location of the tags' holder is compromised by the authentication protocol. Moreover, the static identifier which represents the most valuable information that a tag supposedly transmits in a secure way, can be exposed by an adversary when the authentication protocol is used in combination with one of the search protocols. Finally, we point out how the improved search protocols are vulnerable to traceability attacks, and show the way an attacker can impersonate a legitimate tag.

Keywords: RFID, Mutual Authentication, Search Protocol, Cryptanalysis.

1 Introduction

In RFID systems, readers and tags can employ authentication protocols with the purpose of authenticating each other. These protocols commonly exchange a number of messages between the involved entities. Specifically, one of the protocol entities sends a challenge(s) to the other entity and then it verifies the correctness of the received response(s). To achieve the intended security objectives, readers and tags are often mutually authenticated. Besides authentication, when a reader needs to find a certain tag among a large population of tags, it requires a search protocol. An efficient scheme is difficult to design, and more so if it should be robust enough not to compromise the security and privacy of the system.

Tan *et al.* recently proposed a serverless mutual authentication and a search protocol for RFID systems [20], both heavily based on the use of hash functions. Moreover, three improved search protocols were introduced in their paper. The authors claimed optimal security for both the proposed authentication protocol

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 1–19, 2013.

and the basic/improved search protocols. Nevertheless, in this article, we show important security pitfalls on these schemes.

Paper Organization: We describe Tan *et al.* protocols in § 2. A traceability attack against the mutual authentication protocol is presented in § 3. In § 4, we analyze the privacy protection of confidential information regarding the mutual authentication protocol when it is used in combination with one of the proposed search protocols. Then, traceability and impersonation attacks against the improved search protocols are introduced in § 5 and § 6, respectively. Finally, in § 7 we extract some conclusions.

2 Protocols Description

The notation used through the paper can be consulted in Appendix A. A complete description of Tan *et al.* mutual authentication protocol, search protocol and improved search protocols is provided from Fig. 1 to Fig. 5. in Appendix B. We now give a brief description of these schemes, but we urge the reader to consult the original paper for further details [20].

Tan *et al.* mutual authentication protocol is divided into two phases, the setup and the mutual authentication phase. In the setup phase, the reader R_i authenticates itself to CA and obtains access to tags $T_1, \cdots, T_n$. R_i will receive L_i where:

$$L_i = \{(f(r_i, t_1), id_1), \cdots, (f(r_i, t_n), id_n)\}$$

In the mutual authentication phase, the reader sends a *request* to the tag. The tag replies with a random value n_j. Then, the reader sends its identifier r_i and a random value n_i to the tag. The tag answers a tuple $\{h(f(r_i, t_j))_m, h(f(r_i, t_j \| n_i \| n_j) \oplus id_j\}$, where n_j represents a random value generated by the tag and $h(f(r_i, t_j))_m$ symbolizes the first m bits of $h(f(r_i, t_j))$. Upon receiving these values, the reader hashes every entry in L_i and checks whether the first m bits match with the received value $h(f(r_i, t_j))_m$. After that, it computes $h(f(r_i, t_j) \| n_i \| n_j)$ and obtains id_j by a simple XOR operation. If the obtained id_j matches with the id_j stored in L_i, the reader authenticates the tag.

Tan *et al.* also proposed several search protocols. In the basic protocol, sketched in Fig.2, the reader broadcasts a triplet containing $\{h(f(r_i, t_j) \| n_r) \oplus id_j, n_r, r_i\}$. Each tag T_j computes $h(f(r_i, t_j) \| n_r)$ and XORs it with the received value $h(f(r_i, t_j) \| n_r) \oplus id_j$ to extract id. If $id = id_j$, the matched tag authenticates the reader and replies a tuple consisting of $\{h(f(r_i, t_j) \| n_t \| n_r) \oplus id_j, n_t\}$. As the authors state in the article, this protocol is vulnerable to traceability attacks (see Section 5 for more details). Motivated by this weakness, the authors proposed three improved search protocols.

In the first of these, depicted in Fig. 3, each RFID tag keeps a record of the recent random values used by the reader. Hence, a RFID tag rejects any query for which the random value n_r exists in its list. Specifically, the authors propose that each tag keeps only the last seen random value – denoted as *oldn*.

The second improved solution consists on a challenge and response scheme (see Fig. 4). The reader broadcasts m bits of id_j ($\{id_j\}_m$), its identifier r_i and a challenge r_i. Any tag on reader range that matches the first m bits of the id will reply to the query. So, depending on m, there could be multiple tags that share the same m bits on id and answer the query. This approach is used to avoid the condition where replying to a query can be used to identify a tag. Nevertheless, this protocol does not work well when the tags' id is structured, e.g. the first several bits of the tags' id represent the product code, the next several bits provide the tag origin, manufacturer and so on.

Finally, Tan *et al.* proposed a third protocol, sketched in Fig. 5, in which the use of noise attempts to mask the origin of the replies. Based on this approach, each tag that receives a search query and does not match the request will reply with some probability – λ. The authors claimed that an adversary cannot track a tag since any tag could potentially reply.

3 Traceability Attack on the Mutual Authentication Protocol

We can find in the literature a significant number of RFID authentication protocols based on hash-functions. Nevertheless, two main drawbacks dissuade authors from its practical use. Firstly, the support on-chip of hash functions may be questioned due to their demand for circuit size, memory and power consumption [7]. Secondly, the search process in the readers – matching between the values received from the tag and the records stored in the reader – often implies heavy computation load.

Tan *et al.* [20] proposed that the tags transmit $h(f(r_i, t_j))_m$ as a mechanism to improve the search time for the reader. In this Section, we evaluate how privacy location can be compromised by using this confidential information. More precisely, we use traceability model proposed by Phan [3], which is a reformulation of the model initially proposed by Juels and Weis [12] (the reader is urged to consult Appendix C for details).

3.1 Proposed Attack

We show how Tan *et al.* protocol puts at stake the privacy location of tags' holders because tags can be tracked with a high probability. Specifically, an adversary A has to follow the steps described below:

- Phase 1 (Learning): A sends an Execute(R_i, T_0, k) query and acquires the public messages passed over the insecure radio channel $\{n_0, n_i, r_i, h(f(r_i, t_0))_m, h(f(r_i, t_0)\|n_i\|n_0) \oplus id_0\}$. Then, $\mathcal{A}$ stores $X = h(f(r_i, t_0))_m$ as an static search index linked to T_0.
- Phase 2 (Challenge): $\mathcal{A}$ chooses two fresh tags $\{T_0, T_1\}$ whose associated identifiers and keys are $\{id_0, t_0\}$ and $\{id_1, t_1\}$, respectively. He then sends a Test(k', T_0, T_1) query. As a result, and depending on a chosen random bit

$b \in \{0,1\}$, A is given an static search index $Y = h(f(r_i, t_b))_m$ from the set $\{h(f(r_i, t_0))_m, h(f(r_i, t_1))_m\}$.

– Phase 3 (Guessing): $\mathcal{A}$ finishes $\mathcal{G}$ and outputs a bit $\widetilde{b}$ as its conjecture of the value b. In particular, $\mathcal{A}$ utilizes the following simple decision rule:

$$\begin{cases} \text{if } X == Y & \widetilde{b} = 0 \\ \text{if } \quad X \neq Y & \widetilde{b} = 1 \end{cases} \tag{1}$$

We emphasize here that only static values are used for the computation of the search indexes X and Y. That is, these ones are independent of the random numbers associated to each session and are unequivocally linked to a particular tag.

In [20], the authors define β as the probability that, given a tag, the probability that when a reader reads in another tag having the same first m bits, the two tags are the same. So, the advantage of an adversary after eavesdropping one authentication session is described below:

$$Adv_A^{\mathsf{UNT}}(q,1) = |(\frac{1}{2} \cdot 1 + \frac{1}{2} \cdot \beta) - \frac{1}{2}| = \frac{\beta}{2}$$

If $\beta = 1$ the advantage of the adversary is maximal and the smaller the β, the more privacy protection is offered. So, the benefits in the efficiency of the protocol compromise the privacy location. The authors suggested that the way of combating this problem is that tags only answer $h(f(r_i, t_j)\|n_j\|n_j) \oplus id_j$. Nevertheless, their proposal then just turns into another proposal mixed in the huge clutter of authentication protocols based on hash functions and the benefits claimed of being a novel and efficient are ruined. In other words, efficiency in the protocol is only obtained when privacy location is at risk.

3.2 A Note on the Second Improved Search Protocol

In this protocol (see Fig. 4), the authors follow a challenge and response scheme. The reader broadcasts the first m bits of id_j – denoted as $[id_j]_m$ – and the matched tag replies $\{h(f(r_i, t_j)\|n_t) \oplus id_j,\ n_t\}$. The adversary (A) can follow the same strategy described in the previous section and track tags. We roughly describe the process to avoid repetition. In the Leaning Phase, A acquires the first m bits of the static identifier $(X = [id_j]_m)$ linked to T_0. Then, the challenge phase is executed; depending on a chosen random bit $b \in \{0,1\}$, A captures the first m bits of a static identifier $Y = [id_b]_m \in \{[id_0]_m, [id_1]_m\}$. Finally, in the Guessing Phase, A hypothesizes on b value by using this following decision rule:

$$\begin{cases} \text{if } X == Y & \widetilde{b} = 0 \\ \text{if } \quad X \neq Y & \widetilde{b} = 1 \end{cases} \tag{2}$$

In the original protocol, Tan *et al.* assume that several tags can share the same $[id]_m$. For that reason, they emphasize that the proposed solution does not work

well for tags with an structured id. Under this condition, we consider the case where each tag is assigned a random id, which complies with the authors' requirements. If we asumme that we have a population of N tags and the reader sends $[id]_m$, $\frac{N}{2^m}$ tags are expected to share the same $[id]_m$. Nevertheless, given T_0 with $[id_0]_m$, any randomly selected tag T_1 satisfies $[id_0]_m = [id_1]_m$ with the probability of 2^{-m}. Hence, we can define $\gamma = \frac{1}{2^m}$ as the probability that, given a tag, the probability when the reader reads another tag having the same first m bits, the two tags are different (i.e. $\gamma = \beta - 1$ according definition of the above section). The adversary fails in her traceability attempt when T_1 is randomly chosen at the Challenge-Phase and it shares the same m-bits of the static identifier with T_0 (i.e. $[id_0]_m = [id_1]_m$). So, after conducting the above attack, the adversary advantage is:

$$Adv_A^{\mathsf{UNT}}(q,1) = |(\frac{1}{2} + \frac{1}{2} \cdot (1-\gamma)) - \frac{1}{2}| = \frac{1}{2} \cdot (1-\gamma) \leq \frac{1}{2}$$

To determine the lower bound of the adversary's advantage in tracking a tag, we consider the other side of the problem where the $[id]_m$ values are not random. On the other hand, according to the authors' assumption, all tags do not share the same $[id]_m$. Hence, the adversary can select its target tag T_0 such that given $[id_0]_m$ and N as the number of tags in its range, at least $\frac{N}{2}$ of the tags have different $[id]_m$ compared to $[id_0]_m$. Therefore, dividing the tags into two separated groups, based on their $[id]_m$, the adversary can select its target tag from the group in minority. Hence, we can define $\gamma \leq \frac{1}{2}$ as the probability that, given a selected tag, the probability when the reader reads another tag having the same first m bits, the two tags are different (i.e. $\gamma = \beta - 1$). The adversary fails in her traceability attempt when T_1 is randomly chosen at the Challenge-Phase and it shares the same m-bits of the static identifier with T_0 (i.e. $[id_0]_m = [id_1]_m$). So, after conducting the above attack, the adversary advantage for the group of the tags that are in the minority considering its $[id]_m$ is:

$$Adv_A^{\mathsf{UNT}}(q,1) = |(\frac{1}{2} + \frac{1}{2} \cdot (1-\gamma)) - \frac{1}{2}| = \frac{1}{2} \cdot (1-\gamma) \geq \frac{1}{4}$$

Summarising both cases, if $m > 0$ – and $0 < \gamma < 1$ consequently – the advantage of the adversary is significant and the privacy location compromised. More precisely, the adversary's advantage is bounded as below:

$$\frac{1}{4} \leq Adv_A^{\mathsf{UNT}}(q,1) \leq \frac{1}{2}$$

4 *id* Disclosure Attack

In this section we analyze the security of the mutual authentication protocol proposed by Tan *et al.* [20] when an RFID system uses this mutual authentication protocol combined with one of the search protocols presented in Section 2 –

except the second improved search protocol sketched in Fig 4. We show how an attacker can disclose the most valuable information stored on tags memory, the unique identifier id. To success in our attack, we make the following assumption about the hash function used in the authentication and search protocols. We assume the usage of an iterated hash function based on Merkle-Damgård (MD) [4,15] with a compression function belongs to a group of PGV compression functions [18] for which the message block M_i is used as the key in the underlying block cipher E.

It must be noted that our assumption on the hash function does not compromise the assumption made in the original paper [20]. More precisely, the authors consider $h(x)$ as a one-way hash function, which is not contradiction of what we assume in our analysis. More over, the construction assumed, that is MD, is very well-known and used in popular hash functions such as MD5 or SHA family. On the other hand, it is a common approach to analyze the security of a provable secure hash-based scheme with an ideal hash function by instantiating it with a real hash function. For instance, in [9] Gauravaram and Knudsen shown an approach to use the Dean's method of finding expandable messages [6,13] for finding a second preimage in the Merkle-Damgård hash function to forge a signature scheme based on a RMX-hash function [10] which uses the Davies-Meyer compression functions.

In Appendix D the hash function model is introduced in order to facilitate the understanding of our proposed attack.

4.1 Proposed Attack

In this section we consider the mutual authentication protocol and the basic search protocol (see Fig. 1 and Fig. 2, respectively). To disclose the id, when the hash function uses MD construction with Davies Mayer (DM) [5] one-way compression function, the adversary (A) does as follows:

1. A eavesdrops on a search session between R_i and T_j and records the query $(X = \{h(f(r_i, t_j)\|n_r) \oplus id_j,\ n_r,\ r_i\})$ sent by R_i.
2. A supplants R_i in a mutual authentication session:
   (a) A starts the protocol by sending a *request* to T_j.
   (b) T_j replies with a random value n_j.
   (c) A sends {X[2], X[3]}, where $X[i]$ symbolizes the i-th value of X.
   (d) A captures the response of the tag $(Y = \{h(f(r_i, t_0))_m, h(f(r_i, t_j)\|n_r\|n_j) \oplus id_j\})$.
3. A knows $X[1] = h(f(r_i, t_j)\|n_r) \oplus id_j$ and $Y[2] = h(f(r_i, t_j)\|n_r\|n_j) \oplus id_j$ and the random numbers $\{n_r, n_j\}$.
4. Taking into account the MD structure and DM construction – our initial assumption –, A uses $X[1]$ and $Y[2]$ to disclose id_j as follows:

a) A computes the XOR between the two values captured:

$$\begin{aligned} & X[1] \oplus Y[2] \\ &= h(f(r_i, t_j)\|n_r) \oplus h(f(r_i, t_j)\|n_r\|n_j) \\ &= h(f(r_i, t_j)\|n_r) \oplus E_{n_j}(h(f(r_i, t_j)\|n_r)) \\ &\quad \oplus h(f(r_i, t_j)\|n_r) \\ &= E_{n_j}(h(f(r_i, t_j)\|n_r)) \end{aligned}$$

b) A obtains $h(f(r_i, t_j)\|n_r)$ value:

$$h(f(r_i, t_j)\|n_r) = E_{n_j}^{-1}(X[1] \oplus Y[2])$$

c) Finally, A discloses the static identifier id_j:

$$\begin{aligned} id_j &= h(f(r_i, t_j)\|n_r) \oplus X[1] \\ &= E_{n_j}^{-1}(X[1] \oplus Y[2]) \oplus X[1] \end{aligned}$$

So, the adversary can disclose the static identifier of a tag after observing an execution of the search protocol and impersonating a reader in a session of the mutual authentication scheme. Then, just by computing XOR operations and a decryption in which the key is known, the static identifier is revealed, compromising the privacy information.

Remark 1. A similar attack can be executed to disclose the id_j of T_j when the mutual authentication is used in conjunction with the first and the third improved search protocols, sketched on Fig. 3 and Fig. 5 respectively.

Remark 2. A similar attack can be run to disclose the id_j of T_j when the underlying hash function is used with other one-way compression functions that according [2] are indexed by 6, 7 and 8 in Table 2 in Appendix D.

Remark 3. The above attack may not work if the hash function is used along with *MD strengthening* [14]. However, the usage of MD strengthening in this application could be an unrealistic assumption because efficiency is a vital requirement and we only work with messages of length at most three blocks, assuming that $f(r_i, t_j)$ has the same length as n_j, n_r and the message block length of compression function. We note that, for the given protocols, any call to the hash function are of the form $(h(f(r_i, t_j))$, $(h(f(r_i, t_j)\|n_r)$, $(h(f(r_i, t_j)\|n_r\|n_j)$ or $(h(f(r_i, t_j)\|n_j\|n_r)$.

5 Traceability Attack on Search Protocols

Besides RFID authentication protocols, other schemes for performing different RFID operations are used. Searching protocols are one of these mentioned operations and facilitate the seeking of an specific tag from a large population of tags. To provide privacy and security, the scheme has to fulfil two requirements:

1) RFID tags have to authenticate readers before answering; 2) RFID readers have to make sure than only genuine tags receive and understand its query. In other words, tags only have to answer to authenticated readers and readers only have to query authenticated tags.

In Fig. 2, the basic search protocol is sketched. As the authors state in the original article [20], the scheme does not offer protection against traceability attacks. The traceability attack made reference here is quite different from the one introduced in Section 3. Basically, the ultimate aim that an attacker pursues here is the detection of the presence or absence of an specific tag. We now describe how an attacker successes against the basic search protocol. First, the adversary (A) eavesdrops on an execution of the protocol between a reader and a group of tags – A only has to detect and capture the values from the query and the answer. Then, A replies the captured query, the target tag answers – query is legitimate – and finally A captures the answer. Although the obtained value is different from the previous one – result of using the nonce n_t – A can detect the presence/absence of the target tag because only the pursued tag knows the secret information $\{t_j, id_j\}$ necessary to check the legitimacy of the query and generate a valid answer. The attack can be extended by the information obtained by physical observation. For that purpose, A isolates each tag in the group, replies the captured query and waits for the answer.

In [20], several improved search protocols are proposed to minimize the impact of this sort of tracking. Nevertheless, the authors fail in their attempt because the new versions are insecure as the basic protocol. More precisely, in this section we present traceability attacks on the first and the third improved search protocols (see Fig. 3 and Fig. 5 for details) and concerning the second improved protocol an attack was already presented in Section 3.2.

5.1 Traceability Attack on the First Improved Search Protocol

To avoid the reply of previous captured messages and facilitate the success of traceability attacks, the reader has to be forced to use a different random number n_r for each new query. RFID tags can keep a record of the recent challenges used to accomplish this task. More precisely, the authors considered enough that tags only store the last random number used – denoted as *oldn* in Fig. 3. The authors claimed that an adversary can not track a tag because the adversary needs two successful queries. Nevertheless, this claim is incorrect and an adversary (A) can exploit the symmetric of the protocol (query/reply). To mount a traceability attack, A does as described below:

1. A eavesdrops a transaction between R_i and T_i and captures the query $\{X = h(f(r_i, t_j)\|n_r) \oplus id_j,\ n_r,\ r_i\}$ and the answer $\{Y = h(f(r_i, t_j)\|n_t) \oplus id_j,\ n_t\}$ transmitted over the insecure radio channel.
2. A generates a legitimate query from the captured information: $\{Y, X[3]\} = \{h(f(r_i, t_j)\|n_t) \oplus id_j,\ n_t,\ r_i\}$, where $X[i]$ represents the i-th element of X. Specifically,
   (a) From the captured query, the identifier of the reader is copied ($r_i = X[3]$).

   (b) From the captured reply, a fresh authentication token ($Y = \{h(f(r_i, t_j) \| n_t) \oplus id_j,\ n_t\}$) using a new random number n'_r is obtained (i.e. $n'_r = n_t$).
3. To trace T_i, A replies $\{Y, X[3]\} = \{h(f(r_i, t_j) \| n'_r) \oplus id_j,\ n'_r,\ r_i\}$, where $n'_r = n_t$.
4. If any tag T^* answers to the adversary, A detects the presence of T_j again. As the query/answer is based on the knowledge of the private information $\{t_j, id_j\}$ linked unequivocally with one tag, only the target tag could check the query and send the answer.

Alternatively, an adversary (A) can exploit the fact that only one random number is stored on tags memory. To conduct a traceability attack, the next steps are followed by A:

1. A attacks a transaction between R_i and T_i:
   (a) A captures the query ($X = \{h(f(r_i, t_j) \| n_r) \oplus id_j,\ n_r,\ r_i\}$) sent by R_i and stores this value.
   (b) A frustrates the successful execution of the protocol by altering T_i answer to some random value.
2. As consequence of the above error, the reader repeats the search and A does not interfere in the protocol.
   (a) R_i sends query $X' = \{h(f(r_i, t_j) \| n'_r) \oplus id_j,\ n'_r,\ r_i\}$.
   (b) T_j checks X', updates $oldn = n'_r$ and replies $Y' = \{h(f(r_i, t_j) \| n'_t) \oplus id_j,\ n'_t\}$
3. To trace T_i, A replies X which uses a fresh n_r random value.
4. If any tag T^* answers to the adversary, A detects the presence of T_j again. In fact T_j answers – assuming it still presents there– because X is a valid token and a fresh value ($n_r \neq oldn (= n'_r)$)

Summarizing, following one of these strategies, an attacker puts at stake the privacy location. The risk is indeed maximal since the success probability is 1 and the complexity of running the attack is negligible. The attacks can be avoided by following well-known guidelines for designing cryptographic protocols. For instance Principles 4 and 5 in Abadi and Needham's guidelines are broken [1].

5.2 Traceability Attack on the Third Improved Search Protocol

The last solution the authors proposed to avoid traceability attacks consists on using noise to mask the reply of the target tag. In the protocol, each tag which receives a query and mismatches the request ($id \neq id_j$) will reply with some probability (λ). The authors stated that an adversary (A) does not obtain useful information to track a tag by replying a previous query since any tag could reply. Nevertheless, we show how is possible to conduct an efficient traceability attack against this protocol. The main observation for this attack is that when R_i searches T_j, then the target tag T_j will answer with probability 1 while any mismatched tag will reply with probability λ. To mount a traceability attack, assuming a population of N tags in the range, A does as follows:

1. A eavesdrops a transaction between R_i and T_j and captures the query ($X = \{h(f(r_i, t_j)\|n_r) \oplus id_j,\ n_r,\ r_i\}$).
2. A checks the presence of the target tag N_s times. Specifically,
   For $s = 1$ to N_s:
   (a) A replies the query X.
   (b) A counts the number of answered obtained, where c represents the counter.
      i. If T_j is present, it answers $Y = \{h(f(r_i, t_j)\|n'_t) \oplus id_j,\ n'_t\}$. The tag detects an answer and increments the counter ($c = c + 1$).
      ii. For the rest of the tags ($N - 1$), each tag answers $Y = \{rand, n'_t\}$ with probability λ. When a tag replies, A detects the answer and increments the counter ($c = c + 1$).
3. Finally, A uses this simple but quite effective decision rule:

$$\begin{cases} \text{If } c \geq Ns + (N-1) \cdot Ns \cdot \lambda & T_j \text{ is present} \\ \text{If } c < Ns + (N-1) \cdot Ns \cdot \lambda & T_j \text{ is not present} \end{cases}$$

On the above algorithm, at each sending of a query a mismatched tag answers with a probability of λ while the target tag replies with a probability of 1. If T_j is not present and A repeatedly replies N_s queries, A obtains on average $(N-1) \cdot N_s \cdot \lambda$ answers. When T_j is present, the above mentioned value will be $N_s + (N-1) \cdot N_s \cdot \lambda$. Hence, $(N-1) \cdot N_s \cdot \lambda$ can be interpreted as the average value of the "noise" inserted by the mismatched tags. So, the attacker running the above attack can track the target tag, just by counting the number of answers received. Nevertheless, to determine the success probability, we should consider the possible errors:

- $Error_1$: It denotes the case when T_j is present but the above threshold is not satisfied. In this case the algorithm will not trace the tag properly and Pr_{Error_1} represents the probability of this false alarm.
- $Error_2$: It denotes a case when T_j is not present but the above threshold is satisfied. In this case the algorithm wrongly alarms that T_j is present while it is not there. The probability of this false alarm is denoted as Pr_{Error_2}.

$Error_1$ only can happen if T_j is not the target tag. Hence, A has N mismatched tags in the range and each tag will reply with probability λ at each query. It can be modeled as a random process with a binomial distribution with parameters $p = \lambda$ and $n = N \cdot Ns$. $Error_1$ happens if:

$$c \geq Ns + (N-1) \cdot Ns \cdot \lambda$$

Hence, this error can be estimated as follows:

$$Pr_{Error_1} =$$

$$\sum_{i=Ns+(N-1)\cdot Ns\cdot\lambda}^{N\cdot Ns} \binom{N \cdot Ns}{i} \times \lambda^i \times (1-\lambda)^{(N\cdot Ns)-i}$$

On the other hand, $Error_2$ can only happen if T_j is there. Hence, A has $N-1$ mismatched tags in the range that will reply to each query with probability λ. It can be modeled as a random process with a binomial distribution with parameters $p = \lambda$ and $n = (N-1) \cdot Ns$. $Error_2$ happens when for Ns queries, the mismatched tags in the range reply less times than $(N-1) \cdot Ns \cdot \lambda$. Hence, Pr_{Error_2} can be estimated as described below:

$$Pr_{Error_2} =$$

$$\sum_{i=0}^{(N-1)\cdot Ns\cdot\lambda-1} \binom{(N-1)\cdot Ns}{i} \times \lambda^i \times (1-\lambda)^{((N-1)\cdot Ns)-i}$$

Assuming large values for N, it can be approximated as follows:

$$Pr_{Error_2} \cong$$

$$\sum_{i=0}^{(N-1)\cdot Ns\cdot\lambda-1} \binom{N\cdot Ns}{i} \times \lambda^i \times (1-\lambda)^{(N\cdot Ns)-i}$$

Given that $Error_1$ and $Error_2$ will never happen together and defining γ as the probability of the event that T_j is present, the total error probability (Pr_{Error}) can be determined as detailed below:

$$Pr_{Error} = Pr_{Error_1} \times (1-\gamma) + Pr_{Error_2} \times \gamma \leq$$

$$Pr_{Error_1} + Pr_{Error_2}$$

On the other hand, for a binomial distribution with parameters $p = \lambda$ and $n = N \cdot Ns$ repetition, we have the following equality:

$$\sum_{i=0}^{N\cdot Ns} \binom{N\cdot Ns}{i} \times \lambda^i \times (1-\lambda)^{(N\cdot Ns)-i} = 1$$

So, the probability of success ($Pr_{Suc.} = 1 - Pr_{Error}$) can be estimated as follows:

$$Pr_{Suc.} \cong$$

$$\sum_{i=(N-1)\cdot Ns\cdot\lambda}^{Ns+(N-1)\cdot Ns\cdot\lambda-1} \binom{N\cdot Ns}{i} \times \lambda^i \times (1-\lambda)^{(N\cdot Ns)-i}$$

In Table 1 the $Pr_{Suc.}$ for several values of λ, Ns and N has been depicted. It shows that the success probability of attack is considerable. The complexity of the described attack is N_s, which represents the number of queries an adversary has to send and then count the number of answers obtained.

Table 1. The success probability of the traceability attack on the third improved search protocol for different values of λ, N and Ns . In this table H and L denote $Ns + (N-1) \cdot Ns \cdot \lambda - 1$ and $(N-1) \cdot Ns \cdot \lambda$ respectively.

λ	Ns	N	H	L	$N \cdot Ns$	$Pr_{suc.}$
0.1	10	10	18	9	100	0.6745
0.1	100	10	189	90	1000	0.8666
0.1	200	10	379	180	2000	0.9385
0.1	10	100	108	99	1000	0.3731
0.1	100	100	1089	990	10000	0.6211
0.1	200	100	2179	1980	20000	0.6847
0.1	10	1000	1008	999	10000	0.1313
0.1	100	1000	10089	9990	100000	0.3708
0.1	200	1000	20179	19980	200000	0.46934
0.05	10	10	13.5	4.5	100	0.5635
0.05	100	10	144	45	1000	0.7853
0.05	200	10	289	90	2000	0.86
0.05	10	100	58.5	49.5	1000	0.4097
0.05	100	100	594	495	10000	0.5971
0.05	200	100	1189	990	20000	0.6317
0.05	10	1000	508.5	499.5	10000	0.1603
0.05	100	1000	5094	4995	100000	0.4454
0.05	200	1000	10189	9990	200000	0.5161
0.01	10	10	9.9	0.9	100	0.634
0.01	100	10	108	9	1000	0.6683
0.01	200	10	217	18	2000	0.7041
0.01	10	100	18.9	9.9	1000	0.5358
0.01	100	100	198	99	10000	0.5536
0.01	200	100	397	198	20000	0.5661
0.01	10	1000	108.9	99.9	10000	0.3183
0.01	100	1000	1098	999	100000	0.5159
0.01	200	1000	2197	1998	200000	0.5209

6 Impersonation Attacks on Search Protocols

Tan *et al.* [20] claim that the search protocols are resistant again cloning attacks. More precisely, they consider the skimming attack described in [11]. The attacker (A) starts querying T_j and capturing a response. She then copies the response on a fake $RFID$ tag ($\hat{T}_j$). A successes in her attempt when she tricks R_i into believing that $\hat{T}_j$ is T_j. The authors state that as result of using fresh random numbers n_r generated by the challenger (R_i), previous responses are not valid and counterfeit tags are detected.

We now show how the first and the third improved search protocols are vulnerable to impersonation attacks, fooling the reader about the presence of the target tag. More precisely, we propose an attack on-the-fly exploiting the symmetric between a query and an answer. That is, in the original protocols the authors ignored the Principle 4 for designing cryptographic protocols [1]. To conduct the attack, A follows the next steps:

1. A eavesdrops the query $X = \{h(f(r_i, t_j) \| n_r) \oplus id_j,\ n_r,\ r_i\}$ sent by R_i and stores this tuple.
2. In the future when A receives a request $X'$from R_i, she can impersonate T_j by replying $Y = \{X[1], X[2]\}$, where X[i] symbolizes the i-th element of X.
3. Finally, when R_i checks Y it is convinced of the presence of T_j.

So, an attacker just replies messages and simulates the presence of the target tag with a 100% of success. In the original protocols, the mistake is two fold: first the random number n_r generated by R_i (the challenger) does not take part in the response and secondly the frame of the query/answer message is symmetric.

7 Conclusions

In [20], the authors dealt with the design of an authentication protocol without requiring a permanent connection to a central database, which is a thought provoking challenge. Furthermore, Tan *et al.* introduced the problem of efficiently searching for an specific tag in a large population of tags. Nevertheless, we show how these proposals are insecure because an attacker can compromise the privacy of confidential information (*id* disclosure attack) and put at risk the privacy of location (traceability attack). Moreover, an attacker can easily trace and supplant a tag, ruining the usefulness of the search protocols. The complexity of the proposed attacks is negligible and the adversary's success probability is significant – sometimes even maximal as in the *id* disclosure or in the impersonation attack.

References

1. Abadi, M., Needham, R.M.: Prudent Engineering Practice for Cryptographic Protocols. IEEE Trans. Software Eng. 22(1), 6–15 (1996)
2. Black, J., Rogaway, P., Shrimpton, T.: Black-Box Analysis of the Block-Cipher-Based Hash-Function Constructions from PGV. In: Yung, M. (ed.) CRYPTO 2002. LNCS, vol. 2442, pp. 320–335. Springer, Heidelberg (2002)
3. Phan, R.C.-W.: Cryptanalysis of a New Ultralightweight RFID Authentication Protocol –SASI. IEEE Transactions on Dependable and Secure Computing 6, 316–320 (2009)
4. Damgård, I.B.: A Design Principle for Hash Functions. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 416–427. Springer, Heidelberg (1990)
5. Davies, D.W., Price, W.L.: The Application of Digital Signatures Based on Public-Key Cryptosystems. In: Proc. Fifth Intl. Computer Communications Conference, pp. 525–530 (October 1980)
6. Dean, R.D.: Formal Aspects of Mobile Code Security. PhD thesis, Princeton University (1999)
7. Feldhofer, M., Rechberger, C.: A Case Against Currently Used Hash Functions in RFID Protocols. In: Meersman, R., Tari, Z., Herrero, P. (eds.) OTM Workshops 2006. LNCS, vol. 4277, pp. 372–381. Springer, Heidelberg (2006)
8. FIPS. Secure Hash Standard. National Institute for Standards and Technology, pub-NIST:adr (August 2002)
9. Gauravaram, P., Knudsen, L.R.: On Randomizing Hash Functions to Strengthen the Security of Digital Signatures. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 88–105. Springer, Heidelberg (2009)
10. Halevi, S., Krawczyk, H.: Strengthening Digital Signatures Via Randomized Hashing. In: Dwork, C. (ed.) CRYPTO 2006. LNCS, vol. 4117, pp. 41–59. Springer, Heidelberg (2006)
11. Juels, A.: Strengthening EPC Tags Against Cloning. In: Proc. of WiSe 2005, pp. 67–76. ACM Press (2005)
12. Juels, A., Weis, S.: Defining Strong Privacy for RFID. In: Proc. of PerCom 2007, pp. 342–347. IEEE Computer Society Press (2007)
13. Kelsey, J., Schneier, B.: Second Preimages on n-Bit Hash Functions for Much Less than 2^n Work. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 474–490. Springer, Heidelberg (2005)

14. Lai, X., Massey, J.L.: Hash Functions Based on Block Ciphers. In: Rueppel, R.A. (ed.) EUROCRYPT 1992. LNCS, vol. 658, pp. 55–70. Springer, Heidelberg (1993)
15. Merkle, R.C.: One Way Hash Functions and DES. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 428–446. Springer, Heidelberg (1990)
16. National Institute of Standards and Technology. Secure hash standard (SHS). FIPS Publication 180 (May 1993)
17. Preneel, B.: Analysis and Design of Cryptographic Hash Functions. Thesis (Ph.D.), Katholieke Universiteit Leuven, Leuven, Belgium (January 1993)
18. Preneel, B., Govaerts, R., Vandewalle, J.: Hash Functions Based on Block Ciphers: A Synthetic Approach. In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 368–378. Springer, Heidelberg (1994)
19. Rivest, R.L.: RFC 1321: The MD5 Message-Digest Algorithm. Internet Activities Board (April 1992)
20. Tan, C.C., Sheng, B., Li, Q.: Secure and Serverless RFID Authentication and Search Protocols. IEEE Transactions on Wireless Communications 7(4), 1400–1407 (2008)

Appendix

A Notation

Through the paper, we use the following notation:

- R_i: RFID reader i.
- r_i: Static identifier of R_i.
- T_i: RFID tag i.
- id_j: Static identifier of T_i.
- t_i: Secret of T_i.
- n_r: Random number generated by the reader.
- n_t: Random number generated by the tag.
- L_i: Access list for R_i.
- n: Number of entries in L_i.
- $h(x)$: One-way hash function.
- $f(x, y)$: Concatenation of x and y, then applying $h(.)$, $h(x||y)$.
- CA: Trusted party, responsible for authenticating readers and deploying tags.
- m: Number of bits defined by CA, $m < l$.
- l: Output length of hash $h(.)$.
- $A \rightarrow B$: Sending a message from A to B.

B Tan *et al.* Protocols

$$
\begin{aligned}
R_i \rightarrow T_j &: request && (1)\\
R_i \leftarrow T_j &: n_j && (2)\\
R_i \rightarrow T_j &: n_i, r_i && (3)\\
R_i \leftarrow T_j &: h(f(r_i,t_j))_m, h(f(r_i,t_j)||n_i||n_j) \oplus id_j && (4)\\
R_i &: \text{Hash every entry in } L_i \text{ and check} \\
&\quad \text{if first } m \text{ bits match } h(f(r_i,t_j))_m && (5)\\
R_i &: \text{Checks } L_i \text{ for matching } h(f(r_i,t_j))_m && (6)\\
R_i &: \text{Determine } h(f(r_i,t_j)||n_i||n_j), \text{ obtain } id_j && (7)
\end{aligned}
$$

Fig. 1. The mutual authentication protocol proposed by Tan *et al.* [20]

$$
\begin{aligned}
R_i \rightarrow T* &: h(f(r_i,t_j)||n_r) \oplus id_j, n_r, r_i && (1)\\
T* &: \text{Derive } h(f(r_i,t)||n_r) \text{ and XOR with} \\
&\quad h(f(r_i,t_j)||n_r) \oplus id_j && (2)\\
&: \text{If } id = id_j && (3)\\
R_i \leftarrow T_j &: \quad h(f(r_i,t_j)||n_t||n_r) \oplus id_j, n_t && (4)
\end{aligned}
$$

Fig. 2. The basic search protocol proposed by Tan *et al.* [20]

$$
\begin{aligned}
R_i \rightarrow T* &: h(f(r_i,t_j)||n_r) \oplus id_j, n_r, r_i && (1)\\
T* &: \text{Deriving } h(f(r_i,t)||n_r) \text{ and XOR with} \\
&\quad h(f(r_i,t_j)||n_r) \oplus id_j && (2)\\
&: \text{If } id = id_j \text{ and } n_r \neq oldn, \\
&\quad \text{update } oldn = n_r && (3)\\
R_i \leftarrow T_j &: \quad h(f(r_i,t_j)||n_t) \oplus id_j, n_t && (4)
\end{aligned}
$$

Fig. 3. The first improved search protocol proposed by Tan *et al.* [20]

$$
\begin{aligned}
R_i \rightarrow T* &: \text{Broadcast } [id_j]_m, r_i, n_r && (1)\\
T* &: \text{If } id_m = [id_j]_m && (2)\\
R_i \leftarrow T_j &: \quad h(f(r_i,t_j)||n_r||n_t) \oplus id_j, n_t && (3)\\
R_i &: \text{Determines } f(r_i,t_j) \text{ from } L, \text{ obtain } id_j && (4)
\end{aligned}
$$

Fig. 4. The second improved search protocol proposed by Tan *et al.* [20]

$$
\begin{aligned}
R_i \rightarrow T* &: \text{Broadcast } h(f(r_i,t_j)||n_r) \oplus id_j, n_r, r_i && (1)\\
T* &: \text{Derive } h(f(r_i,t)||n_r) \text{ and XOR with} \\
&\quad h(f(r_i,t_j)||n_r) \oplus id_j && (2)\\
&: \text{If } id = id_j : \\
&\quad R_i \leftarrow T_j : h(f(r_i,t_j)||n_t) \oplus id_j, n_t && (3)\\
&: \text{Else :} \\
&\quad R_i \leftarrow T_j : (rand, n_t) \text{ with prob. } \lambda && (4)
\end{aligned}
$$

Fig. 5. The third improved search protocol proposed by Tan *et al.* [20]

C Privacy Model

In RFID schemes, tags (T) and readers (R) interact in protocol sessions. In general terms, the adversary (A) controls the communications between all the

participants and interacts passively or actively with them. Specifically, A can run the following queries:

- Execute(R_i, T_j, k) query. This models a passive attacker. A eavesdrops on the channel, and gets read access to the exchanged messages between R_i and T_j in session k of a genuine protocol execution.
- Test(k', T_0, T_1) query. This does not model any ability of A, but it is necessary to define the untraceability test. When this query is invoked for session k', a random bit is generated $b \in \{0,1\}$. Then, the tokens $\{X_b, Y_b, Z_b, \ldots\}$ from the set $\{\{X_0, Y_0, Z_0, \ldots\}, \{X_1, Y_1, Z_1, \ldots\}\}$ corresponding to tags $\{T_0, T_1\}$ are given to $\mathcal{A}$.

Upon definition of the adversary's abilities, the untraceability problem can be defined as a game G divided into three phases:

- Phase-1 (Learning): A can make any number of Execute queries, which facilitates the eavesdropping of exchanged messages – modelling a passive attacker – over the insecure radio channel.
- Phase-2 (Challenge): A chooses two fresh tags $\{T_0, T_1\}$ whose associated identifiers and keys are $\{id_0, t_0\}$ and $\{id_1, t_1\}$, respectively. He then sends a Test(k', T_0, T_1) query. As a result, and depending on a chosen random bit $b \in \{0,1\}$, A is given the tokens $\{X_b, Y_b, Z_b, \ldots\}$ from the set $\{\{X_0, Y_0, Z_0, \ldots\}, \{X_1, Y_1, Z_1, \ldots\}\}$.
- Phase-3 (Guessing): A ends the game and outputs a bit $\widetilde{b}$ as its conjecture of the value of b.

A's success in winning G is equivalent to the success of breaking the untraceability property offered by the protocol. So the advantage of A in distinguishing whether the messages correspond to T_0 or T_1 is defined as below:

$$Adv_A^{\mathsf{UNT}}(q, kr) = |Pr[\widetilde{b} = b] - \frac{1}{2}|$$

where q is a security parameter (i.e. the bit length of the key shared between the tag and the reader) and kr is the number of times $\mathcal{A}$ runs an Execute query.

Definition 1. *An RFID protocol in an RFID system (S= {R_i, T_0, T_1, ...}) in which an adversary A can invoke* {Execute*(R_i, T_j, k),* Test*(k', T_0, T_1)*} *in a game G, offers resistance against traceability if:*

$$Adv_A^{\mathsf{UNT}}(q, kr) < \varepsilon(q, kr) \tag{3}$$

$\varepsilon(.)$ *being some negligible function.*

D Hash Function Model

A cryptographic hash function maps messages of arbitrary length to fixed-length message digests (hash values). Commonly, to compress messages of arbitrary

length to fixed-length digests, a fixed-input-length compression function is used in a mode of operation. The most commonly used mode of operation is the Merkle-Damgård hash construction [4,15].

The Merkle-Damgård (MD) is a well known hash construction, which is used in almost all popular hash functions such as MD5 [19], SHA-1 [16] and SHA-2 [8]. Given a message $M = M_1 \| ... \| M_l$ and $g : \{0,1\}^n \times \{0,1\}^m \rightarrow \{0,1\}^n$ as a compression one-way function, MD hash function computes the hash value of M as follows:

$$H^g(M) = g(g(...g(IV, M_1), ...), M_l)$$

where IV is the initial value.

In this paper we assume that the hash functions used in the mutual authentication protocol and the search protocols are based on Merkle-Damgå construction with *fixed IV* but without employing *MD strengthening*. It should be noted that the MD construction has a security reduction [4,15], showing how a collision for the hash function entails a collision for the compression function. This is achieved by including the length of the message as part of the message padding. This technique is called *MD strengthening* [14]. The security reduction of the MD construction is valid for arbitrary initial values (IVs). On the other hand, Damgård [4] also observed that a similar reduction is possible in an iterated hash function construction when the IV is fixed but the message length is not appended. In [17], Preneel recommended to fix the IV as well as to employ MD strengthening, and we can find this approach in many standard hash functions (e.g., SHA and RIPEMD families). Nevertheless, the security of the hash function is not compromise whether the IV is fixed and MD strengthening is not used. This second approach is what we do in our proposed attack. Moreover, this approach sounds more realistic for an RFID application as we can avoid extra calls to the compression function.

Compression one-way functions are the cryptographic primitives that hashes fixed-length-input messages to the fixed-length hash values. These primitives are generally used as the building blocks in a mode of operation (e.g. MD) to design hash functions that process messages of arbitrary length. A common approach to build one-way compression functions is to use block ciphers. Preneel, Govaerts and Vandewalle (PGV) [18] showed sixty-four ways of building compression function modes from block ciphers and twelve of these were shown to be both collision and (second) preimage resistant. Henceforth, these schemes are known as PGV compression functions. Some years later, Black, Rogaway and Shrimpton [2] formally showed that these twelve compression functions are collision and (second) preimage resistant in the ideal cipher model. These compression functions are represented in Table 2, where E symbolizes an ideal block cipher. A general form of these compression functions are as described below:

$$g(H_i, M_i) = E_{K_E}(PT_E) \oplus U$$

where the plaintext PT_E, the key K_E and the feed-forward value U belong to the set $\{M_i, H_i, M_i \oplus H_i, v\}$ and v is a known constant value.

Table 2. PGV compression functions. The column with $\imath$ represents the PGV schemes that are collision and (second) preimage resistant [2]. The column with $\jmath$ represents the numbering of the compression functions as in [18], which was also put forward in [2]. In this Table, $w_i = m_i \oplus H_{i-1}$ and v is a constant value.

$\imath$ [2]	$\jmath$ [18]	PGV: $h_i =$
	1	$E_{m_i}(m_i) \oplus v$
	2	$E_{H_{i-1}}(m_i) \oplus v$
	3	$E_{w_i}(m_i) \oplus v$
	4	$E_v(m_i) \oplus v$
	5	$E_{m_i}(m_i) \oplus m_i$
1	6	$E_{H_{i-1}}(m_i) \oplus m_i$
9	7	$E_{w_i}(m_i) \oplus m_i$
	8	$E_v(m_i) \oplus m_i$
	9	$E_{m_i}(m_i) \oplus H_{i-1}$
	10	$E_{H_{i-1}}(m_i) \oplus H_{i-1}$
11	11	$E_{w_i}(m_i) \oplus H_{i-1}$
	12	$E_v(m_i) \oplus H_{i-1}$
	13	$E_{m_i}(m_i) \oplus w_i$
3	14	$E_{H_{i-1}}(m_i) \oplus w_i$
	15	$E_{w_i}(m_i) \oplus w_i$
	16	$E_v(m_i) \oplus w_i$
	17	$E_{m_i}(H_{i-1}) \oplus v$
	18	$E_{H_{i-1}}(H_{i-1}) \oplus v$
	19	$E_{w_i}(H_{i-1}) \oplus v$
	20	$E_v(H_{i-1}) \oplus v$
	21	$E_{m_i}(H_{i-1}) \oplus m_i$
	22	$E_{H_{i-1}}(H_{i-1}) \oplus m_i$
12	23	$E_{w_i}(H_{i-1}) \oplus m_i$
	24	$E_v(H_{i-1}) \oplus m_i$
5	25	$E_{m_i}(H_{i-1}) \oplus H_{i-1}$
	26	$E_{H_{i-1}}(H_{i-1}) \oplus H_{i-1}$
10	27	$E_{w_i}(H_{i-1}) \oplus H_{i-1}$
	28	$E_v(H_{i-1}) \oplus H_{i-1}$
7	29	$E_{m_i}(H_{i-1}) \oplus w_i$
	30	$E_{H_{i-1}}(H_{i-1}) \oplus w_i$
	31	$E_{w_i}(H_{i-1}) \oplus w_i$
	32	$E_v(H_{i-1}) \oplus w_i$
	33	$E_{m_i}(w_i) \oplus v$
	34	$E_{H_{i-1}}(w_i) \oplus v$
	35	$E_{w_i}(w_i) \oplus v$
	36	$E_v(w_i) \oplus v$
	37	$E_{m_i}(w_i) \oplus m_i$
4	38	$E_{h_i}(w_i) \oplus m_i$
	39	$E_{w_i}(w_i) \oplus m_i$
	40	$E_v(w_i) \oplus m_i$
8	41	$E_{m_i}(w_i) \oplus H_{i-1}$
	42	$E_{H_{i-1}}(w_i) \oplus H_{i-1}$
	43	$E_{w_i}(w_i) \oplus H_{i-1}$
	44	$E_v(w_i) \oplus H_{i-1}$
6	45	$E_{m_i}(w_i) \oplus w_i$
2	46	$E_{H_{i-1}}(w_i) \oplus w_i$
	47	$E_{w_i}(w_i) \oplus w_i$
	48	$E_v(w_i) \oplus w_i$
	49	$E_{m_i}(v) \oplus v$
	50	$E_{H_{i-1}}(v) \oplus v$
	51	$E_{w_i}(v) \oplus v$
	52	$E_v(v) \oplus v$
	53	$E_{m_i}(v) \oplus m_i$
	54	$E_{H_{i-1}}(v) \oplus m_i$
	55	$E_{w_i}(v) \oplus m_i$
	56	$E_v(v) \oplus m_i$
	57	$E_{m_i}(v) \oplus H_{i-1}$
	58	$E_{H_{i-1}}(v) \oplus H_{i-1}$
	59	$E_{w_i}(v) \oplus H_{i-1}$
	60	$E_v(v) \oplus H_{i-1}$
	61	$E_{m_i}(v) \oplus w_i$
	62	$E_{H_{i-1}}(v) \oplus w_i$
	63	$E_{w_i}(v) \oplus w_i$
	64	$E_v(v) \oplus w_i$

In this article, we consider a group of PGV compression functions that use M_i as the K_E. For instance, one of these compression functions is known as Davies Mayer (DM) [5], which is indexed by Preneel *et al.* [18] as the 25^{th} scheme and by Black *et al.* [2] as the 5^{th} scheme. Given a block cipher E, this compression function accepts the i^{th} chaining value H_i and a message block M_i and compresses these as follows:

$$g(H_i, M_i) = E_{M_i}(H_i) \oplus H_i$$

Other secure schemes that we consider in this article are the schemes 6^{th}, 7^{th} and 8^{th} – following the Black *et al.* [2] indexing.

Yet Another Ultralightweight Authentication Protocol That Is Broken

Gildas Avoine and Xavier Carpent

Université catholique de Louvain
B-1348 Louvain-la-Neuve
Belgium

Abstract. Eghdamian and Samsudin published at ICIEIS 2011 an ultralightweight mutual authentication protocol that requires few bitwise operations. The simplicity of the design makes the protocol very suitable to low-cost RFID tags. However, we demonstrate in this paper that the long-term key shared by the reader and the tag can be recovered by an adversary with a few eavesdropped sessions only.

Additionally, we provide the backbone of some attacks on a series of similar recent protocols, and highlight important common weaknesses in the design of ultralightweight protocols.

Keywords: Authentication, Ultralightweight protocol, RFID.

1 Introduction

The market pressure to lower the price of tags is such that it has become a major topic of research to design an RFID protocol requiring very few gates and little computational power on the tag side. Several families of protocols have been proposed, such as the influential HB family (see [5] for a thorough presentation of the HB family), and other "human authentication" protocols. In [11], Peris-Lopez, Hernandez-Castro, Estevez-Tapiador, and Ribagorda introduced a mutual protocol, called LMAP, which is the first of what came to be known as the "ultralightweight protocols family". Many proposals followed (see [2] for a comprehensive introduction to this protocol family), but almost all of them have been broken. These protocols rely on very simple building blocks, such as bitwise operations ($\oplus$,$\vee$, $\wedge$), modular addition ($+$), or data-dependent rotations ($\mathrm{Rot}(x, y)$). They often do not require the tag to generate randomness, and require tags to update their state every successful authentication.

Recently, Eghdamian and Samsudin proposed a new protocol in that family, claiming more security than its predecessors. We show in this paper how a passive attack can recover the 96-bit secret of a tag, using only 20 authentication sessions on average.

We also show similar attacks on RPAP (by Ning, Liu and Yang [10]), PUMAP (by Bassil, El-Beaino, Itani, Kayssi and Chehab [4]), and DIDRFID and SIDFRID (by Lee [9]). We finally point out traceability attacks on RAPP (by Tian, Chen and Li [13]), and Improved LMAP+ (by Gurubani, Thakkar and Patel [8]).

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 20–30, 2013.

The highlighted attacks show once more that most of the protocols of this class can be broken with little effort.

The paper is divided as follows. In Sect. 2, we present Eghdamian and Samsudin's protocol. Our attack on it is thoroughly described in Sect. 3. In Sect. 4, we briefly describe a series of other ultralightweight protocols and miscellaneous attacks on them. We highlight some common weaknesses in the design of ultralightweight protocols. We finally conclude in Sect. 6.

2 Eghdamian and Samsudin's Protocol

The protocol designed by Eghdamian and Samsudin [7] consists of four messages, represented on Fig. 1. First of all, the reader sends an hello message, then the

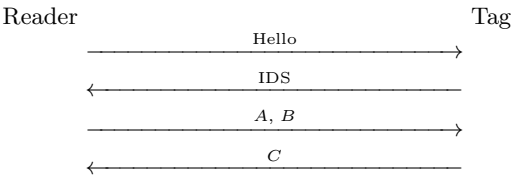


Fig. 1. Eghdamian and Samsudin's Protocol

tag sends its IDS. This IDS allows the reader to identify the tag and find the corresponding key K. If this identification step fails for some reason (error on the channel, tag not synchronized, false IDS), the reader sends a second request, to which the tag responds the old value of IDS. After the identification stage, the reader sends A and B, and the tag C. The content of A, B, and C is as follows:

$$A = K \oplus N \tag{1}$$

$$B = \mathrm{Rot}(K, N) \wedge \mathrm{Rot}(N, K) \wedge \mathrm{Rot}(N, N) \tag{2}$$

$$C = \mathrm{Rot}(K + \mathrm{Rot}(N, N), \mathrm{Rot}(K, K) \vee N) \tag{3}$$

where $\mathrm{Rot}(X, Y)$ means that X is rotated of $\mathcal{H}(Y)$ bits to the left, where $\mathcal{H}(Y)$ denotes the Hamming weight of Y. The symbol N represents a random value. After a successful authentication, the tag updates its key and session identifier as follows:

$$K^{next} = \mathrm{Rot}(N + \mathrm{Rot}(K, K), \mathrm{Rot}(N, N) \wedge K) \tag{4}$$

$$\mathrm{IDS}^{next} = K \wedge \mathrm{Rot}(N, K \vee N) \tag{5}$$

Let L denote the length of all the variables (recommended to be 96 in [7]):

$$|K| = |N| = |A| = |B| = |C| = |IDS| = L.$$

3 Attack on Eghdamian and Samsudin's Protocol

We introduce in this section a key-recovery attack that allows an adversary to recover the key K shared by the reader and the tag. The attack requires a passive adversary to eavesdrop one authentication session where a property on the Hamming weight of N is ensured, as detailed below. If the adversary is active and knows the current IDS of her target, she can perform her attack without the presence of the targeted tag.

3.1 Discovering the Hamming Weight of N

The first step of the attack aims to recover $\mathcal{H}(N)$. Below B_i denotes the bit at index i of B, with B_0 being the least significant bit of B. From Eq. (2), we know that:

$$\forall i, 0 \leq i < L, (B_i = 1) \Rightarrow (K_{i-\mathcal{H}(N) \bmod L} = N_{i-\mathcal{H}(N) \bmod L} = 1).$$

Using Eq (1), we deduce:

$$\forall i, 0 \leq i < L, (B_i = 1) \Rightarrow (A_{i-\mathcal{H}(N) \bmod L} = 0). \tag{6}$$

Consequently, a candidate r for $\mathcal{H}(N)$ is discarded if Eq (6) is not satisfied. If only one candidate r among the n possible ones remains, then $\mathcal{H}(N) = r$. Experimentally, we observed that this case occurs with a probability close to 0.9 when $L = 96$. When more that one candidate remain, the adversary can keep the few candidates and discard the wrong ones later in the attack, or she can simply eavesdrop another authentication session in order to be luckier and obtain a single candidate.
We consider from now on that the adversary knows $\mathcal{H}(N)$.

3.2 Recovering Half of the Secret Bits

The adversary assumes that $\mathcal{H}(K) = \mathcal{H}(N)$. This assumption will be denoted H_1 in the following. Whenever H_1 is true, Eq (2) yields:

$$B = \mathrm{Rot}(K, N) \wedge \mathrm{Rot}(N, N),$$

and so:

$$\mathrm{Rot}^{-1}(B, N) = K \wedge N. \tag{7}$$

where Rot^{-1} means the right-rotation. We will denote below:

$$\widetilde{B} := \mathrm{Rot}^{-1}(B, N).$$

From Eq (1), we know that $A_i = 0$ implies that either $K_i = N_i = 0$ or $K_i = N_i = 1$. Consequently:

$$\forall i, 0 \leq i < L, (A_i = 0) \Rightarrow (K_i = \widetilde{B}_i).$$

This technique allows the adversary to recover half of the secret bits on average. Given that $\mathcal{H}$ follows a binomial distribution, Vandermonde's identity allows to demonstrate that the assumption H_1 actually occurs with probability $\binom{2L}{L}/2^{2L}$. When $L = 96$, this value is close to 0.058, which implies that the adversary should eavesdrop about 18 authentication sessions on average in order to observe one where the property $\mathcal{H}(N) = \mathcal{H}(K)$ is satisfied.

3.3 Recovering More Secret Bits

The adversary can increase the number of revealed bits of the secret key by exploiting the IDS following the session where H_1 is satisfied. Indeed, we know from Eq (5) that:

$$\mathrm{IDS}^{next} = K \wedge \mathrm{Rot}(N, K \vee N).$$

We conclude that

$$\forall i, 0 \leq i < L, \left(\mathrm{IDS}_i^{next} = 1\right) \Rightarrow \left(K_i = 1\right). \tag{8}$$

3.4 Recovering Still More Secret Bits

Once some bits of K and N are known, the adversary can exploit them to recover more bits of K. For that, we can first trivially notice that:

$$K \vee N = (K \wedge N) \vee (K \oplus N). \tag{9}$$

When H_1 holds, we deduce, by inserting Eq (1) and Eq (7) in Eq (9):

$$K \vee N = A \vee \widetilde{B}. \tag{10}$$

Therefore, Eq (5) can be rewritten using Eq (10) as:

$$\mathrm{IDS}^{next} = K \wedge \mathrm{Rot}(N, A \vee \widetilde{B}). \tag{11}$$

If the adversary already knows i such that $K_i = 1$ then using Eq (1) and Eq (11), we deduce:

$$K_{i-\mathcal{H}(A \vee \widetilde{B})} = A_{i-\mathcal{H}(A \vee \widetilde{B})} \oplus \mathrm{IDS}_i^{next}. \tag{12}$$

Likewise, if the adversary already knows i such that $K_{i-\mathcal{H}(A \vee \widetilde{B})} \oplus A_{i-\mathcal{H}(A \vee \widetilde{B})} = 1$ then using Eq (1) and Eq (11), we deduce:

$$K_i = \mathrm{IDS}_i^{next}. \tag{13}$$

These two last steps can further be iterated a few times, until no more information can be gathered. At that point, most of the bits of K are known. We have observed experimentally that an average of 73 bits of K are discovered.

3.5 Recovering the Remaining Secret Bits with a Passive Adversary

If the adversary is passive, she can recover the remaining secret bits performing a reasonable exhaustive search on the 23 unknown bits (on average). Candidates can be tested on C and B. If no suitable candidate is found in the exhaustive search, then the hypothesis $\mathcal{H}(K) = \mathcal{H}(N)$ was wrong, and another authentication attempt must be eavesdropped on.

3.6 Recovering the Remaining Secret Bits with an Active Adversary

An active adversary can block the message C in order to cancel the update on the reader side, and thus force the tag to use the same IDS and K in the following session. This allows her to collect A, B, C messages for the same K, but different N, and therefore guess all the bits of K, with no exhaustive search required.

4 Attacks on Other Protocols

In this section, several privacy and key-recovery attacks on a series of recent similar protocols are introduced. The protocols are not fully described but, instead, the key-points in their design that open the door for an attack are highlighted.

4.1 Ning, Liu and Yang's Protocol

RPAP was proposed by Ning, Liu and Yang in [10]. The main novelty is that the secret between the reader and a tag is partitioned into three sub-secrets, and the way the partition is done depends on a parameter d chosen and sent by the reader. The secret S is partitioned such that:

$$\begin{aligned} S_1 &= [S]_{L-d:L-1} \\ S_2 &= [S]_{d:L-d-1} \\ S_3 &= [S]_{0:d-1}, \end{aligned}$$

with $[x]_{a:b}$ denoting the number comprised of bits of x from a to b. The sub-secrets are 0-padded on the most significant bits when appropriate. Note that no information was given in [10] regarding how the reader should choose d, other than ranging from 1 to $L/2$.

A first important weakness is the way the message D (sent by the tag) is designed:

$$D = (S'_1 \vee S'_2) \oplus S_3,$$

with S'_1 and S'_2 defined as:

$$\begin{aligned} S'_1 &= \mathrm{Rot}(S_1, r_1, d) \\ S'_2 &= \mathrm{Rot}(S_2, r_2, d) \end{aligned}$$

A good estimator for each bit of S_3 is $\overline{D}$ (with probability of 3/4). In a handful of runs, an eavesdropper can thus easily recover the lower half of S.

There are other weaknesses in the design of the protocol that can help an attacker discover most of the secret S in few protocol runs. For instance, after recovering partially S, an adversary knows $(S'_1 \vee S'_2)$ from D. Therefore, when $[D \oplus S_3]_i = 0$, then $[S'_1]_i = [S'_2]_i = 0$ (where $[x]_i$ denotes the i-th bit of x), and when $[D \oplus S_3]_i = 1$, then $[S'_1]_i = [S'_2]_i = 1$ with probability 2/3. This gives further information on $S_1 \oplus r_1$ and $S_2 \oplus r_2$, which, in conjunction with other weaknesses gives information on S_1 and S_2.

One such other weakness lies in the message A (sent by the reader) which is built as:

$$A = (IDS_T \vee S_1) \oplus r_1.$$

Since IDS_T is public (it plays the same role as IDS in [7]), the adversary can easily get half of the bits of r_1, and the other half of $S_1 \oplus r_1$ on average at each run. The construction of B (sent by the reader) is also weak:

$$B = IDS_T \oplus (S_2 + r_2).$$

Here, IDS_T is essentially useless since public, and the adversary gets $S_2 + r_2$ trivially.

While all these issues are not important on their own (except the first one), they are very dangerous when considered together, and allow an eavesdropper to recover most bits of S in a few runs.

4.2 Bassil, El-Beaino, Itani, Kayssi and Chehab's Protocol

Bassil, El-Beaino, Itani, Kayssi and Chehab proposed in [4] a new authentication for RFID using PUF's (physically unclonable functions), called PUMAP. Regardless of the use of PUF's, the protocol uses constructions that are similar to other ultralightweight authentication protocols.

PUMAP follows the same scheme as Eghdamian and Samsudin's Protocol (see Fig. 1). The reader sends messages A,B and C to the tag, which are defined as follows:

$$\begin{aligned} A &= SVT \oplus SVR \oplus n_1 \\ B &= \mathrm{Rot}(SVR + n_2, SVT) \\ C &= \mathrm{Rot}(SVT \oplus SVR \oplus n_1, n_2), \end{aligned}$$

where $\mathrm{Rot}(X, Y)$ here means that X is rotated by $(Y \bmod L)$ bits to the left, and SVT and SVR are essentially the analogues of respectively IDS and K in [7]. The former is thus public, the latter secret. The other values are nonces.

The first attack we suggest is an active desynchronization one. Note that C is simply $\mathrm{Rot}(A, n_2)$. This means that an adversary has a probability of $1/L$ of forging a valid (i.e., one accepted by the tag) triplet (A, B, C) if she just sends a triplet (X, Y, X) with X and Y being arbitrary values. When receiving one such triplet, a tag updates SVT and SVR, and desynchronizes with the system.

An adversary just has to keep sending forged triplets until one is accepted. She needs to do this L times on average.

The second attack allows an eavesdropper to guess the next SVR at each run, and thus to trace and/or impersonate a tag. The messages sent by the tag after receiving A, B and C are defined as[1]:

$$\begin{aligned} D &= \mathrm{Rot}(\mathrm{Rot}(n_1 + (n_2 \oplus SVT) + SVR, n_2), n_1) \\ E &= \mathrm{Rot}(SVT^{next} \oplus n_2, n_1) \\ F &= \mathrm{Rot}(SVR^{next} \oplus n_1, n_2) \end{aligned}$$

Note that there are only L possibilities for the rotation in E. An eavesdropper getting SVT^{next} on the next session (or skimming the tag) thus has L candidates for n_2. Using B and then A, she gets the corresponding candidates for SVR and n_1. These candidate triplets (n_2, SVR, n_1) can then be tested against D. Once a correct set of values has been found, SVR^{next} can be obtained from F and n_1.

4.3 DIDRFID and SIDRFID

In [9], Lee presents two new ultralightweight authentication protocols, DIDRFID and SIDRFID. We present a full key-recovery attack on each of them. Rotations are used in both protocols, and use the Hamming weight of the second argument, much like the ones in [7].

The equations relevant for the attack in DIDRFID are the following:

$$\begin{aligned} A &= K \oplus R \\ DIDT^{next} &= \mathrm{Rot}(R, R \vee K) \oplus \mathrm{Rot}(K, R \wedge K) \\ K^{next} &= \mathrm{Rot}(R, R \wedge K) \oplus \mathrm{Rot}(K, R \vee K), \end{aligned}$$

where $DIDT$ is the equivalent of IDS in [7], K is the secret key, and R is a nonce. We thus have that

$$DIDT^{next} \oplus K^{next} = \mathrm{Rot}(A, R \vee K) \oplus \mathrm{Rot}(A, R \wedge K).$$

There are thus L^2 possibilities for K^{next}, which can be tested on the next session. Moreover, given the biased nature of the rotations, and given that the rotations are using Hamming weights, an eavesdropper usually needs much less than L^2 guesses. An eavesdropper thus gets the whole key of a tag by simply listening to one protocol run.

We will not detail SIDRFID, because the protocol uses a master key in the tag. This solution is dangerous because an adversary, after compromising a single tag, obtains this master key. She can then impersonate any tag in the system after eavesdropping one single protocol run with her victim.

[1] Note that there is an unmatched bracket for D in [4], but both attacks work regardless.

4.4 RAPP

Tian, Chen and Li introduce in [13] a new building block for ultralightweight protocols, as well as a new protocol using it, called RAPP. The new operator is called the permutation Per. We do not cover its definition here and refer the interested reader to the original paper. However, one bad feature of this construction, as pointed by the authors, is that it is Hamming weight-invariant (much like the rotations). We provide a traceability attack that highlight the weakness of this new operator and the design of RAPP.

The relevant equations are:

$$\begin{aligned} A &= \mathrm{Per}(K_2, K_1) \oplus n_1 \\ C &= \mathrm{Per}(n_1 \oplus K_1, n_1 \oplus K_3) \oplus ID \\ K_1^{next} &= \mathrm{Per}(K_1, n_1) \oplus K_2 \\ K_2^{next} &= \mathrm{Per}(K_2, n_2) \oplus K_1, \end{aligned}$$

where n_1 is a nonce. We point out the following fact:

$$\mathcal{H}(x \oplus y) = \mathcal{H}(x) + \mathcal{H}(y) - 2\mathcal{H}(x \wedge y),$$

for any x, y. As a corollary, we have that

$$\mathcal{H}(x \oplus y) \equiv \mathcal{H}(x) \oplus \mathcal{H}(y) \pmod 2.$$

This result has the following implications in RAPP:

$$\begin{aligned} \mathcal{H}(K_1^{next}) &\equiv \mathcal{H}(K_1) \oplus \mathcal{H}(K_2) \pmod 2 \\ \mathcal{H}(K_2^{next}) &\equiv \mathcal{H}(K_2) \oplus \mathcal{H}(K_1) \pmod 2. \end{aligned}$$

This implies that, after the very first run of the protocol, we have that $\mathcal{H}(K_1) \equiv \mathcal{H}(K_2) \equiv 0 \pmod 2$. Furthermore,

$$\begin{aligned} \mathcal{H}(A) &\equiv \mathcal{H}(K_2) \oplus \mathcal{H}(n_1) \pmod 2 \\ \mathcal{H}(C) &\equiv \mathcal{H}(n_1) \oplus \mathcal{H}(K_1) \oplus \mathcal{H}(ID) \pmod 2. \end{aligned}$$

An eavesdropper therefore gets easily that $\mathcal{H}(ID) \equiv \mathcal{H}(A) \oplus \mathcal{H}(C) \pmod 2$. This allows her to trace a tag.

4.5 Improved LMAP+

In [8], Gurubani, Thakkar and Patel propose an improved version of LMAP+, itself an extension of LMAP [11]. The improved LMAP+ is supposed to guarantee untraceability, but we show that this is not the case.

The messages in Improved LMAP+ are the following:

$$\begin{aligned} A &= (PID \oplus K_1) + r \\ B &= PID + K_2 + r \\ C &= PID \oplus (K_3 + r) \\ PID^{next} &= (PID \oplus r) + K_1 + K_2 + K_3 \\ K_1^{next} &= (K_1 \oplus r) + PID^{next} + K_2 \\ K_2^{next} &= (K_2 \oplus r) + PID^{next} + K_3 \\ K_3^{next} &= (K_3 \oplus r) + PID^{next} + K_1, \end{aligned}$$

where PID plays the same role as IDS in [7] and r is a nonce. A very natural thing to do when analyzing messages containing both XOR's and modular additions such as these is to look at the least significant bit (LSB) position (transforming the sums in XOR's). This allows to note that:

$$PID^{(n+2)} = r^{(n)} \oplus r^{(n+1)},$$

where the notation at the exponent is used to denote the value of that variable at a given protocol run. An eavesdropper can thus get the LSB of each nonce by making a single hypothesis on an initial value. The LSB of the keys can then be obtained using:

$$\begin{aligned} \mathrm{lsb}(K_1) &= \mathrm{lsb}(A \oplus PID \oplus r) \\ \mathrm{lsb}(K_2) &= \mathrm{lsb}(B \oplus PID \oplus r) \\ \mathrm{lsb}(K_3) &= \mathrm{lsb}(C \oplus PID \oplus r) \\ \mathrm{lsb}(K_1^{next}) &= \mathrm{lsb}(K_1 \oplus r \oplus PID^{next} \oplus K_2) \\ \mathrm{lsb}(K_2^{next}) &= \mathrm{lsb}(K_2 \oplus r \oplus PID^{next} \oplus K_3) \\ \mathrm{lsb}(K_3^{next}) &= \mathrm{lsb}(K_3 \oplus r \oplus PID^{next} \oplus K_1), \end{aligned}$$

which allows an eavesdropper to trace a tag. Although this has not been verified, we believe a full recovery attack could also be done using the same technique as the attack on LMAP by Bárász, Boros, Ligeti, Lója and Nagy [3], that is, further guess the bit just after the LSB, than the one after that, and so on.

5 Discussion on Weaknesses

From the weaknesses exploited in this paper, we can highlight some weak constructions.

The use of biased operations such as OR ($\vee$) and AND ($\wedge$) has often led to vulnerabilities (see [1,3] for instance, as well as the attacks presented in this paper). Although they bring non-linearity, and seem good when combined to other types of operations, an attacker may exploit the bias when used on their own, or weakly "shielded".

The combined use of modular additions ($+$) and XOR ($\oplus$) seems good, but it has been proved to be weak in some cases (see the attack on LMAP [3] for instance). One major point is that the modular addition is a XOR in the least significant bit, and the leakage of this bit is enough for performing a privacy attack. Moreover, if an adversary knows the least significant bit of the operands, the second bit can usually be guessed as well and so on. It has also been shown that when the operands are biased or partially known, information can be gathered on their sum the same way it can be done with their XOR ([2]).

Data-dependent rotations have been allegedly first used for RFID protocols in SASI ([6]), and are since then often part of the building blocks used in ultralightweight protocols (either using the modular or the Hamming weight version). It has been shown repeatedly that although they bring non-linearity at a cheap cost, they are dangerous if carelessly used. The output only has L possible outcomes, which makes guessing and trying an easy task.

Operations affecting the Hamming weight (such as OR and XOR) or other external measures are sometimes problematic. On the contrary, some operations such as rotations and permutations from [13] are Hamming weight-preserving, which allows an adversary to guess some information on the operands, allowing traceability for instance.

Using public messages in the construction of others has sometimes little to no cryptographic use. This is particularly the case for IDS. Since this information is public, the adversary has access to it and can reverse the operations (provided these are reversible).

Symmetry, although appealing, sometimes allows simplifications in the protocol messages and eases the task of an attacker. Notable examples include the attack of Peris-Lopez, Hernandez-Castro, Estevez-Tapiador and Van der Lubbe on Lee, Hsieh, You and Chen's protocol ([12]) and the attack on DIDRFID presented in this paper.

6 Conclusion

We have shown in this paper that Eghdamian and Samsudin's ultralightweight protocol is not secure, since a passive adversary can recover the key of a tag in an average of 20 authentication sessions. Although this number depends on L, the attack remains very efficient, even for bigger values of L than the recommended 96.

We also show key-recovery attacks on RPAP [10], PUMAP [4], DIDRFID [9] and SIDFRID [9], as well as traceability attacks on RAPP [13], and Improved LMAP+ [8].

These attacks are an additional example of the lack of security of ultralightweight protocols, and they question the relevance of this approach to design authentication protocols for RFID.

References

1. Avoine, G., Carpent, X., Martin, B.: Strong Authentication and Strong Integrity (SASI) Is Not That Strong. In: Ors Yalcin, S.B. (ed.) RFIDSec 2010. LNCS, vol. 6370, pp. 50–64. Springer, Heidelberg (2010)
2. Avoine, G., Carpent, X., Martin, B.: Privacy-friendly synchronized ultralightweight authentication protocols in the storm. Journal of Network and Computer Applications 35(2), 826–843 (2012)
3. Bárász, M., Boros, B., Ligeti, P., Lója, K., Nagy, D.: Breaking LMAP. In: Conference on RFID Security, Malaga, Spain (July 2007)
4. Bassil, R., El-Beaino, W., Itani, W., Kayssi, A., Chehab, A.: PUMAP: A PUF-based ultra-lightweight mutual-authentication RFID protocol. International Journal of RFID Security and Cryptography 1(1), 58–66 (2012)
5. Bosley, C., Haralambiev, K., Nicolosi, A.: HB^N: An HB-like protocol secure against man-in-the-middle attacks. Cryptology ePrint Archive, Report 2011/350 (2011)
6. Chien, H.-Y.: SASI: A New Ultralightweight RFID Authentication Protocol Providing Strong Authentication and Strong Integrity. IEEE Transactions on Dependable and Secure Computing 4(4), 337–340 (2007)
7. Eghdamian, A., Samsudin, A.: A Secure Protocol for Ultralightweight Radio Frequency Identification (RFID) Tags. In: Abd Manaf, A., Zeki, A., Zamani, M., Chuprat, S., El-Qawasmeh, E. (eds.) ICIEIS 2011. CCIS, vol. 251, pp. 200–213. Springer, Heidelberg (2011)
8. Gurubani, J.B., Thakkar, H., Patel, D.R.: Improvements over Extended LMAP+: RFID Authentication Protocol. In: Dimitrakos, T., Moona, R., Patel, D., McKnight, D.H. (eds.) IFIPTM 2012. IFIP AICT, vol. 374, pp. 225–231. Springer, Heidelberg (2012)
9. Lee, Y.-C.: Two ultralightweight authentication protocols for low-cost RFID tags. Applied Mathematics and Information Sciences 6(2S), 425–431 (2012)
10. Ning, H., Liu, H., Yang, C.: Ultralightweight RFID authentication protocol based on random partitions of pseudorandom identifier and pre-shared secret value. Chinese Journal of Electronics 20(4), 701–707 (2011)
11. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., Ribagorda, A.: LMAP: A Real Lightweight Mutual Authentication Protocol for Low-cost RFID tags. In: Workshop on RFID Security – RFIDSec 2006, Graz, Austria (July 2006); Ecrypt
12. Peris-Lopez, P., Hernandez-Castro, J.C., Estevez-Tapiador, J.M., van der Lubbe, J.C.A.: Security Flaws in a Recent Ultralightweight RFID Protocol. In: Workshop on RFID Security – RFIDSec Asia 2010. Cryptology and Information Security, vol. 4, pp. 83–93. IOS Press, Singapore (2010)
13. Tian, Y., Chen, G., Li, J.: A new ultralightweight RFID authentication protocol with permutation. IEEE Communications Letters 16(5), 702–705 (2012)

Improved Anonymity for Key-Trees

Thijs Veugen[1,2] and Michael Beye[1,⋆]

[1] Information Security and Privacy Lab.,
Faculty of Electrical Engineering, Mathematics and Computer Science,
Delft University of Technology, The Netherlands
M.R.T.Beye@tudelft.nl
[2] Technical Sciences, TNO, The Netherlands
thijs.veugen@tno.nl

Abstract. Randomized hash-lock protocols for Radio Frequency IDentification (RFID) tags offer forward untraceability, but incur heavy search on the server. Key trees have been proposed as a way to reduce search times, but because partial keys in such trees are shared, key compromise affects several tags. Buttyán et al. have defined measures for the resulting loss of anonymity in the system, and approximated their measures by means of simulations. We will further improve upon their trees, and provide a proof of optimality. Finally, an efficient recursive algorithm is presented to compute the anonymity measures.

Keywords: RFID, hash-lock protocol, key-tree, anonymity, anonymity set, authentication delay.

1 Introduction

We consider the problem of authenticating many Radio Frequency IDentification (RFID) tags through randomized hash-lock protocols, in an efficient way. The tags are authenticated towards the reader through a challenge-response mechanism. Each tag authenticates itself using some secret key combined with a random value (*nonce*), and to authenticate the tag, the reader will have to check the keys of all tags in order to find a match. Since this task is very intensive for the reader, an authentication tree is used. Each leaf of the tree represents a tag, and each edge corresponds to a specific key. Every tag is assigned the keys that lie on its path from the root of the tree (see Figure 1). During the authentication protocol, a tag is authenticated step by step, i.e. edge by edge, such that the computational load of the reader, and thus the total authentication time, is lowered.

However, the authentication mechanism should still remain secure. If hardware-level tampering is taken into account, keys that were assigned to compromised tags can become known to the adversary. Because partial keys are shared between neighboring tags in the tree, several additional tags may be partially

⋆ Part of this research was performed at TNO for a master's thesis for the University of Utrecht (UU). Special thanks go to Gerard Tel (UU) for his advice.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 31–47, 2013.

broken as well. How to construct the tree such that the number of (partially) broken tags will be minimal in case of one or more compromises?

This paper considers the trade-off between efficiency (minimizing authentication time), and security (minimizing the number of partially compromised tags), of such authentication mechanisms. While Buttyán, Holczer and Vajda [4] chose to keep the number of tags equal to the number of leaves in the tree, our main contribution will be to allow it to increase.

The layout of this paper is as follows: Section 2 will outline related work, with an emphasis on Buttyán et al.'s previous work on the optimization of hash-trees. In Section 3, the optimization problem is modified resulting in an improved solution, and its effect is quantified. Finally, conclusions will be drawn in Section 4. Lengthy proofs of three theorems are found in the appendices.

2 Related Work

Hash-chain protocols are meant to provide *forward untraceability*, by updating tag IDs in a one-way manner. This way, past IDs cannot be recovered, even through tampering. Examples are OSK (by Ohkubo, Suzuki and Kinoshita in [13]) and Yeo and Kim's protocol [18]. In [2], Avoine and Oechslin suggest applying time-memory trade-offs (based on Hellmann tables [7]) to hash-chain protocols (namely OSK and an improved version thereof). Hash-chain protocols have weaknesses, including *protocol exhaustion* (when the end of a chain is reached, continued updating of tag IDs will make them traceable) and *desynchronization* (server and tag chains can become out of sync if tags are queried by third parties).

A different class of hash-based authentication schemes called *Hash-lock protocols* (due to Weis et al.) was devised to solve the aforementioned problems. Tags are locked and unlocked, using hashes of their ID as the key. The *static hash-lock scheme* [17] is vulnerable to both *replay attacks* and *tracking*, but in the same paper, Weis, Sarma, Rivest and Engels offer the *randomized hash-lock scheme* as a solution to such attacks: it adds *tag freshness* (a *nonce* generated by the tag) to prevent reader impersonation and tracking. The nonce is used as a challenge, and is hashed together with the tag's ID to form a one-time-use authentication key (the expected response). Juels and Weis [8] later added reader freshness to also prevent tag impersonation.

Note that precomputation cannot be used in these protocols, because the use of freshness makes the search space too large – one would need to compute values not only for each tag, but for each tag ID in combination with all possible nonces. Other solutions are required to reduce search complexity.

Molnar and Wagner were the first to propose using a *tree of secrets* for RFID tags [9]. Although originally used for a system built around exclusive-OR and a pseudo-random function, it can be applied to other challenge-response building blocks. Damgård and Østergaard Pedersen [5] use the same concept, but speak of *correlated keys*. Nohara, Nakamura, Baba, Inoue and Yasuura in their "K-steps protocol" ([10], also dubbed NIBY) propose to apply trees to the hash-lock setting. They use the term *group IDs* rather than correlated keys, and

their trees are unconventional (being of non-uniform depth). Note that all these approaches use a sequence of group- and sub-group IDs to quickly and gradually narrow down a tag's identity. As Molnar and Wagner mention, *partial keys in such a tree should be chosen independently and uniformly from a key space of sufficient entropy.* Failure to do so would make the system vulnerable to attack. If partial keys are chosen properly, the adversary will have a large key space to search, while the owner of the system can efficiently search through a limited subspace (the actual tree).

The trade-off that exists between efficiency and security in tree-based protocols was already pointed out by Avoine and Oechslin [2], with respect to Molnar's original trees. Because tags share their partial keys, if one tag is compromised (i.e. has its memory probed through invasive tampering), an adversary learns partial keys for several other tags as well. This will enable him to decipher their responses in some of the verification steps, resulting in reduced anonymity and facilitating tracking.

A paper of particular interest is by Buttyán, Holczer and Vajda [4], where the concept of trees with *variable branching factors* is introduced, to better preserve anonymity in case of attack. Our work provides an optimization of Buttyán's solution, allowing the number of leaves in the tree to increase beyond the number of tags.

2.1 Adaptive Adversaries and Metric

Altough this work is dedicated to static adversaries that choose compromised tags in a random way, some interesting relations can be found with other papers on adaptive adversaries that selectively choose compromised tags possibly based on some extra (side-channel) knowledge about the tags.

As in [4], we use the average anonymity set size as a metric for the level of privacy. In this metric each (subsequent) tag is considered equally likely to be compromised and therefore suits the static adversary model. Because in the adaptive adversary model different (groups of) tags could be distinghuished, Nohl and Evans [11] propose to measure information leakage in bits (or nats) which allows quantifying the potential gain of an adversary.

In succeeding work Nohl and Evans [12] investigate the trade-off between level of privacy and the cost of protection suggesting an optimal tree of depth two. A similar tree was found in [1] by Avoine, Buttyán, Holczer and Vajda who try to further improve the balance between complexity and privacy in a new authentication protocol. In short, the tags are divided into λ groups, where each group shares a group-key. Every tag also has an ID. This group-based scheme can be seen as a tree of depth 2, where every group-ID is tried, but the last stage (unique ID) only requires one decryption instead of exhaustive search. This means that the tree can be even wider at the top than a Buttyán tree, and thus attains a higher anonymity score.

However, we choose not to follow this example because we believe that the *group-based authentication protocol* in [1] has inherent flaws. Its suspected weakness lies in the fact that the final stage of narrowing down IDs is essentially

skipped (the unique ID can be simply decrypted and read). If an attacker can choose his tags with some confidence, he can very quickly remove all anonymity within the system by choosing one tag from each group. Tree-based systems still preserve some measure of anonymity in these cases.

Recently, Beye and Veugen [3] analysed the case of adaptive adversaries in trees with variable branching factors. They suggest a so called Hourglass tree that provides both efficient authentication and privacy protection against intense targeted attacks. A similar approach could be used to extend our results from static to adaptive adversaries.

2.2 Notation

In this paper we use the following notation, thereby generalizing Buttyán's notation in [4]:

- $T = \{t_1, \cdots, t_N\}$: set of all tags in the system
- N: size of T, or actual number of tags in the system
- $B = (b_1, \ldots, b_d)$: a "branching factor vector" (or tuple), representing a tree of depth d
- $\sum(B)$: shorthand for $\sum_{i=1}^{d} b_i$, or the sum over all elements in B
- $\prod(B)$: shorthand for $\prod_{i=1}^{d} b_i$, or the product over all elements in B
- N': number of leaves in the tree ($\prod(B)$), or maximum number of tags in the system, $N' \geq N$
- c: number of compromised tags
- $P(t_i)$: helper function that returns the anonymity set to which tag t_i belongs (see Definition 1)
- P_j: anonymity set j, $1 \leq j \leq \ell$
- $\bar{S}$: average size over all anonymity sets in a given configuration
- $\bar{S}_c(B)$: expected value of $\bar{S}$, averaged over all configurations containing c compromised tags in the tree with branching factor vector B (see Definition 2)
- $R(B)$: resistance to single member compromise for a tree with branching factor vector B
- $R_c(B)$: resistance to c member compromise for a tree with branching factor vector B, $R_c(B) = \bar{S}_c(B)/N'$

2.3 Buttyán Trees

Buttyán et al. [4] observed the time-anonymity trade off and noted that *narrow, deep trees allow faster search; it is wide, shallow trees that provide more anonymity.* Clearly, if many tags share the same partial keys, many tags can be excluded from the search space after each authentication stage, thus making search faster. The increased anonymity can be intuitively explained by the fact that when partial keys are shared between fewer tags, the amount of information gained by compromising a single tag is limited. Buttyán uses the concept of *anonymity sets* (Pfitzmann and Köhntopp [14], Samarati and Sweeney [15], Díaz [6]) to quantify matters.

Definition 1. *Assume a tag t_i sends a given message m (or participates in a protocol execution). For an observer O, the anonymity set $P(t_i)$ contains all tags that O considers possible originators of m. Because all tags in $P(t_i)$ are indistinguishable to O, t_i is anonymous among the other tags in the set.*

Anonymity sets provide a sliding scale for anonymity, where belonging to a larger set implies a greater degree of anonymity. Total anonymity holds if the set encompasses all possible originators in the whole system (one is indistinguishable among all N tags in T), and belonging to a singleton set implies a complete lack of anonymity.

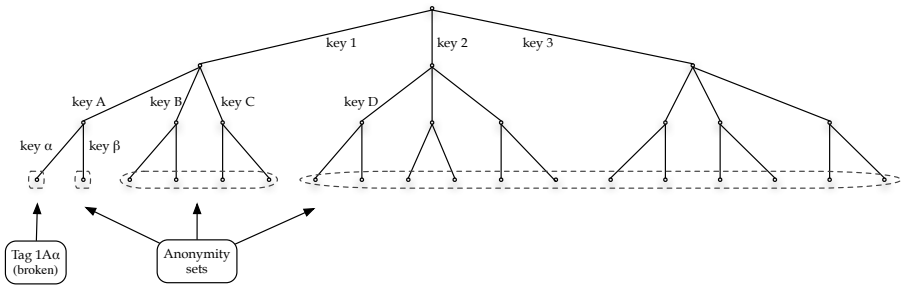


Fig. 1. Hash tree with a single broken tag [4]

To measure the level of anonymity offered by a tree, Buttyán looks at the level of anonymity provided for a randomly selected member. This *expected size of the anonymity set that a randomly selected member will belong to*, is denoted $\bar{S}$. One could also view it as *the average anonymity set size over all tags*, as shown in Equation 1 [4]. Note that $\bar{S}$ can be computed for any given scenario where a tree is broken into anonymity sets. Note that, for $c > 1$, the sizes of anonymity sets within the tree can vary, as different configurations of broken tags are formed. Configurations containing the same (number and size of) anonymity sets are considered identical, because sets can always be ordered in ascending order without loss of generality.

$$\bar{S} = \sum_{i=1}^{N} \frac{|P(t_i)|}{N} = \sum_{j=1}^{\ell} \frac{|P_j|}{N} |P_j| = \sum_{j=1}^{\ell} \frac{|P_j|^2}{N} \ , \tag{1}$$

where $P(t_i)$ is a function that returns the anonymity set to which tag t_i belongs, P_j denotes an anonymity set and ℓ is the number of sets.

Buttyán then defines R, the *resistance to single member compromise*, as $\bar{S}$ computed for a scenario where *a single tag* is broken, and then normalizing the result (as in Samarati and Sweeney [15] generalized by Díaz [6]). Note that because we can freely order the anonymity sets, $c = 1$ leads to a single unique configuration. With its range of $[0, 1]$, $R = \frac{\bar{S}}{N}$ is independent of N, allowing

for easy comparison between systems of different sizes. In the scenario of single member compromise as depicted in Figure 1, the number of anonymity sets is equal to $d + 1$.

We will refer to trees with a constant branching factor as "*Classic trees*". Buttyán proposes the use of trees with different, independent branching factors on each level, sorted in descending order as shown in Figure 1. Trees will be described by their branching factor vectors $B = (b_1, \ldots, b_d)$, where the variables b_i $(1 \leq i \leq d)$ are integers larger than 1 denoting the branching factor at level i.

Buttyán et al. in [4] reach the conclusion that the branching factors near the root contribute more to $\bar{S}$ and R. For trees with variable branching factors this means that a deep, top heavy tree can potentially outperform a shallow classic tree.

They present a *greedy* algorithm that recursively finds the branching factor vector B that maximizes R, given a number N of tags and a maximum authentication delay D_{max}. It starts with the prime factorization of N and tries to combine prime factors as long as the sum $\sum(B)$ (authentication delay) remains acceptable. An important assumption is that the number of leaves in the tree is equal to the number of tags, i.e. $\prod(B) = N$.

However, Buttyán recognizes that trees need to stand up to more than single tag compromise. We suggest to express $\bar{S}$ for the general case as follows:

Definition 2. *$\bar{S}_c(B)$ expresses $\bar{S}$ as the average over all $\binom{N}{c}$ possible distributions of c compromised members across the tag set T which consists of the $N = \prod(B)$ leaves of the tree represented by branching factor vector B.*

Our notation is a natural extension of Buttyán's $\bar{S}_{\langle - \rangle}$, directly incorporating B and c. Depending on how each successive member is picked from the tree, different anonymity sets are broken down.

3 Improved Hash-Trees

Our main observation is that Buttyáns condition $\prod(B) = N$ can lead to inferior solutions. Particularly when the number N has large prime factors, resulting in a small number of candidate branching factor vectors. We prefer the condition $\prod(B) \geq N$, which we will show leads to better results. An added advantage in practice is that it allows to maintain a small buffer of extra keys (see discussion in Section 3.1). Our optimization problem now becomes:

Problem 1. Given the total number N of members and the upper bound D_{max} on the maximum authentication delay, find the vector $B = (b_1, \ldots, b_d)$ that maximizes $R(B)$ subject to the following constraints:

$$\prod(B) = \prod_{i=1}^{d} b_i \geq N, \text{ and } \sum(B) = \sum_{i=1}^{d} b_i \leq D_{max} \ . \tag{2}$$

The anonymity measure $R(B)$ used here refers to the full tree with $\prod(B) = N'$ tags, of which exactly one is compromised, i.e. $c = 1$. Theorem 3 will later show that the same holds for the anonymity measure of the partial tree with $N \leq N'$ tags.

Theorem 1. *The maximal $R(B)$ under the constraints $\prod(B) \geq N$ and $\sum(B) \leq D_{max}$ is achieved by the lexicographically largest vector B that satisfies these constraints.*

The proof of Theorem 1 is given in the Appendix. The following theorem, whose proof is in the appendix, shows how to optimize the product of a branching vector, while keeping the sum constant and ignoring the lexicographic order. The notation (3^*) is used to denote a (possibly empty) branching factor vector of arbitrary dimension consisting solely of factors 3.

Theorem 2. *Let $D \geq 2$ be a fixed integer and let $\prod_D^{max}$ be the largest product $\prod(B)$ attained by branching factor vectors B with sum $\sum(B) = D$. Then this maximal product is attained by branching factor vectors B with $\sum(B) = D$ that have one of the following shapes: (3^*), $(4, 3^*)$ or $(3^*, 2)$.*

So when searching for the vector B that optimizes $\prod(B)$, it is sufficient to search within the limited set of vectors that have one of the above described shapes. In fact, the value D mod 3 directly determines which of the three shapes should be chosen (see Appendix B).

When considering Problem 1, we know that when $D = D_{max}$ and $\prod_D^{max} < N$, there can be no solution that satisfies both constraints. On the other hand, when $\prod_D^{max} \geq N$, there is at least one solution. The obvious way to find the branching factors of the lexicographically largest solution, is to take a *greedy* approach. It means that the first branching factor is optimized first, then the second, etc. The algorithm depicted in Figure 2, which is denoted further on by Algorithm 2, takes N and D_{max} as input and solves this problem recursively [16]. A specific branching factor is allowed, when a suitable tail (according to Theorem 2) with a large enough product exists.

3.1 Consequences of Larger Trees

Algorithm 2 can lead to trees that exceed the strictly required number of leaves (with $N' > N$). We argue that this has practical advantages, but should also be taken into account when judging the anonymity of such trees.

A larger tree will allow for addition of tags at a later time, which may be desirable in practice. Ideally, creating and balancing a tree should be done only once, and therefore the tree should accommodate all the tags *ever expected to enter the system*. In systems where growth is anticipated, having a larger tree that is ready for the future is good practice.

Also, since we are defending against tampering attacks, replacement of compromised tags should be taken into consideration. Replacement tags should contain *new key material*, lest they be reintroduced with keys that are already fully

```
function B = VB_f(N, d_{MAX}) % VB = Veugen-Beye

Precondition: d_{MAX} > 1
Postcondition:
  B is the lexicographically largest vector satisfying
  prod(B) >= N and sum(B) <= d_{MAX}


B := [d_{MAX}]; % Start with a tree of depth one

while (prod(B) < N) and (b_1 > 2)
    b_1 := b_1 - 1; % next candidate for b1
    prod(B) := b_1 * prod^{MAX}_{d_{MAX} - b_1};
    % maximal product given first factor b_1
end;

if
    prod(B) < N          ->  "No solution exists.";
    d_{MAX} - b_1 <= 1   ->  B := b_1; % no tail left;
    else                 ->  B := [b_1 VB_f(N/b_1, d_{MAX} - b_1)];
                             % find next branching factor
end;
```

Fig. 2. Recursive function for finding an optimal solution B of Problem 1

disclosed (immediately limiting their anonymity). Having unused leaves in the tree seems ideal for this purpose.

When choosing *which* leaves to actually use as tags (initially and for replacements), we suggest to select a sufficient number of branches at the level $d-1$ *at random*, and to randomly initialize tags from these branches. This to create a subtree of initialized tags that is as close to the original (optimal) shape as possible, without introducing order in the system which might be exploited.

Finally note that tags corresponding to uninitialized leaves in the tree cannot be encountered by adversaries in the field. For this reason, they do not contribute to the size of the set among which targets need to be distinguished. However, given that the resistance is actually the average anonymity set size normalised per tag, it should intuitively remain roughly equal. This is formally proven in the following theorem.

Theorem 3. *If N tags are placed uniformly at random in a tree with $\prod(B) = N' > N$, then the expected resistance R_c to c member compromise satisfies*

$$R_c(B) < R_c < R_c(B) + \frac{N' - N}{N^2}$$

Because of the result of Theorem 3, whose proof is in the appendix, it makes sense to estimate the resistance R_c by the full tree resistance $R_c(B)$. This contradicts with previous work of Beye-Veugen [3] who gratuitously used a scaling factor

$\frac{N}{N'}$ to adjust their anonymity measures. There is a flaw in the proof of their Theorem 4 where they pose that $E[\frac{1}{N}\sum_{i=1}^{N}\frac{|P(t_i)|}{N}] = E[\frac{1}{N}\sum_{i=1}^{N'}\frac{|P(t_i)|}{N}]$ which explains the erroneous appearance of the factor $\frac{N}{N'}$. However, with respect to their full paper it is only a minor flaw and doesn't affect their main conclusions.

3.2 Comparison of Performance

Since our search space is larger than Buttyán's, our trees potentially perform better in two ways:

1. Given the same maximal delay D_{max}, we might find a lexicographically larger tree that provides better anonymity (increase in R).
2. Given the same (or at least not worse) resistance to compromise R, we might find a tree with lower $\sum(B)$ thus decreasing the authentication delay.

The results of both approaches are depicted in Table 1 where for three different categories 1000 random instances (N, D_{max}) have been generated and the results have been averaged. The intervals for the parameters N and D_{max} have been chosen such that their sizes resemble those of Buttyán.

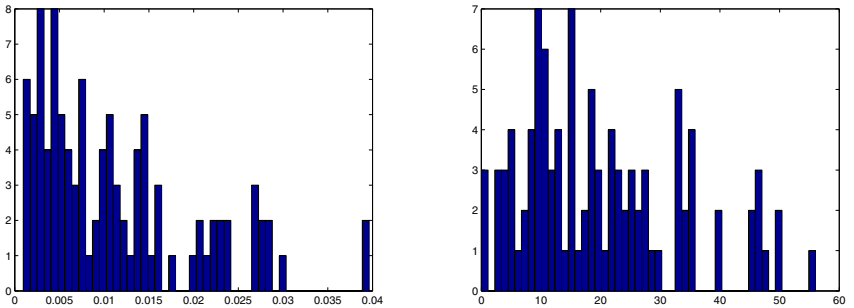
Table 1. Increase of performance given 1000 random instances

N		D_{max}		# of solvable instances		Average	Average
min	max	min	max	Buttyán	Our work	increase in R	decrease in D
1000	10000	40	120	221	1000	0.0126	10.6290
10000	100000	50	150	101	1000	0.0112	20.0099
100000	1000000	60	180	46	1000	0.0160	27.1087

Remarkably, a huge number of instances turn out to be unsolvable within Buttyán's optimization problem. Analysis learns that this is due to the large prime factors of these values of N which raise the delay to an unacceptable level. Indeed, the minimally achievable delay in Buttyán's setting equals the sum of all prime factors of N. As argued in Theorem 1, our minimally achievable delay is roughly $3\log_3 N$ (when all branching factors are three, see Theorem 2) which explains that all instances are solvable within Problem 1.

To obtain better insight in our actual improvements, the performance of the 101 solvable instances with $10^4 \leq N \leq 10^5$ and $50 \leq D_{max} \leq 150$ is analyzed in more detail by two histograms showing the distribution of the increase in R (Figure 3(a)) and the decrease in D (Figure 3(b)) respectively over 50 equally sized bins. Figure 3(a) shows e.g. that we were able to increase the resistance of compromise of 2 instances by a value between 0.0392 and 0.04.

The achievable increment in R may seem modest but is comparable with Buttyán's improvement with respect to the Classic tree. The advantage will be more significant for larger values of c as shown in Figure 4(b). The attainable slump in authentication delay by our new trees can be considered substantial. So besides from the fact that many instances are unsolvable in Buttyán's setting, our trees outperform Buttyán's trees on both higher R and lower D_{max}.



(a) Resistance to compromise (b) Delay

Fig. 3. Histograms of improved performance over 101 random instances

3.3 Multiple Compromised Tags

Subsection 3.2 has already shown that our proposal yields lexicographically larger B's than Buttyán's approach, and consequently better anonymity measures when $c = 1$. The computation of resistance to compromise $R_c(B)$ becomes more difficult for $c > 1$. Buttyán noted that computing $\bar{S}_c(B)$ is hard, and therefore suggested an alternative measure $\bar{S}_0$ corresponding with an even distribution of c compromised tags across T which he used to approximate $\bar{S}_c(B)$.

Proposition 1. *Although not stated explicitly in [4], $\bar{S}_0$ actually represents the worst-case choice of c compromised tags across T resulting in the minimal value of $\bar{S}$.*

Proof. Assume that we are allowed to choose tags to be compromised sequentially, with the aim to minimize the average anonymity set size. The first compromised tag leads to a unique configuration (as described further on). Each subsequent compromised tag leads to a new configuration, with more anonymity sets (of varying, decreasing size). To minimize the average set size in the *resulting* configuration, the next tag to be compromised should be chosen from (one of) the largest anonymity set(s) in the *current* configuration. When sorting anonymity sets in ascending order, we observe that this is equivalent to choosing tags (as) evenly (as possible given the tree structure) across T. By induction, our claim holds for any c. □

Buttyán[4], and Beye and Veugen [3] used simulations to approximate $\bar{S}_c(B)$, but we present an efficient algorithm for recursively computing the exact resistance and compare our approach to Classic and Buttyán trees by means of numerical computations.

Let $U_c(B) = \sum_{i=1}^{N} |P(t_i)|$ be the anonymity set size added over all tags, after c particular tags from the tree with branching factor vector B have been compromised. We would like to compute

$$\bar{S}_c(B) = \frac{\bar{U}_c(B)}{\prod(B)}$$

where $U_c(B)$ is averaged over all possible choices of c out of $\prod(B)$ tags. Note that resistance $R_c(B)$ to c-member compromise equals $\bar{S}_c(B)/\prod(B)$.

When one tag of the $N = \prod(B)$ tags of the tree with branching factor vector $B = (b_1, b_2, \dots b_d)$ is compromised, the tree falls into $d+1$ anonymity sets (see [4] and Figure 1). The first anonymity set S_0 consists of the compromised tag, the other d sets S_j, $1 \le j \le d$, correspond to the subtrees with branching factor vector $(b_j - 1, b_{j+1}, \dots, b_d)$ and therefore have size $|S_j| = (b_j - 1)b_{j+1} \dots b_d$. This leads to the following recursive relation for computing $\bar{U}_c(B)$.

$$\bar{U}_c(b_1, b_2, \dots b_d) = \begin{cases} \prod_{i=1}^{d} b_i^2 & \text{if } c = 0 \\ 1 + \bar{U}_{c-1}(b_d - 1) & \text{if } c > 0 \text{ and } d = 1 \\ 1 + \sum_{i=0}^{c-1} \sum_{j=1}^{d} f_i^j \cdot \bar{U}_i(b_j - 1, b_{j+1}, \dots, b_d) & \text{if } c > 0 \text{ and } d > 1 \end{cases}$$

where the frequencies f_i^j are readily computed for $0 \le i < c$, $1 \le j \le d$ by binomial coefficients:

$$f_i^j = \frac{\binom{|S_j|}{i}\binom{N-1-|S_j|}{c-1-i}}{\binom{N-1}{c-1}}$$

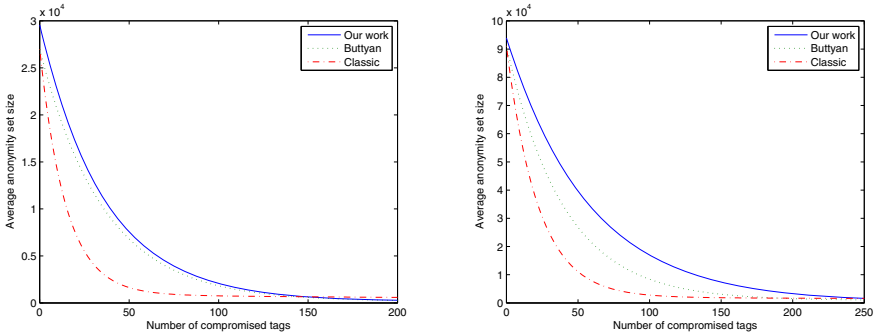
and which represent the relative number of ways to choose i tags from anonymity set S_j and the remaining $c-1-i$ tags from the other anonymity sets. Note that $f_i^j = 0$ whenever $i > |S_j|$ or $c - 1 - i > N - 1 - |S_j|$.

We wrote a recursive MATLAB function AS_f to recursively compute $[\bar{U}_0(B), \dots, \bar{U}_c(B)] = AS_f(B, c)$ which is available at our site [16].

While similar figures arise for larger values of N we compute, as in [4], $\bar{S}_c(B)$ for the configuration with $N = 30^3 = 27000$ and $D_{max} = 3 \cdot 30 = 90$ to make a fair comparison. The optimal tree computed by Buttyán is $(72, 5, 5, 5, 3)$, slightly improved by our Algorithm 2 to $(73, 5, 3, 3, 3, 3)$. Figure 4(a) compares their $\bar{S}_c(B)$ with the classic tree $B = (30, 30, 30)$ that has constant branching factors.

Buttyáns optimal tree for the second configuration $(N, D_{max}) = (45^3, 3 \cdot 45) = (91125, 135)$ is $(81, 25, 15, 3)$, which is further increased by Algorithm 2 to $(116, 5, 3, 3, 3, 3, 2)$. In Figure 4(b) their $\bar{S}_c(B)$ is compared with the classic tree $B = (45, 45, 45)$. We will discuss how these results relate to our hypotheses and claims.

In both figures, our tree outperforms, as expected, both the Buttyán and the classic tree in terms of $\bar{S}_c(B)$. We observe that the performance of our tree in no case drops below that of the Buttyán tree. The difference between both configurations is explained by the prime factorizations of $30 = 5 \cdot 3 \cdot 2$ and $45 = 5 \cdot 3 \cdot 3$ which gives a little more playground to Buttyán in the first configuration. In the second configuration, the gain of Buttyán's tree with respect to the Classic tree is comparable to our gain with respect to Buttyán's tree. Given that our tree has a 0.0073 higher resistance to single member compromise than Buttyán's tree, the improvement in R of the second configuration is even less



(a) $N = 30^3$ and $D_{max} = 90$ (b) $N = 45^3$ and $D_{max} = 135$

Fig. 4. Comparison of $\bar{S}_c(B)$ for two configurations

than expected by a random instance as shown in Table 1. The reason is that both N's have more small prime factors than expected on average.

In each Figure we observe a turning point where the classic tree starts to outperform the other two trees. This occurs at $c \approx 2b_1$. At this point, the decrease of $\bar{S}$ slows causing the graph to *seemingly* settle into a steady minimum. We can explain this by the fact that at around this point, the last very large anonymity set is expected to have been broken down, because each top-level branch can be expected to contain at least one compromised tag. Because subsequent compromised tags then fall into smaller sets, the adversary will learn little new information; he has obtained the most important keys in the tree already. In such a worrying scenario, what little amount of anonymity tags have left depends upon the keys in lower branches. Classic trees retain slightly more anonymity, because they have larger branching factors at the bottom levels. However, given the (by then) minimal values of $\bar{S}$ overall, the *absolute* advantage is not large.

4 Conclusions and Future Work

Our proposed Algorithm 2 yields better results than Buttyán's original approach, when it comes to finding the lexicographically largest B. We have provided proof that the solution is optimal in terms of optimization problem 1. The problem that many instances are unsolvable within Buttyán's optimization setting has been solved by our modification. The solvable instances can be further optimized by our algorithm to either increase the resistance to compromise as expressed by $R_c(B)$, or to lower the authentication delay. Algorithm 2 can result in trees with $N' \geq N$, which can be advantageous in growing systems or when replacing compromised tags, and whose resistance to compromise has been proven commensurably.

For future research it might be interesting to precisely investigate to what extend $R_c(B)$ increases by lexicographically larger B for $c > 1$. Our recursive

formula for computing $R_c(B)$ opens up this possibility. This could even be extended to adaptive adversary scenarios as described in [3].

References

1. Avoine, G., Buttyán, L., Holczer, T., Vajda, I.: Group-based private authentication. In: IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, pp. 1–6 (2007)
2. Avoine, G., Oechslin, P.: A Scalable and Provably Secure Hash Based RFID Protocol. In: International Workshop on Pervasive Computing and Communication Security – PerSec 2005, pp. 110–114. IEEE, IEEE Computer Society, Kauai Island (2005)
3. Beye, M., Veugen, T.: Anonymity for Key-trees with Adaptive Adversaries. In: Rajarajan, M., Piper, F., Wang, H., Kesidis, G. (eds.) SecureComm 2011. LNICST, vol. 96, Springer, Heidelberg (2012)
4. Buttyán, L., Holczer, T., Vajda, I.: Optimal Key-Trees for Tree-Based Private Authentication. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 332–350. Springer, Heidelberg (2006)
5. Damgård, I., Pedersen, M.Ø.: RFID Security: Tradeoffs between Security and Efficiency. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 318–332. Springer, Heidelberg (2008)
6. Díaz, C.: Anonymity Metrics Revisited. In: Dolev, S., Ostrovsky, R., Pfitzmann, A. (eds.) Anonymous Communication and its Applications. Dagstuhl Seminar Proceedings, vol. 05411. Internationales Begegnungs-und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany (2006)
7. Hellman, M.: A cryptanalytic time-memory trade-off. IEEE Transactions on Information Theory 26, 401–406 (1980)
8. Juels, A., Weis, S.A.: Defining Strong Privacy for RFID. In: PERCOMW 2007: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshops, pp. 342–347. IEEE Computer Society, Washington, DC (2007)
9. Molnar, D., Wagner, D.: Privacy and security in library RFID: issues, practices, and architectures. In: CCS 2004: Proceedings of the 11th ACM Conference on Computer and Communications Security, pp. 210–219. ACM, New York (2004)
10. Nohara, Y., Nakamura, T., Baba, K., Inoue, S., Yasuura, H.: Unlinkable identification for large-scale rfid systems. Information and Media Technologies 1(2), 1182–1190 (2006)
11. Nohl, K., Evans, D.: Quantifying Information Leakage in Tree-Based Hash Protocols (Short Paper). In: Ning, P., Qing, S., Li, N. (eds.) ICICS 2006. LNCS, vol. 4307, pp. 228–237. Springer, Heidelberg (2006)
12. Nohl, K., Evans, D.: Hiding in groups: On the expressiveness of privacy distributions. In: 23rd International Information Security Conference (SEC 2008), Milan (September 2008)
13. Ohkubo, M., Suzuki, K., Kinoshita, S.: Cryptographic Approach to "Privacy-Friendly" Tags. In: RFID Privacy Workshop. MIT, MA (2003)
14. Pfitzmann, A., Köhntopp, M.: Anonymity, Unobservability, and Pseudonymity - A Proposal for Terminology. In: Federrath, H. (ed.) Anonymity 2000. LNCS, vol. 2009, pp. 1–9. Springer, Heidelberg (2001)

15. Samarati, P., Sweeney, L.: Generalizing data to provide anonymity when disclosing information. In: Proceedings of the Seventeenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems (PODS), Seattle, WA, USA, p. 188 (1998)
16. Veugen, T., Beye, M.: Matlab code for "improved anonimity of key-trees". In: RFIDsec (2012), http://isplab.tudelft.nl/content/improved-anonimity-key-trees
17. Weis, S.A., Sarma, S.E., Rivest, R.L., Engels, D.W.: Security and Privacy Aspects of Low-Cost Radio Frequency Identification Systems. In: Hutter, D., Müller, G., Stephan, W., Ullmann, M. (eds.) Security in Pervasive Computing 2003. LNCS, vol. 2802, pp. 201–212. Springer, Heidelberg (2004)
18. Yeo, S.-S., Kim, S.K.: Scalable and Flexible Privacy Protection Scheme for RFID Systems. In: Molva, R., Tsudik, G., Westhoff, D. (eds.) ESAS 2005. LNCS, vol. 3813, pp. 153–163. Springer, Heidelberg (2005)

A Proof of Theorem 1

In this appendix we proof Theorem 1. By $B\backslash\{b_1, \cdots, b_j\}$, we denote the vector $(b_{j+1}, \ldots, b_d)$, where $1 \leq j \leq d$.

The first observation is that for an optimal B, $\sum(B) = D_{max}$, otherwise $D_{max} - \sum(B)$ could be added to any element of B without violating the constraints while increasing $R(B)$. So we assume $\sum(B) = D_{max}$ in the proof, which uses four Lemmas, similar to the Lemmas of Buttyán's work [4]. It's also clear that an optimal B will have branching factors at least 2. The first Lemma, Lemma 1, shows that a branching vector can always be improved by ordering its elements in decreasing order. Lemma 3, using some bounds from Lemma 2, shows that given two branching factor vectors, the one with the larger first element is always at least as good as the other. Lemma 4 generalizes Lemma 3 by stating that given two branching factor vectors the first j elements of which are equal, the vector with the larger $(j+1)$-st element is always at least as good as the other.

These Lemma's together show that a lexicographically larger branching factor vector will always be at least as good as the lexicographically smaller branching factor vector (in case $\sum(B) = D_{max}$), so indeed the solution with maximal $R(B)$ to Problem 1 is achieved by the lexicographically largest vector that satisfies the constraints.

Lemma 1. *Let B be a branching factor vector, and let B^* be the vector that consists of the sorted permutation of the elements of B in decreasing order. If B satisfies the constraints of Problem 1, then B^* satisfies them too, and $R(B^*) \geq R(B)$.*

Proof. Since $\prod(B)$ is not altered by the permutation, we can refer to Buttyán's proof [4] of Lemma 1. □

Lemma 2. *Let $B = (b_1, \ldots, b_d)$ be a sorted branching vector (i.e. $b_1 \geq b_2 \geq \ldots \geq b_d$). We can give the following lower and upper bounds on $R(B)$:*

$$\left(1 - \frac{1}{b_1}\right)^2 \leq R(B) \leq R(b_1) = \frac{1 + (b_1 - 1)^2}{b_1^2}$$

Proof. The lower bound is identical to Buttyán, hence the proof [4] is as well. The upper bound is an improvement w.r.t. Buttyán, and is proven as follows. Let $M = \prod(B)$, then $\prod(B \backslash b_d) = M/b_d$. We derive for $d > 1$:

$$\begin{aligned}
R(B) &= \frac{1}{M^2}\left(1 + (b_d - 1)^2 + \sum_{i=1}^{d-1}(b_i - 1)^2 \prod_{j=i+1}^{d} b_j^2\right) \\
&= \frac{1}{M^2}\left(1 + (b_d - 1)^2 + \sum_{i=1}^{d-2}(b_i - 1)^2 \prod_{j=i+1}^{d} b_j^2 + (b_{d-1} - 1)^2 b_d^2\right) \\
&= R(B\backslash b_d) - \frac{b_d^2}{M^2}\left(1 + (b_{d-1} - 1)^2\right) + \frac{1}{M^2}\left(1 + (b_d - 1)^2 + (b_{d-1} - 1)^2 b_d^2\right) \\
&= R(B\backslash b_d) + \frac{2 - 2b_d}{M^2} \\
&< R(B\backslash b_d)
\end{aligned}$$

and by recursively applying this inequality also $R(B) \leq R(b_1)$. □

Lemma 3. *Let $B = (b_1, \ldots, b_d)$ and $B' = (b'_1, \ldots, b'_{d'})$ be two sorted branching factor vectors (i.e. $b_1 \geq b_2 \geq \ldots \geq b_d$, $b'_1 \geq b'_2 \geq \ldots \geq b'_{d'}$) that satisfy the constraints of Problem 1. Then, $b_1 > b'_1$ implies $R(B) \geq R(B')$.*

Proof. We first prove the statement for $b'_1 \geq 3$. From Lemma 2 we know that

$$R(B') \leq \frac{1 + (b'_1 - 1)^2}{{b'_1}^2}$$

and

$$R(B) \geq \left(1 - \frac{1}{b_1}\right)^2 > \left(1 - \frac{1}{b'_1 + 1}\right)^2$$

which follows from the fact that $b_1 > b'_1$. A straightforward calculation shows that $(1 - \frac{1}{b'_1+1})^2 \geq \frac{1+(b'_1-1)^2}{{b'_1}^2}$ whenever $b'_1 \geq 3$, and thus $R(B) \geq R(B')$.

So the remaining case is $b'_1 = 2$. Since B' is ordered, each element of B' will equal 2. If $d' = 1$ then by our previous assumption $D_{max} = \sum(B') = 2$, but this contradicts $D_{max} = \sum(B) \geq 3$, so we know $d' \geq 2$. The resistance $R(B')$ is readily computed as $R(B') = \frac{1}{3}(2 \cdot 4^{-d} + 1)$, which will be at most $\frac{3}{8}$ (when $d' = 2$). Since $R(B) \geq (1 - \frac{1}{b_1})^2 > (1 - \frac{1}{3})^2 = \frac{4}{9}$, it follows that also in this case $R(B) \geq R(B')$. □

Lemma 4. *Let $B = (b_1, \ldots, b_d)$ and $B' = (b'_1, \ldots, b'_{d'})$ be two sorted branching factor vectors (i.e. $b_1 \geq b_2 \geq \ldots \geq b_d$, $b'_1 \geq b'_2 \geq \ldots \geq b'_{d'}$) that satisfy the constraints of Problem 1. Let j, $1 \leq j < \min(d, d')$, be such that $b_i = b'_i$ for all i, $1 \leq i \leq j$, and $b_{j+1} > b'_{j+1}$, then $R(B) \geq R(B')$.*

Proof. It is easy to show that $R(B) = \left(\frac{b_1 - 1}{b_1}\right)^2 + \frac{1}{b_1^2} \cdot R(B \backslash b_1)$. Therefore, since $b_1 = b'_1$, $R(B) \geq R(B')$ whenever $R(B \backslash b_1) \geq R(B' \backslash b'_1)$. By recursively applying this rule, and using Lemma 3, which shows that $R(B \backslash \{b_1, \ldots, b_j\}) \geq R(B' \backslash \{b'_1, \ldots, b'_j\})$, the proof is complete. The proof of Lemma 4 is similar to the proof of Buttyán's Lemma 4 [4]. □

B Proof of Theorem 2

This appendix contains the proof of Theorem 2.

Proof. Let B be a branching factor vector with $\sum(B) = D$. The proof is given by considering different cases.

Suppose B has a branching factor b_i equal to 1. Since $\sum(B) \geq 2$, there must be another branching factor b_j. Then, we could add b_i to b_j to increase $\prod(B)$ without modifying $\sum(B)$, meaning $\prod(B) \neq \prod_D^{max}$. Therefore, an optimal B (with $\prod_D^{max}$) contains no branching factor equal to 1.

Suppose B has a branching factor $b_i \geq 5$. Since $(b_i - 3) \cdot 3 > b_i$, we can increase $\prod(B)$ without modifying $\sum(B)$, by making an extra factor 3, meaning $\prod(B) \neq \prod_D^{max}$. Therefore, an optimal B contains only branching factors 2, 3 or 4.

Suppose B has two branching factors $b_i = b_j = 4$ $(i \neq j)$. Since $3 \cdot 3 \cdot 2 = 18 > 16 = 4 \cdot 4$, we can increase $\prod(B)$ without modifying $\sum(B)$ by changing b_i and b_j to 3 and adding an extra 2, meaning $\prod(B) \neq \prod_D^{max}$. Therefore, the optimal B contains at most one branching factor 4.

Suppose B has two branching factors $b_i = b_j = 2$ $(i \neq j)$. Since $2 \cdot 2 = 4$, we could just as well substitute these branching factors by a single 4, making B lexicographically larger. Therefore, $\prod_D^{max}$ can be attained by at most one branching factor 2.

Suppose B has two branching factors $b_i = 2$ and $b_j = 4$. Since $2 \cdot 4 = 8 < 9 = 3 \cdot 3$, we can increase $\prod(B)$ without modifying $\sum B$ by substituting both factors by 3, meaning $\prod(B) \neq \prod_D^{max}$. Therefore, an optimal B will not contain both branching factors 2 and 4.

By considering these five cases, it follows that $\prod_D^{max}$ will be attained in one of the following cases:

1. B contains only 3's;
2. B contains one 4 and an arbitrary number of 3's;
3. B contains one 2 and an arbitrary number of 3's.

Consequently when $\sum(B) = D$, and we order the elements descendingly, $\prod_D^{max}$ will be attained by:

1. $B = (3^*)$, when $D \bmod 3 = 0$;
2. $B = (4, 3^*)$, when $D \bmod 3 = 1$;
3. $B = (3^*, 2)$, when $D \bmod 3 = 2$. □

C Proof of Theorem 3

In this appendix Theorem 3 is formally proved.

Proof. Intuitively, the average size of an anonymity set is decreased by a factor $\frac{N}{N'}$ when $N' - N$ tags are removed from the full tree. Therefore, the expected resistance should not decrease. We first proof this for $N' = N + 1$ and use the result to generalize the statement to arbitrary N.

Let $P_1, \ldots P_\ell$ be the anonymity sets of the full tree after c tags have been compromised, so $\sum_{j=1}^{\ell} |P_j| = N'$. The average anonymity set size over all tags $\bar{S}_{N'}$ will equal

$$\bar{S}_{N'} = \sum_{j=1}^{\ell} \frac{|P_j|^2}{N'}$$

Note that $\bar{S}_{N'}$ is an instantiation of $\bar{S}_c(B)$ for a particular choice of c compromised tags. When one tag is uniformly chosen to be removed, the probability that this tag is chosen from the j^{th} anonymity set will equal $\frac{|P_j|}{N'}$. So the average anonymity set size over all remaining N tags $\bar{S}_N$ will equal

$$\bar{S}_N = \sum_{j=1}^{\ell} \frac{|P_j|}{N'} \cdot \frac{1}{N} \left\{ (|P_j| - 1)^2 + \sum_{i=1, i \neq j}^{\ell} |P_i|^2 \right\}$$

We derive

$$\sum_{j=1}^{\ell} |P_j| \cdot \left\{ (|P_j| - 1)^2 + \sum_{i=1, i \neq j}^{\ell} |P_i|^2 \right\} = \sum_{j=1}^{\ell} |P_j| \cdot \left\{ 1 - 2|P_j| + \sum_{i=1}^{\ell} |P_i|^2 \right\}$$

$$= N' - 2 \sum_{j=1}^{\ell} |P_j|^2 + N' \sum_{i=1}^{d} |P_i|^2$$

and thus $N \cdot N' \cdot \bar{S}_N = N' + (N' - 2)N' \bar{S}_{N'}$ or

$$N \cdot \bar{S}_N = 1 + (N' - 2)\bar{S}_{N'}$$

Finally, choose ϵ, $0 < \epsilon < N'$, such that $\bar{S}_{N'} = N' - \epsilon$, then $N' + N'(N' - 2)\bar{S}_{N'} = \bar{S}_{N'} + \epsilon + N'(N' - 2)\bar{S}_{N'} = \epsilon + (N' - 1)^2 \bar{S}_{N'}$, and thus $R_N = \frac{\bar{S}_N}{N} = \frac{1 + (N' - 2)\bar{S}_{N'}}{N^2} = \frac{\epsilon + (N' - 1)^2 \bar{S}_{N'}}{N' \cdot N^2} = R_{N'} + \frac{\epsilon}{N' \cdot N^2}$. It follows that R_N and $R_{N'}$ are almost equal:

$$R_{N'} < R_N < R_{N'} + \frac{1}{N^2}$$

Since this holds for every choice of c compromised tags, it also holds for the average case. The generalized upperbound for $1 \leq N < N'$ easily follows. □

Hidden Bits Approach for Authentication in RFID Systems*

Marek Klonowski, Krzysztof Majcher, Wojciech Macyna, and Filip Zagórski**

Institute of Mathematics and Computer Science
Wroclaw University of Technology

Abstract. We present an authentication scheme designed for tiny and strictly constrained devices like RFID-tags. Authentication is based on the symmetric key shared between a tag and a reader. While a tag needs to perform only simple operations in order to authenticate: to pick $n/2+b$ random values and then to compute XORs of some values, a reader needs to try 2^b possible values to check if a tag replied correctly (b is small). At the same time eavesdropping adversary after r executions of the protocol needs to solve a system of rk-multivariate quadratic equations over $GF(2)$ with $nk + rb$ variables.

We present a security discussion of proposed solutions.

Keywords: RFID, lightweight cryptography, authentication.

1 Introduction

In this paper we present an authentication mechanism for ultra-weak devices like RFID-tags providing some level of security for very moderate computations. Our solution is in fact a secure version of a method proposed in [6].

There is a long list of papers devoted to various methods believed to provide security methods to extremely weak devices like RFID-tags. However, most of them cannot be applied to ultra-light devices, because they are based on a standard cryptographic mechanisms like hash functions.

On the other hand, there are other methods based on the non-standard algorithmic approach. Let us recall two lines which are very close to ours.

HB/HB+ Family. HB [18] is a lightweight secret-key protocol for a RFID-tag identification. It bases on the human-to-computer authentication protocol designed by Hopper and Blum (HB, [17]). The security of the HB/HB+ schemes is provable – it is based on the "learning parity with noise" (LPN) problem, which was proved to be NP-hard [3]. However, in recent years, there have been several attacks against LPN problem. Most of them (e.g. *LF2* from [20] or [21]) are tune-ups of the BKW algorithm (Blum, Kalai, Wasserman 2003, [4]).

* Partially supported by Polish Ministry of Science and Higher Education, grant N N206 2701 33.

** Corresponding author.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 48–58, 2013.

The main problem of the HB-scheme is the high communication complexity. According to [20], for a key of 128-bit length and a noise parameter $\frac{1}{4}$, amount of data transmitted between a tag and a reader is about 20KB. And for lower values of the noise parameter, which save on communication bandwidth, there are practical attacks (i.e, [16]).

HB+ protocol is also vulnerable to the man-in-the-middle-type attack proposed by Gilbert, Robshaw, and Silbert [12]. Since then, many other schemes have been proposed to design an LPN-based protocol, which is secure against man-in-the-middle attacks [5,22,14]. Some of them have been already broken – see [10,13,23].

CKK-Type. A fairly practical method for a privacy-preserving authentication has been proposed in [6]. The underlying idea is based on embedding in each RFID-tag (as a secret) linear relations between randomly generated n bits. The proposed method allows to scan a single tag only up to n times (full key recovery). On the other hand, only $n+k$ bits transmitted are needed to recognize a tag by a reader that knows the secret w.h.p. even in a huge batch of potential tags. The paper proposes an extension of the basic method that was believed to extend "tag lifetime". However, this extension was also efficiently attacked in [15] and later completely broken in [19].

In the Section 2.2 we present CKK scheme, since it is one of the building blocks of our proposal. Our work can be treated as a secure version of the CKK approach [6].

Notation. Let us, for any natural l, denote $[l] = \{1, \ldots, l\}$ and by $A||B$ the concatenation of strings A and B, a dot (scalar) product of the vectors x, y by $<x, y>$. For a vector x, its length is denoted by $|x|$ and the i-th coordinate by $x[i]$, in particular, it is the i-th bit of the bitstring x.

For a finite set A , by $x \in_R A$ we understand an object chosen uniformly at random from A.

2 Protocol

2.1 Idea

Our scheme can be regarded as an extension of the CKK method. On the other hand some ideas make the scheme similar to HB-like protocols. The idea is as follows: instead of sending a full CKK response, some randomly chosen bits of independent part (i.e. the input of the function) are blinded. This approach reveals much less information to an adversary eavesdropping the communication. We show that this approach gives a significant improvement of security for relatively small cost. What is more, this cost is paid entirely by a reader which is usually a device with much stronger computational power.

2.2 Building Blocks

In this section we describe the CKK protocol used in our solution as a building block. Procedure $\mathcal{CKK}$ can be treated as a function of two arguments: a secret key s, that contains k bit-strings $s_1, \ldots, s_k$ and a bit-string a. All bit-strings are of the length n. The output of the function is a bit-sting of the length r. The i-th bit of the output is a dot product of strings s_i and a.

$\mathcal{CKK}$ **protocol**
Public parameters: n, k
Secret key: $s = s_1, \ldots, s_k, s_i \in \{0,1\}^n$

Tag		Reader
chooses $a \in_R \{0,1\}^n$		
computes for $i = 1, \ldots, k$		
$r[i] = <a, s_i>$		
$r = (r[1], \ldots, r[k])$		
$out = (a, r)$	$\xrightarrow{out}$	check for $i = 1, \ldots k$
		$r[i] \stackrel{?}{=} <a\|s_i>$

For the sake of a notation clarity, we would just write $r = CKK_s(a)$.

Many variants of this protocol has been broken, nevertheless this protocol provides security against passive adversary eavesdropping the communication between the reader and the tag up to $O(n)$ times. More precisely it is expressed by the theorem proved in [6].

2.3 Protocol Description

Setup – Creation of a Tag A secret key s is established exactly as in the case of the CKK protocol (cf. Section 2.2). A key is shared between a tag and a reader. In the phase of manufacturing a key is embedded in a tag. Each tag has also its fixed identifier ID. We assume that this value may be commonly known.

Blinding Function $Blinded(\cdot,\cdot) : \{0,1\}^n \times 2^{[n]} \to \{0,1,\_\}^n$ is defined as follows: let $y = Blinded(x, B)$. If $i \notin B$ then $y[i] = x[i]$, otherwise $y[i] = \_$. One can say, that $Blinded(x, B)$ is a vector x with "blinded" positions from the set B. For $x = (0,1,1,0,1,0,0,0)$ and $B = \{2,3,6\}$ we get $Blinded(x, B) = (0, \_, \_, 0, 1, \_, 0, 0)$

Authentication We assume that $2|n$, and a challenge string sent by a READER has length $n/2$ while another $n/2$ bits are randomly chosen by a TAG, k is the length of the output. We denote by $b \geq 2$ the number of bits of the answer that remains hidden to the READER. The authentication protocol is following:

READER sends "Hello" message.
TAG sends "Hello" message and its identifier ID.

READER chooses at random $d \in_R \{0,1\}^{n/2}$ and sends d to TAG.
TAG Executes the following steps:

1. Chooses at random value $e \in \{0,1\}^{n/2}$
2. Computes $r := \mathcal{CKK}_s(d||e)$
3. Chooses at random subset B of b distinct values from $[n/2]$
4. Sends $(Blinded[e,B], r)$

READER Accepts if $\exists x, B', s$ such that:

- The secret key assigned to the tag with identifier ID is s
- $r == \mathcal{CKK}_s(d||x)$
- $Blinded[e,B] == Blinded[x,B']$

2.4 Analysis

Complexity Communication complexity is low, a reader needs to send $n/2$ bits. The tag sends $|ID| + \frac{n}{2} + k$ bits during each execution of the authentication protocol (and taking into account need of encoding of the "blinded" bits, the amount of bits sent on average is $|ID| + \frac{3}{2}n + \frac{b}{2} + k$).

Computations on a tag's side is a single CKK execution with n bits - i.e., at most kn XOR operations. A reader, during the verification phase needs to "check" CKK for at most 2^b possible hidden bit values (since it does not know exact values chosen by a tag).

2.5 Correctness

A tag, having a secret key s acting according to the protocol is always verified positively.

Now we compute the probability that a randomly chosen response will be accepted by a reader as a given tag. Let us consider a situation when an adversary is going to simulate a tag T with the secret key s. Suppose that during the execution of the protocol a reader sends to the tag a challenge $d \in \{0,1\}^{\frac{n}{2}}$, and an adversary gives a response $(Blinded[e,B], r)$, where e, and r are randomly chosen vectors of the appropriate length, and B is a randomly chosen b-elements subset of $[n/2]$. Let us denote by F a function $F : \{0,1\}^b \to \{0,1\}^k$ such that $F(b_1, b_2, ..., b_b) = \mathcal{CKK}_s(d||e_{b_1,b_2,...,b_b})$, where $e_{b_1,b_2,...,b_b}$ is the completion by bits $\{b_1, b_2, ..., b_b\}$ of the vector $Blinded[e,B]$.

The probability that the adversary will be accepted by the reader as the tag T is in fact equal to the probability that the vector r is in image of F. More precisely, the probability equals to $\frac{|Im(F)|}{|\{0,1\}^k|}$.

One can see that F is a linear function (because $\mathbb{CKK}_s$ is a linear function). Since the secret key s is chosen randomly, then with high probability a matrix of function F has a rank equal $min\{b,k\}$. So a dimension of the image of F is $min\{b,k\}$.

Finally the search probability is 2^{k-b} if $k > b$, or 1 otherwise.

2.6 Practical Parameters

The size of the key embedded in a tag is proportional to $O(nk)$ (each bit of the answer part may depended on up to n bits of the independent part).

Parameter n obviously influences on: number of gates, size of communication between a tag and a reader and security. This parameter needs to be greater than k (dis-ambiguity).

Value of the parameter b should be chosen at the level that from one side it makes a corresponding (to observation) system of quadratic equations hard to solve [8,1]. But on the other hand it cannot be chosen to high because a reader needs to perform in the worst case 2^b checks before a correct bit-values are found.

Parameter k corresponds to the probability of a successful authentication for a randomly chosen bits. So it is responsible for the expected time of exhaustive search (brute-force attack). Moreover because of the fact that some bits are hidden and there is a possibility of ambiguity, a value $k - b$ should be also taken into account for the brute-force attack.

Dependency between parameters is following: $b < k - b < k \leq n/2$. Security level of m bits is achieved by $m := k - b$ (80-bit security for $b = 20, k = 100, n = 200$). Ineficiency (related to the key size) is inherited from the original scheme [6], what one gets in return is information-theoretic security for up to k-tag reads (even when an adversary knows hidden bits, the corresponding system of linear equations is underdefined), above k reads security depends on hardness of solving quadratic equations (see Section 3.3).

3 Security Discussion

In this section we present some arguments for the security of the protocol. We present a sketch of the security proof in Section 3.3.

Here we present that our protocol is immune against typical attacks on a class of protocols based on a linear functions (in the sense that a response of the tag is some modification of the linear function value on a challenge). When one studies published attacks on such protocols, one can see that each of them consists of three following sub-algorithms:

- an adversary builds a system of equations based on collected observations, a solution of this system leads to a key-recovery.
- an adversary collects a set of pairs: challenge-response, until collected challenges form a basis of the linear space.
- an adversary seeks pairs: challenge-response, which the challenge parts are linear dependent, i.e., they sum up to a zero vector.

We show that in the case of the protocol every of the approaches listed above does not lead to an efficient attack.

Observations as a Quadratic System. A transcription of r rounds of the protocol corresponds to a quadratic system of $E = rk$ equations with $V = nk+br$ binary variables. Each equation contains $n - b$ linear terms and b terms of the second order (of the form $x_{ij}y_j$). Each execution of the protocol generates b new variables that appear only in k equations.

As an example, see how the observations lead to the system of quadratic equations:

$$(1,\_,0,\_,1,1:1,0) \leftrightarrow \begin{cases} x_{1,1} \oplus b_{1,2}x_{1,2} \oplus 0 \oplus b_{1,4}x_{1,4} \oplus x_{1,5} \oplus x_{1,6} = 1 \\ x_{2,1} \oplus b_{1,2}x_{2,2} \oplus 0 \oplus b_{2,4}x_{2,4} \oplus x_{2,5} \oplus x_{2,6} = 0 \end{cases}$$

$$(\_,0,1,0,0,\_:1,1) \leftrightarrow \begin{cases} b_{2,1}x_{1,1} \oplus 0 \oplus x_{1,3} \oplus 0 \oplus 0 \oplus b_{2,6}x_{1,6} = 1 \\ b_{2,1}x_{2,1} \oplus 0 \oplus x_{2,3} \oplus 0 \oplus 0 \oplus b_{2,6}x_{2,6} = 1 \end{cases}$$

$$(\_,1,0,0,\_,0:1,1) \leftrightarrow \begin{cases} b_{3,1}x_{1,1} \oplus x_{1,2} \oplus 0 \oplus 0 \oplus b_{3,5}x_{1,5} \oplus 0 = 0 \\ b_{3,1}x_{2,1} \oplus x_{2,2} \oplus 0 \oplus 0 \oplus b_{3,5}x_{2,5} \oplus 0 = 1 \end{cases}$$

According to [9,11,2], the system of the random quadratic equations is NP-hard for the parameters E, V that are used in the protocol.

3.1 Collecting a Basis

In [6] it has been shown that the basic scheme is secure against a passive adversary as long as the adversary can only collect up to $O(n)$ different observations. However, in [15] has been proved that even a few more observations lead to corrupting of the system and retrieving the key (thanks to the linear combination of the collected observations).

In the theorem below we show that having even n observations is far not enough to reconstruct the basis.

Lemma 1. *Let us assume that the adversary has collected n vectors that represent n observations – a matrix M_0. If there is a supplement of M_0 with determinant equal 1, there exist $2^{n(b-1)}$ different supplements with determinant 1.*

Proof. Note that each vector of M_0 there are exactly b symbols $-$ (i.e., a blinded bit). Let M be a supplement of M_0 with determinant 1.

For $1 \leq j \leq n$ let us consider the j-th row of a matrix M_0. Let $I = \{i_1, i_2, ..., i_b\}$ be the set of indices of $-$ symbol in j-th row. For $1 \leq k \leq d$, we denote by M_{i_k} a matrix such that in the j-th row in the i_k-th column there is 1 and 0 on the other positions. The other rows are exactly as in the matrix M.

Let I_0, I_1 be the partitions of I such that $i_k \in I_0$, if and only if $det(M_{i_k}) = 1$. Let us consider a vector v of the length n, such that for $i \notin I$ i-th bit of v is 0 and $|\{i \in I_0 : i\text{-th bit of } v \text{ is } 1\}|$ is even. Let us consider matrix M_v such that

j-th row of it is a sum of v and j-th row of M and the rest are the same as in M. Since determinant is a n-linear function, then $det(M_v) = 1$. Note that there is $2^{|I_0|-1}\dot{2}^{|I_1|} \leq 2^{b-1}$ such vectors.

As we observed, changing one row leads to 2^{b-1} different linearly independent supplements. Since there are n rows, there are $2^{n(b-1)}$ linearly independent supplements. □

From the point of view of the adversary, the above lemma says that collecting even n observations does not give precise information about real values.

3.2 Linear Dependence between Challenges

Let us recall that most of the attacks against HB and CKK-type protocols use explicitly knowledge about linear dependencies of eavesdropped observations. Following lemmas show that in the case of the presented protocol it is hard to point linear relations between observations when only blinded observations are given to the adversary.

In the lemma below we show that an adversary given huge set of independent parts (coming possibly from various tags) cannot easily find linearly dependent supplement.

We say that the i-th column of the set is *blinded* if there is at least one vector with blinded bit on the i-th position, i.e. there is a $1 \leq k \leq t$ $v_k[i] =' -'\}$.

Lemma 2. *Let $v_1, v_2, \ldots, v_t$ be a set of blinded independent parts observed by the adversary. Let l be a number of blinded columns for this set. Moreover, all vectors sums to 0 on positions different from blinded columns. The probability that a random supplement of $v_1, v_2, ..., v_t$ sums to 0, is $\frac{1}{2}^l$.*

Proof. Let $v = v_1 + v_2 + \ldots + v_t$ be sum of all vectors. We consider the $v[i] = \sum_k v_k[i]$, if i is not a blinded column $v[i] = 0$ by the assumption. Otherwise, one can easily see that $v[i] = 0$ for exactly half of assignments of blinded bits. Since there are l blinded columns, a probability that a random assignment gives $v = 0$ is $1/2^l$.

The lemma below shows that the number of blinded columns is relatively high w.h.p.

Lemma 3. *Let $v_1, v_2, \ldots, v_t$ be vectors such that each of them has independently chosen b blinded bits. Let X be the random variable that denotes the number of blinded columns for the matrix build of those vectors. Then for every $0 < \varepsilon < 1$*

$$\Pr[X < (1-\varepsilon)n/2 \cdot (1 - \left(1 - \frac{2b}{n}\right)^t)] < \exp\left(-\frac{\varepsilon^2 n \cdot (1 - \left(1 - \frac{2b}{n}\right)^t)}{4}\right) .$$

Proof. One can easy see that the probability that a particular column is blinded is $1 - \left(1 - \frac{2b}{n}\right)^t$. As a consequence, X stochastically dominates the binomial distribution $Bin(n/2, 1 - \left(1 - \frac{2b}{n}\right)^t)$ and $E[X] = n/2 \cdot (1 - \left(1 - \frac{2b}{n}\right)^t)$. The fact follows from the Chernoff bound.

3.3 Security

In the current section we present a sketch of the security proof. The sketch contains of two steps. In the first part we argue that a system of equations corresponding to the protocol executions (for $k = 1$) is hard to be solved by an adversary. The second part presents a reduction of a problem for an arbitrary $k > 1$ to the case of $k = 1$, i.e. we show that the protocol is as hard as solving a system of multivariate quadratic equations over $GF(2)$.

Definition 1. *An (n, k, l)-adversary is an algorithm $\mathcal{A}$ which on an input:*

- *a sequence of pairs $\{(x_i, y_i)\}_{i=1,\ldots,l}$, such that x_i is a vector over $\{0, 1, \_\}^n$, and y_i is a vector over $\{0, 1\}^k$ (set of observations),*
- *x^0 a vector of the length $\frac{n}{2}$ (a challenge).*

produces an output:

- *x^1 - a vector of the length $\frac{n}{2}$ over $\{0, 1, \_\}$, with exactly b blinded bits,*
- *$y \in \{0, 1\}^k$.*

Definition 2. *We say that an (n, k, l)-adversary is ϵ-adversary if and only if for every sequence of $\{(x_i, y_i)\}_{i\in[l]}$ that corresponds to some key of a tag T and for a randomly chosen challenge x_0 by a Reader, the probability that an answer of the adversary is accepted as an answer of T is greater than $1 - \epsilon$, i.e.:*

$$Pr(A(x^0, x^1) = T(x^0, x^1)) \geq 1 - \epsilon.$$

MQ Problem – Security for $k = 1$. A general MQ problem is the problem of solving systems of multivariate quadratic equations, i.e. finding a value $x \in GF(q)^n$ such that for every equation from a given system $S = (Q_1, \ldots, Q_m)$ it holds $Q_i(x) = 0$ (of course such an x may not exist).

Difficulty of solving instances of MQ depends on values of n, m and q. We are interested only in the case where $q = 2$. When the number of equations is significantly smaller than the number of variables, finding a solution can be made efficiently [7]. The same happens in the opposite case, i.e. for over-defined system, when one has access to a number of equations which is of order of $\approx n^2$ – in such a case there are efficient linearization-based algorithms.

But when one deals with arbitrary values of m and n, solving quadratic equations is NP-hard even for the case of $GF(2)$ [9,11]. The case which corresponds to the protocol: $q = 2, m = cn$ for $c > 1$ but small compared with $\frac{n}{2}$ is also a hard one [2]. The best algorithms like XL [8] or F4/F5 [1] are exponential in n for a randomly chosen quadratic system.

There are n variables corresponding to the key (for $k = 1$). Each execution of the protocol gives an adversary a new quadratic equation with b new variables. So after l executions of the protocol there are $m = l$ equations and $n + bl$ variables. Those equations have a very special form: they are relatively short (there are at most $n + 1$ terms); most of the variables (exactly bl of them) occurs only once; there are no terms of the form x_i^2, only of the form $x_i x_j$.

Reduction – Security for $k > 1$. Here we show that if the protocol is secure for $k = 1$ then it is also secure for larger values of k. This comes from the fact that one can analyze data observed during protocol execution independently for each answer bit independently since the only common values are:

- a challenge x^0 sent by a Reader,
- a random string x^1 generated by a Tag,
- b blinded places in x^1.

On the other hand, key-bits are distinct (there are k groups of the length of n each) as well as answer bits (there are k of them).

We show that if there exists an (n, k, l)-adversary then there exists $(n, 1, l)$-adversary.

Theorem 1. *If there exists a polynomial time (n, k, l)-adversary then there exist a polynomial time $(n, 1, l)$-adversary.*

Proof. Let A be a polynomial time (n, k, l)-adversary. Suppose, that we have a sequence $\{(x_i, y_i)\}_{i\in[l]}$, with $|x_1| = n$ and $|y_i| = 1$, is recognized by a Reader as a tag T. Denote by s a secret of a tag T. Consider a sequence $\{(x_i, y'_i)\}_{i\in[l]}$, where $|y'_i| = k$, and $y'_i = (y_i, 0, 0..., 0)$. One can see that each element of this sequence could be recognized by R as a tag T with a secret $S = (s, s_1, s_2, ..., s_{k-1})$, where each s_i is equal to zero. Let us take a random x_0, a tuple of the length $\frac{n}{2}$. Suppose that an algorithm on a sequence $\{(x_i, y'_i)\}_{i\in[l]}$, and on x_0 gives an output x_1, y, so we have: $Pr((x_0, x_1, y)$ is recognized for a Reader as a tag $T') = 1 - \epsilon$.

Now let suppose that (x_0, x_1, y) is recognized by a Reader as a tag T', and let b be first bit of a vector y. It is easy to see, that (x_0, x_1, b) is recognized by a Reader as a tag T.

4 Conclusion

We presented a new authentication scheme for lightweight devices, as for example RFID tags. Our protocol can be seen as a combination of originally insecure CKK with inefficient HB.

The main innovation of the paper is that we move computational complexity from a tag to a reader.

Future work includes:

- Proof of security by reduction to some hard problem, for example MQ-problem.
- Proposing the new identification scheme based on the paradigm of moving of the computational complexity from a tag to a reader.

References

1. Ars, G., Faugère, J.-C., Imai, H., Kawazoe, M., Sugita, M.: Comparison Between XL and Gröbner Basis Algorithms. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 338–353. Springer, Heidelberg (2004)

2. Bardet, M.T.: Etude des systèmes algébriques surdéterminés (2004)
3. Berlekamp, E.R., McEliece, R.J., van Tilborg, H.C.A.: On the inherent intractability of certain coding problems. IEEE Trans. Info. Theory, 384–386 (1978)
4. Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. Journal of the ACM 50(4), 506–519 (2003)
5. Bringer, J., Chabanne, H., Kevenaar, T.A.M., Kindarji, B.: Extending Match-On-Card to Local Biometric Identification. In: Fierrez, J., Ortega-Garcia, J., Esposito, A., Drygajlo, A., Faundez-Zanuy, M. (eds.) BioID MultiComm 2009. LNCS, vol. 5707, pp. 178–186. Springer, Heidelberg (2009)
6. Cichoń, J., Klonowski, M., Kutyłowski, M.: Privacy Protection for RFID with Hidden Subset Identifiers. In: Indulska, J., Patterson, D.J., Rodden, T., Ott, M. (eds.) PERVASIVE 2008. LNCS, vol. 5013, pp. 298–314. Springer, Heidelberg (2008)
7. Courtois, N., Goubin, L., Meier, W., Tacier, J.-D.: Solving Underdefined Systems of Multivariate Quadratic Equations. In: Naccache, D., Paillier, P. (eds.) PKC 2002. LNCS, vol. 2274, pp. 211–227. Springer, Heidelberg (2002)
8. Diem, C.: The XL-Algorithm and a Conjecture from Commutative Algebra. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 323–337. Springer, Heidelberg (2004)
9. Fraenkel, A.S., Yesha, Y.: Complexity of solving algebraic equations. Inf. Process. Lett., 178–179 (1980)
10. Frumkin, D., Shamir, A.: Un-trusted-hb: Security vulnerabilities of trusted-hb. Cryptology ePrint Archive, Report 2009/044 (2009)
11. Garey, M.R., Johnson, D.S.: Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman (1979)
12. Gilbert, H., Sibert, H., Robshaw, M.: An active attack against a provably secure lightweight authentication protocol. IEEE Electronic Letters 41, 1169–1170 (2005)
13. Gilbert, H., Robshaw, M.J.B., Seurin, Y.: Good Variants of HB^+ Are Hard to Find. In: Tsudik, G. (ed.) FC 2008. LNCS, vol. 5143, pp. 156–170. Springer, Heidelberg (2008)
14. Gilbert, H., Robshaw, M.J.B., Seurin, Y.: $HB^\#$ Increasing the Security and Efficiency of HB^+. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 361–378. Springer, Heidelberg (2008)
15. Gołębiewski, Z., Majcher, K., Zagórski, F.: Attacks on CKK Family of RFID Authentication Protocols. In: Coudert, D., Simplot-Ryl, D., Stojmenovic, I. (eds.) ADHOC-NOW 2008. LNCS, vol. 5198, pp. 241–250. Springer, Heidelberg (2008)
16. Gołębiewski, Z., Majcher, K., Zagórski, F., Zawada, M.: Practical Attacks on HB and HB+ Protocols. In: Ardagna, C.A., Zhou, J. (eds.) WISTP 2011. LNCS, vol. 6633, pp. 244–253. Springer, Heidelberg (2011)
17. Hopper, N.J., Blum, M.: Secure Human Identification Protocols. In: Boyd, C. (ed.) ASIACRYPT 2001. LNCS, vol. 2248, pp. 52–66. Springer, Heidelberg (2001)
18. Juels, A., Weis, S.A.: Authenticating Pervasive Devices with Human Protocols. In: Shoup, V. (ed.) CRYPTO 2005. LNCS, vol. 3621, pp. 293–308. Springer, Heidelberg (2005)
19. Krause, M., Stegemann, D.: More on the Security of Linear RFID Authentication Protocols. In: Jacobson Jr., M.J., Rijmen, V., Safavi-Naini, R. (eds.) SAC 2009. LNCS, vol. 5867, pp. 182–196. Springer, Heidelberg (2009)
20. Levieil, É., Fouque, P.-A.: An Improved LPN Algorithm. In: De Prisco, R., Yung, M. (eds.) SCN 2006. LNCS, vol. 4116, pp. 348–359. Springer, Heidelberg (2006)

21. Lyubashevsky, V.: The Parity Problem in the Presence of Noise, Decoding Random Linear Codes, and the Subset Sum Problem. In: Chekuri, C., Jansen, K., Rolim, J.D.P., Trevisan, L. (eds.) APPROX 2005 and RANDOM 2005. LNCS, vol. 3624, pp. 378–389. Springer, Heidelberg (2005)
22. Munilla, J., Peinado, A.: Hb-mp: A further step in the hb-family of lightweight authentication protocols. Comput. Netw. 51(9), 2262–2267 (2007)
23. Ouafi, K., Overbeck, R., Vaudenay, S.: On the Security of HB$^{\#}$ against a Man-in-the-Middle Attack. In: Pieprzyk, J. (ed.) ASIACRYPT 2008. LNCS, vol. 5350, pp. 108–124. Springer, Heidelberg (2008)

Designated Attribute-Based Proofs for RFID Applications

Gergely Alpár[1,2], Lejla Batina[1,3], and Wouter Lueks[1,2]

[1] Radboud University Nijmegen, ICIS/Digital Security group
Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands
{gergely,lejla,w.lueks}@cs.ru.nl
[2] TNO Information and Communication Technology, The Netherlands
[3] K.U. Leuven ESAT/SCD-COSIC and IBBT
Kasteelpark Arenberg 10, B-3001 Leuven-Heverlee, Belgium
lejla.batina@esat.kuleuven.be

Abstract. Recent research has shown that using public-key cryptography in order to meet privacy requirements for RFID tags is not only necessary, but also now practically feasible. This has led to the development of new protocols like the Randomized Schnorr [6] identification protocol. This protocol ensures that the identity of a tag only becomes known to authorised readers.

In this paper we generalize this protocol by introducing an attribute-based identification scheme. The proposed scheme preserves the designation of verification (i.e., only an authorised reader is able to learn the identity of a tag) while it allows tags to prove any subset of their attributes to authorised readers. The proposed scheme is proven to be secure and narrow-strong private.

Keywords: RFID, identification, authentication, elliptic curve cryptography, security, privacy, attribute-based credential.

1 Introduction

We rely on the security of embedded systems in our daily lives when using, e.g., public transportation, mobile phones, e-banking applications and pay TV systems. Typically, these systems are implemented using smart cards and Radio Frequency IDentification (RFID) tags that are extremely limited in resources such as area, memory and power consumption. As a result of these limitations, ensuring security and privacy in RFID systems is one of the most difficult challenges today.

Recently, with the development of privacy-sensitive RFID services the attention of the research community has yet again returned to Public-Key Cryptography (PKC) for RFID systems. The main reason is the need to give the users more privacy, but other properties such as scalability and anti-counterfeiting are also very important. Although critics still consider public-key systems too expensive for passive tags, a number of companies and academic groups have

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 59–75, 2013.

already designed PKC-based chips for RFID protocols [5,10,11]. While the first performance results are promising, the design of new PKC-based protocols is necessary to get viable solutions. One of the reasons for this is that standard solutions for authentication, e.g., signatures, typically require a hash function as well, which adds additional burden in terms of gates and power consumption.

Elliptic Curve Cryptography (ECC) has typically been the preferred setting for creating PKC-based protocol. Numerous protocols were designed aiming at security and privacy of RFID systems based (exclusively) on the Elliptic Curve (EC) point multiplication [6,15]. Starting from an authentication of a single tag, a number of more complex protocols emerged such as grouping proofs [13] and hierarchical proofs [1]. It has become obvious that this trend will continue as lightweight cryptography is turning into the main component of modern communication networks and new applications are ever emerging.

Attribute-based credentials[1] (ABCs) in combination with selective disclosure can be used to solve many of the existing privacy problems. Furthermore, showing an ABC does not require more exponentiations on the tag's side than some other protocols demonstrated to be computationally feasible on RFID chips [1,3]. We demonstrate how RFID systems can benefit from the concepts of ABCs in such a way that a tag proves all or a subset of its secret attributes to designated verifier readers. Our protocols are proved to be secure against impersonation attacks and narrow-strong private.

1.1 Related Work

The discrete logarithm (DL) problem is considered to be hard[2], that is, finding the exponent a of a random point $A = aP$ with respect to a publicly-known base point P is computationally infeasible [14]. A related hard problem can be stated with respect to more base points: in the discrete logarithm representation (DL-REP) problem, introduced by Brands [4] and employed in Microsoft's U-Prove technology [16], given a random point A and base points $P_0, \ldots, P_l$ one has to find exponents $a_0, \ldots, a_l$ such that $A = \sum_0^l a_i P_i$. While in a DL problem the exponent a is uniquely determined, in a DL-REP problem there are several tuples $(a'_0, \ldots, a'_l)$ for which $A = \sum_0^l a'_i P_i$.

Zero-knowledge proofs of knowledge are cryptographic techniques to prove the knowledge of a secret value without revealing any information about the secret itself. Schnorr [18] proposed an interactive protocol that enables a prover to show the knowledge of a DL value to a verifier, that is, a secret scalar corresponding to a public point. A similar method by Brands [4] can be applied to prove the knowledge of a DL-REP of a point with respect to a tuple of base points. A prover

[1] Attribute-based credential: (aka. anonymous credentials) An ABC is a composite commitment that carries multiple values, so-called attributes, and signed by a trusted authority; in this paper the signing is ignored as the initialization of RFID tags are out of scope of this study.

[2] Although in this paper we discuss schemes in an ECC setting, they work in any groups in which the discrete logarithm problem is hard.

can also reveal a subset of scalars in a DL-REP while only proving knowledge about the others; this procedure is called *selective disclosure* which is one of the most relevant functionalities in relation to ABCs [7].

While the techniques above provide a high-degree of security for the prover by not necessarily revealing secret pieces of data, any party interacting with the prover (or eavesdropping) can learn selectively disclosed information. In some scenarios, prior verifier authentication is not possible (i.e., a secure channel cannot be assumed between the prover and the intended verifier) and, therefore, this may not be desirable. In particular, in the context of RFID systems, possibly malicious readers can interrogate RFID tags. *Designated verifier proofs* are proofs of knowledge where only a designated verifier can obtain identity information. Bringer et al. [6] present a scheme that extends a Schnorr identification to a designated verifier proof. Since their technique randomizes the messages for a party that does not know the secret key, they call it the "Randomized Schnorr" scheme.

Restriction of verification has a long history in cryptography. Undeniable signatures have been introduced in 1989 by Chaum and van Antwerpen [9] and have been enhanced to zero-knowledge proofs of ownership by Chaum [8]. An undeniable signature cannot be verified without interacting with its signer. Furthermore, during the proving protocol no external parties learn anything about the validity or invalidity of the signature. Jacobsson et al. [12] propose a more general notion, the designation of verification that a statement is true. The idea is that the prover generates a zero-knowledge proof that can only be produced by him and the verifier. Since the verifier knows that she was not the one who created the proof, she becomes convinced about the validity of the statement; however, she cannot convince any third party that the proof was produced by the prover and not by herself. Saeednia et al. [17] improve the notion of designated verifier signatures, in which not only the verifier but anybody can simulate transcripts of valid proof conversations. They also propose an efficient designated verifier signature using the Fiat–Shamir heuristic.

In the context of RFID schemes, Bringer et al. [6] present a similar but interactive scheme, in which a tag demonstrates its identifier. While the tag proves the knowledge of a secret key, it reveals its identifier only to the designated verifier. Our schemes are also designated verifier proofs. More precisely, we generalize the scheme of Bringer et al., but we allow for multiple attributes. To our best knowledge, we introduce the first designated selective disclosure protocol in which a prover can reveal any subset of attributes but only to a verifier that knows all corresponding secret keys.

1.2 Our Contribution

By encoding several attributes as exponents a_i's in a point $A = \sum_0^l a_i P_i$, the number and variety of applications increase considerably. As Lee et al. [15] have shown that multiple point multiplications are feasible on a passive RFID tag, DL-REP related protocols also become realizable in such limited environments. In this paper we propose designated verifier proofs of knowledge of DL-REPs.

Furthermore, by applying a new proof technique, a subset of exponents can also be shown to a designated verifier.

Unlike in Brands' selective disclosure schemes, a prover cannot send the revealed exponents in advance or during a protocol run in clear; however, those exponents have to be computable for an eligible verifier. Table 1 shows a summary of these zero-knowledge proofs. Although a trivial solution to reveal certain attributes would be to send them encrypted to the verifier, this causes significant overhead.

Table 1. Designated proofs of knowledge of discrete logarithm values; highlighted protocols are presented in this paper for the first time

Problem	**ZK proof**	**Designated ZK proof**
DL	Schnorr [18]	Randomized Schnorr [6]
DL-REP	U-Prove showing [4]	Designated DL-REP proof
DL-REP	U-Prove selective disclosure [4]	Designated partial DL-REP proof

We prove that the proposed designated verifier schemes are zero-knowledge, secure against impersonation, and narrow-strong private. Moreover, we show how these general building blocks can be employed to design secure RFID applications.

The remainder of the paper is organized as follows. First we describe the cryptographic background and relevant protocols in Section 2. Second we introduce the new Designated Verifier DL-REP and Designated Verifier Partial DL-REP proofs in Section 3. Then we study feasibility of our schemes and some possible RFID applications in Section 4. Finally, we conclude the paper in Section 5.

2 Cryptographic Background

An identification scheme is an interactive protocol in which a prover, i.e., a tag in this setting, convinces a verifier that it has the identity it claims to have. Before we introduce our proposal for a designated attribute-based proof system in Section 3, we describe cryptographic requirements and prior relevant protocols.

2.1 Basic Set-Up

Throughout this paper let E be an elliptic curve defined over a finite field $\mathbb{F}_{q^k}$, where $q = 2$ or a large prime (in this case $k = 1$). Let $(G, +)$ denote a cyclic group of prime order p of points on the curve E, generated by a point P. The fields of characteristic 2, i.e., when $q = 2$, are more suitable for hardware implementations and hence for RFID tags, but ECC protocols conceptually apply for arbitrary fields.

We use capital letters, like A and P to denote points on the elliptic curve. Scalars are written using lower case letters. We write kP to denote the point P

added k times to itself. Finally, we denote by $x \in_R \mathbb{Z}_p$ that x is chosen uniformly at random from the set $\mathbb{Z}_p$.

We use a number of hardness assumptions to prove the security and privacy of our systems. We assume that the following problems are hard[3].

Definition 1 (Discrete Logarithm (DL) problem). *Given a generator $P \in G$ and a multiple $A = aP$ of P, where $a \in_R \mathbb{Z}_p$, determine a.*

Definition 2 (Decisional Diffie-Hellman (DDH) problem). *Given a generator P, and the points $A = aP$, $B = bP$ and $C = cP$, where $a, b \in_R \mathbb{Z}_p$, determine whether $c = ab$.*

2.2 Security and Privacy Model

The protocols we consider are typical authentication protocols. This means that the tag and the reader engage in a protocol, at the end of which the reader will either be convinced about the identity of the tag (and in our case also the validity of the attributes) or it will report failure. In this paper we show that our two new protocols satisfy two different requirements: security and privacy. The former roughly means that it is difficult for an adversary to pretend to be a valid tag, while the latter means that an adversary cannot distinguish legitimate tags from simulated tags.

Just as Bringer et al. [6], we follow the security model proposed by Vaudenay [19]. In his model, Vaudenay describes how adversaries can interact with a set of tags. Besides offering methods for communicating with and choosing from the tags as well as communicating with the reader, the model also exposes two additional oracle calls. The level of access to these two additional oracles defines the type of the adversary.

The first additional oracle is the result-oracle. As it is typical in identification protocols, the reader draws one of the following two conclusions at the end of the protocol. It either concludes that the tag it communicated with has been successfully identified as the tag with identity I, or it reports failure. The result-oracle will return only the success/failure status of the reader. In our protocols we do not allow this type of queries, hence resulting in a *narrow* adversary (as opposed to a *wide* one that is allowed to make such queries).

The second additional oracle is the corruption oracle. This allows the adversary to corrupt a tag, and hence learn all its secrets. We consider only *strong* attackers, i.e., attackers that can obtain the secrets of any tags they choose. In the privacy game, further attacks on the privacy of these tags are allowed afterwards, while they are (of course) explicitly prohibited in the security game.

Given this model we can now give games to define the security model.

Definition 3 (Security game). *Assume that there exists a system of t tags that can be interrogated via the identification protocol, then the game consists of two phases:*

[3] We state these assumptions in the ECC setting.

1. *In the first phase, the adversary is allowed to interrogate any tag multiple times. Furthermore, it is allowed to corrupt any tags of its choosing.*
2. *In the second phase, the adversary communicates with the verifier to impersonate one of the uncorrupted tags of the system.*

An RFID scheme is secure if no adversary can win the Security game above with non-negligible probability.

Intuitively, our notion of privacy for these types of protocols means that it is not possible to link two different executions of the protocol. This property is often referred to as unlinkability. In the Vaudenay model it is captured as follows. Even though the adversary is given the identifiers of the tags it talked to at the end of the game, it cannot distinguish between the setting in which it communicates with actual tags and the setting in which it communicates with simulated tags. Note that in the latter case the simulator didn't know the identifiers. Hence, any information leak on the identifiers can be used by the adversary to gain an advantage. A system has narrow-strong privacy if no adversary can win the following game against a challenger with non-negligible probability.

Definition 4 (Narrow-Strong Privacy Game). *Assume that there exists a system of t tags that can be interrogated via the identification protocol. First, the challenger generates a bit $b \in_R \{0,1\}$ and depending on b, it runs different experiments:*

- *If $b = 0$, the adversary is allowed to directly talk to any tag of its choice.*
- *If $b = 1$, the adversary is not allowed to interrogate tags directly but the challenger, without interacting with the actual tags, simulates them.*

Then in the corruption phase, the adversary can receive all the tag's private information by corrupting it. At the end of the game, the adversary must guess the value of bit b.

Since we are in a strong setting, the challenger can obtain the tags identifiers using the corruption query; therefore, this is not mentioned separately in the game.

2.3 Randomized Schnorr Scheme

Bringer et al.'s Randomized Schnorr [6] scheme is secure and narrow-strong private by the definitions above. Each prover (tag) has a secret key x and an identifier $I = xP$, while each verifier has a secret key v and a corresponding (designating) public key $V = vP$. Verifiers store a list of valid tag identifiers.

During a protocol run (see Fig. 1), not only does a tag prove the knowledge of its secret key, but it also hides its identifier from any external party. Using its secret key v, the verifier can compute the tag's identifier. Therefore, the prover's secret key and its identifier are protected: First, as this scheme is a modified Schnorr identification, no adversary can learn anything about the tag's secret key x. Second, without the knowledge of v, no adversary can compute I.

Prover $x, I = xP$	P, $V = vP$	**Verifier** v
$\alpha, \beta \in_R \mathbb{Z}_p$		
$A_1 := \alpha P$		
$A_2 := \beta V$	$\xrightarrow{A_1, A_2}$	
	$\xleftarrow{c}$	$c \in_R \mathbb{Z}_p^*$
$r := c \cdot x + \alpha + \beta \pmod p$	$\xrightarrow{r}$	Verification: $I = c^{-1}(rP - A_1 - v^{-1}A_2)$ check whether I is a valid identifier

Fig. 1. Randomized Schnorr [6] identification, i.e., Designated Verifier Schnorr identification. (There are 2 point multiplications on the Prover's side.)

2.4 Discrete Logarithm Representation (DL-REP)

Discrete logarithm representations [4] were introduced by Brands[4]. Given a set of $l+1$ generators (base points, in case of ECC) $P_0, \ldots, P_l$ in a group, participants can commit to l (attribute) values $x_1, \ldots, x_l$. We say that the DL-REP of I is $(x_0, \ldots, x_l)$ with respect to $(P_0, \ldots, P_l)$ if $I = \sum_0^l x_i P_i$.

While the identifier I (as a cryptographic commitment) hides the attributes $(x_1, \ldots, x_l)$ because of the extra scalar x_0 unconditionally, it binds the prover only computationally. However, this computation is infeasible as any oracle that, after changing some exponents, can compute a new DL-REP $x'_0, \ldots, x'_l$ with respect to the same base points $P_0, \ldots, P_l$ can be used to break the discrete logarithm problem.

In [4] Brands builds an anonymous credential system on commitments in which a credential is a commitment, like I above, (blindly) signed by a credential authority. Using such a credential and zero-knowledge proof techniques, a prover is able to demonstrate to a verifier that she knows the secret values in the commitment without actually showing them. Moreover, a prover can selectively disclose values corresponding to a disclosure index set $\mathcal{D} \subseteq \{1, \ldots, l\}$ (see Fig. 2). Having these values $(x_i)_{i \in \mathcal{D}}$, the verifier can compute a partial commitment $com - \sum_{i \in \mathcal{D}} x_i P_i$ and the prover can prove the knowledge of all other secret values. Note that in case of $\mathcal{D} = \emptyset$, this scheme is a proof of knowledge of all exponents – we will refer to this protocol as *U-Prove showing protocol*.

3 Designated Verifier DL-REP Proofs

A designated verifier DL-REP proof is an interactive identification protocol in which a prover reveals his unique identifier and at the same time proves knowledge of the identifier's DL-REP with respect to points $P_0, \ldots, P_l$ in a way that

[4] We will often refer to schemes by Brands as U-Prove since basically, they are the main building blocks in Microsoft's U-Prove technology [16].

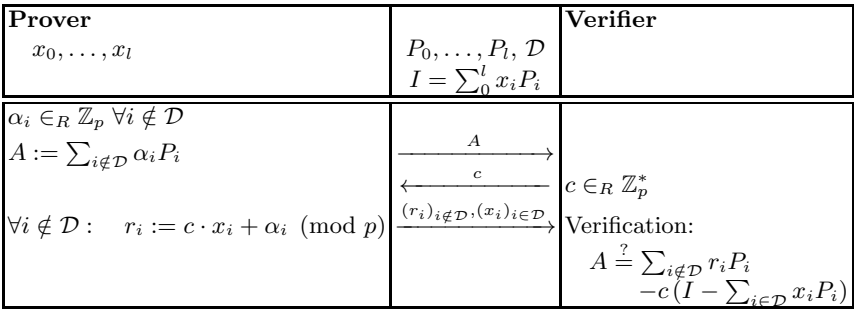


Fig. 2. Selective disclosure protocol in [4] where attributes (committed values) in $\mathcal{D}$ are disclosed, while for all the others only a proof of knowledge is given. Here l is the number of attributes; d is the size of the disclosure set $\mathcal{D}$, and I is the commitment following the identification notation. (There are $l - d + 1$ point multiplications on the Prover's side.)

only the verifier can verify the proof and compute the identifier I. Finally, the verifier checks I in his database that stores all valid tag identifiers. Note that, unlike in the Schnorr proof [18] or the U-Prove showing protocol [4] in which the identifier is a common input value and it is confirmed by the verification equation, only the verifier learns the identifier in this scheme.

Firstly, we show that the Randomized Schnorr scheme can be generalized in a natural way resulting in a secure and narrow-strong designated verifier DL-REP proof. Secondly, we introduce the designated selective disclosure, i.e., a protocol that allows for a Designated Verifier Partial DL-REP proof.

3.1 Designated Verifier DL-REP Proof

Setup

- $\mathsf{SetupSystem}(1^k) \longrightarrow par$ outputs parameters par with the group description and the base points $P_0, \ldots, P_l$.
- $\mathsf{SetupVerifier}(par) \longrightarrow (v, V)$ generates a private/public key pair for the Verifier, where the public key $V = v \cdot \sum_0^l P_i$. If the key pair has already been generated in the system, the algorithm outputs that.
- $\mathsf{SetupTag}(par) \longrightarrow ((x_0, \ldots, x_l), I)$ generates attributes $(x_0, \ldots, x_l)$ and an identifier I for a tag, where the identifier $I = \sum_0^l x_i P_i$.

Protocol. The Designated Verifier DL-REP proof in Figure 3 is clearly correct as the value computed by the verifier in the last step will be always equal to the prover's identifier I. Furthermore, the proof is zero-knowledge since, given I, the verifier herself could generate a valid transcript by selecting $r_0, \ldots, r_l$ and c uniformly at random from $\mathbb{Z}_p$ and A_2 uniformly at random from G. Then $A_1 := \sum_0^l r_i P_i - cI - v^{-1} A_2$ will be distributed uniformly in G.

Prover		**Verifier**
$x_0, \ldots, x_l$ $I = \sum_0^l x_i P_i$	$P_0, \ldots, P_l$ $V = v \cdot \sum_0^l P_i$	v
$\alpha_0, \ldots, \alpha_l, \beta \in_R \mathbb{Z}_p$ $A_1 := \sum_0^l \alpha_i P_i$ $A_2 := \beta V$ $\forall i \in 0, \ldots, l:$ $r_i := c \cdot x_i + \alpha_i + \beta \pmod p$	$\xrightarrow{A_1, A_2}$ $\xleftarrow{c}$ $\xrightarrow{r_0, \ldots, r_l}$	$c \in_R \mathbb{Z}_p^*$ Verification: $I := c^{-1}(\sum_0^l r_i P_i - A_1 - v^{-1} A_2)$ check whether I is a valid identifier

Fig. 3. Designated verifier DL-REP proof; i.e., proof of knowledge of a DL-REP of I w.r.t. $P_0, \ldots, P_l$ (There are $l + 2$ point multiplications on the Prover's side.)

To use the full potential of DL-REPs, we want to make selective disclosure proofs, that is, a scheme in which a prover should be able to prove the knowledge of any subset of attributes. In U-Prove, the revealed attributes are either common input, or they are sent through a private channel to the verifier. While the former releases information to an external party, the latter presumes some encryption with the verifier's key. Since neither of these solutions is suitable in an RFID set-up, we should extend designated verification to include selective disclosure.

A naive approach to selective disclosure is the following. The prover proves the knowledge of a reduced set of attributes (e.g., without attribute x_2) which would enable the verifier to compute a partial identifier. Adding to it possible attribute points (e.g., $x_2 P_2$) by trial and error. The verifier then tries all possible attribute points until it obtains a valid identifier. However, this solution clearly does not scale for several attributes with a lot of possible values.

In the next section we extend the current scheme to a designated selective disclosure scheme that does not have the drawback mentioned above. The security and narrow-strong privacy for this designated DL-REP scheme will follow from the corresponding results for the scheme in the next section.

3.2 Designated Selective Disclosure

In the previous section we introduced a designated zero-knowledge proof of knowledge of a DL-REP. The prover tag does not reveal its attributes, only the fact that it actually knows them. To make the construction more practical, we propose another scheme, the Designated Verifier Partial DL-REP scheme, or simply designated selective disclosure.

In this scheme a verifier can compute and check the identifier of a tag. Furthermore, it can compute an attribute points[5] only if the prover disclosed it and

[5] Unlike in traditional selective disclosure, not the actual attributes x_j but the corresponding points $x_j P_j$ are disclosed. However, note that the proof includes the fact that the tag stores attributes x_j.

the verifier is entitled to see it according to a so-called entitlement set $\mathcal{E}$ (by which we mean a set of indices that defines which attributes a verifier is entitled to see). Note that even if the verifier does not have all designated attribute private keys v_i, he can compute the identifier I and those attributes he is entitled to see, as determined by $\mathcal{D} \cap \mathcal{E}$.

Setup. The algorithm SetupVerifier in the Setup is slightly modified because of the designated attribute verification.

- $\mathsf{SetupVerifier}(par, \mathcal{E}) \longrightarrow (v, V, (v_i, V_i)_{\mathcal{E}})$ generates a private/public key pair and a set of pairs for the Verifier, where the latter set depends on the entitlement index set $\mathcal{E}$. If the identification key and the entitlement keys have already been generated in the system, the algorithm outputs those.

Protocol. Assume that a tag is interrogated to reveal a set of attributes corresponding to the disclosure index set $\mathcal{D}$ (see Figure 4). Then it has to perform an identification in which the designated verifier, who is entitled to read attributes in $\mathcal{E}$, can compute the following values:

- identifier I of the prover tag;
- disclosed attribute points $C_i = x_i P_i$ that were disclosed by the prover and for which the verifier has the corresponding attribute verifier key v_i.

After generating random values for all attributes and for the designation, the prover can compute the commitment points A_1, A_2 for the DL-REP and $(B_i)_{i \in \mathcal{D}}$ for the designated selective disclosure. Following the challenge–response phase, the verifier can first compute identifier I like in the normal Designated Verifier DL-REP scheme, that is, without the use of the entitlement set $\mathcal{E}$. Second, the verifier can reconstruct attribute points $C_j = x_j P_j$ in case $j \in \mathcal{D} \cap \mathcal{E}$, that is, both the prover included B_j in the proof and the verifier is entitled to see the attribute point of index j. We note that the entitlement set $\mathcal{E}$ can be empty.

Security against Impersonation. We show that it is not possible for an adversary to impersonate any valid tag, even though it was allowed to communicate with valid tags before.

Theorem 1. *Assuming the original Selective Disclosure U-Prove scheme is secure against active impersonation attacks the Designated Verifier Partial DL-REP proof is also secure against active impersonation attacks.*

Proof (Sketch.). We show how an adversary against the Designated Verifier Partial DL-REP system can be used to break the security of the U-Prove Selective Disclosure scheme. To do so, we build an adversary $\mathcal{B}$ that essentially translates between these two systems. For the disclosed attributes we can easily 'undisclose' them to mimic the Designated Verifier Partial DL-REP scheme. To go back we simply remember which attribute value corresponds to which public value. The other direction is the same as in Bringer et al. [6].

Prover		**Verifier**
$x_0, \ldots, x_l$	$P_0, \ldots, P_l$	$v, (v_i)_{i \in \mathcal{E}}$
$I = \sum_0^l x_i P_i$	$\forall i \in \mathcal{D}: V_i = v_i P_i$ $V = v \cdot \sum_0^l P_i$	
$\alpha_0, \ldots, \alpha_l, \beta \in_R \mathbb{Z}_q^*$ $A_1 := \sum_0^l \alpha_i P_i$ $A_2 := \beta V$		
$B_i = (\alpha_i + \beta) V_i \quad \forall i \in \mathcal{D}$	$\xrightarrow{A_1, A_2, (B_i)_{i \in \mathcal{D}}}$	
	$\xleftarrow{c}$	$c \in_R \mathbb{Z}_q^*$
$\forall i \in 0, \ldots, l:$ $r_i := c \cdot x_i + \alpha_i + \beta \pmod p$	$\xrightarrow{r_0 \ldots r_l}$	First verify that the identifier is correct: $I = c^{-1}(\sum_0^l r_i P_i - A_1 - v^{-1} A_2)$ Then for each $j \in \mathcal{D} \cap \mathcal{E}$ compute attribute C_j: $C_j = I - c^{-1}(\sum_{i \neq j} r_i P_i - A_1 - v^{-1} A_2 + v_j^{-1} B_j)$

Fig. 4. Designated verifier DL-REP proof in which attributes in $\mathcal{D}$ are disclosed. (There are $l + 2 + d$ point multiplications on the Prover's side.)

Narrow-Strong Privacy. This proof uses a somewhat similar approach as the privacy proof of the Randomized Schnorr scheme [6]. There the authors show that the game in Definition 4 can be reduced to the following. If an adversary breaks the narrow-strong privacy of their Randomized Schnorr scheme, then it has to be able to distinguish tuples of the form $(A_1 = \alpha P, A_2 = \beta P, r = \alpha + \beta)$, where α and β are random from tuples of the form $(A_1 = \alpha P, A_2 = \beta P, r)$, where also r is random. Furthermore, they show that any adversary that can do so can be used to break DDH.

Theorem 2. *Assuming the hardness of the DDH-problem the Designated Verifier Partial DL-REP scheme is narrow-strong private.*

Proof (Sketch.). We extend traces for the Randomized Schnorr scheme to full traces for the Designated verifier DL-REP scheme. We do this in such a way that the new responses are random if and only if the response of the original instance was random. Hence any adversary against the Designated Verifier DL-REP scheme can be converted into a Randomized Schnorr adversary. Since the latter is secure under the DDH-assumption, the result follows.

4 Feasibility and Applications

In this section we discuss practical implications of our proposal. First we describe possible implementations and we follow up with some applications.

4.1 Feasibility of Our Proposal

To show the feasibility of the proposed protocols for RFID tags, we consider an ECC-based architecture, for example, the one presented in [2]. The EC processor described is very compact and the performance of 1 point multiplication, even when frequency is lowered enough to keep the total power low, is still acceptable. More precisely, the ECC-based grouping proofs as in [2] require two or three point multiplications and, even in the latter case, running time to complete the proof should stay below 300 ms. In addition our selective disclosure protocol achieves similar performance as hierarchical proofs [1] in which the performance depends on the number of levels in the hierarchy.

Similar remarks are valid for the memory requirements of a single tag. Assuming l attributes, a tag has to store $l+1$ values where each is 160 bits long as the group keys in the hierarchical proof protocols. Having, for instance, 4 attributes to store, a tag requires 800 bits memory (assuming a curve over a 160-bit field). This is completely acceptable even for passive tags as attributes could be stored in the ROM memory, which is (unlike registers) considered very cheap, in the same way as the ECC parameters.

4.2 Envisioned RFID Applications

As mentioned above, new RFID security applications requiring a strong level of privacy are emerging constantly. Examples from previous works include yoking (or grouping) proofs and hierarchical proofs. Hence, an immediate need for designated attribute-based proofs is clear.

Considering the example of hierarchical proofs, our solution could be deployed meeting exactly the same requirements as envisioned by the tree structure of the hierarchical proofs [1]. To obtain the same functionality, one could sort the attributes according to their order of importance. More precisely, choose x_1 to be less important i.e. less privacy/security critical and therefore, the first secret verification key v_1 can be stored on a lot of readers, while v_3, for example, only at a very limited set of verifiers, etc. This infrastructure is easily incorporated in the designated attribute-based proofs as introduced above. In this way, we achieve not just a more fine-grained access control for tags, but also more fine-grained permissions for readers.

A typical real-life scenario can be found in the medical domain. Patients carry medicines that can be scanned (and sometimes should) by legitimate authorities (e.g., customs officers) while maintaining some privacy for the user. In this case, the highest level of verification, i.e., the lowest index is left for medical staff providing first aid in accidents or other emergency cases.

5 Conclusions

We proposed a new scheme, the Designated Verifier (Partial) DL-REP proof: a tag, storing a DL-REP of its unique identifier, can reveal an arbitrary subset of its

attributes to a designated verifier. This scheme relies on recent designs of RFID chips that allows for the use of elliptic curve cryptography. While any authentic verifier can check the tag's validity, it can only compute those attribute points that it is entitled to. On the one hand, a tag can contain many semantically different attributes, a reader, on the other hand, can gain access only certain subset of these.

We proved that the protocol is secure and narrow-strong private in the Vaudenay model. Therefore, the scheme is powerful and reliable and it enables further architectural developments in which tags and verifiers can have fine-grained permissions.

In Section 4.2, we show that the scheme enables the development of new protocols for specific applications. Nevertheless, in the context of RFID systems further study is needed to examine whether extentions to the attribute-based proofs, such as proving predicates and linear dependencies among attributes, or verifying more tags at the same time, can be applied in a meaningful manner.

5.1 Future Work

An actual implementation of the protocol offers new opportunities for research. It allows us to obtain results (in terms of timing, power consumption, area, memory, code size, battery time) and to test applicability of the protocols on RFID tags. Moreover, an implementation on other mobile devices, such as smart cards or mobile phones can offer interesting results as well.

We are aware that the protocols proposed in this paper are computationally demanding for most RFID systems. We believe, however, that RFID applications can be designed that are tailored and simplified to the specific hardware and yet they preserve required security and privacy properties of our schemes.

Our schemes could be deployed in privacy-sensitive contexts, such as electronic health records. Patients' physical characteristics, permanent and temporary conditions and their medication can be stored in credentials, and revealed only in circumstances and to recipients only if it is really necessary. Furthermore, given more expensive user devices, computational problems emerge to a smaller extent than with RFID tags.

Acknowledgements. This work was supported in part by the research program Sentinels[6] as projects *Mobile IDM* (10522) and *Revocable Privacy* (10532) and by the European Commission under contract number ICT-2007-216676 ECRYPT NoE phase II.

[6] Sentinels is being financed by Technology Foundation STW, the Netherlands Organization for Scientific Research (NWO), and the Dutch Ministry of Economic Affairs.

References

1. Batina, L., Lee, Y.K., Seys, S., Singelée, D., Verbauwhede, I.: Privacy-Preserving ECC-Based Grouping Proofs for RFID. In: Burmester, M., Tsudik, G., Magliveras, S., Ilić, I. (eds.) ISC 2010. LNCS, vol. 6531, pp. 159–165. Springer, Heidelberg (2011)
2. Batina, L., Lee, Y.K., Seys, S., Singelée, D., Verbauwhede, I.: Extending ECC-based RFID authentication protocols to privacy-preserving multi-party grouping proofs. Personal and Ubiquitous Computing 16(3), 323–335 (2012)
3. Batina, L., Seys, S., Singelée, D., Verbauwhede, I.: Hierarchical ECC-Based RFID Authentication Protocol. In: Juels, A., Paar, C. (eds.) RFIDSec 2011. LNCS, vol. 7055, pp. 183–201. Springer, Heidelberg (2012)
4. Brands, S.A.: Rethinking Public Key Infrastructures and Digital Certificates: Building in Privacy. MIT Press, Cambridge (2000)
5. Braun, M., Hess, E., Meyer, B.: Using Elliptic Curves on RFID Tags. IJCSNS International Journal of Computer Science and Network Security 8(2), 1–9 (2008)
6. Bringer, J., Chabanne, H., Icart, T.: Cryptanalysis of EC-RAC, a RFID Identification Protocol. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 149–161. Springer, Heidelberg (2008)
7. Camenisch, J., Krontiris, I., Lehmann, A., Neven, G., Paquin, C., Rannenberg, K., Harald, Z.: D2.1 Architecture for Attribute-based Credential Technologies. Deliverable, ABC4Trust EU Project (December 2011)
8. Chaum, D.: Zero-Knowledge Undeniable Signatures (extended abstract). In: Damgård, I.B. (ed.) EUROCRYPT 1990. LNCS, vol. 473, pp. 458–464. Springer, Heidelberg (1991)
9. Chaum, D., van Antwerpen, H.: Undeniable Signatures. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 212–216. Springer, Heidelberg (1990)
10. Fan, J., Knezevic, M., Karaklajic, D., Maes, R., Rozic, V., Batina, L., Verbauwhede, I.: FPGA-based testing strategy for cryptographic chips: A case study on Elliptic Curve Processor for RFID tags. In: 15th IEEE International On-Line Testing Symposium (IOLTS 2009), Sesimbra-Lisbon, Portugal, June 24-26, pp. 189–191. IEEE (2009)
11. Hein, D., Wolkerstorfer, J., Felber, N.: ECC Is Ready for RFID – A Proof in Silicon. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 401–413. Springer, Heidelberg (2009)
12. Jakobsson, M., Sako, K., Impagliazzo, R.: Designated Verifier Proofs and Their Applications. In: Maurer, U.M. (ed.) EUROCRYPT 1996. LNCS, vol. 1070, pp. 143–154. Springer, Heidelberg (1996)
13. Juels, A.: "Yoking-Proofs" for RFID Tags. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW 2004), pp. 138–143. IEEE Computer Society (2004)
14. Koblitz, N.: Elliptic curve cryptosystems. Mathematics of Computation 48, 203–209 (1987)
15. Lee, Y.K., Batina, L., Singelée, D., Verbauwhede, I.: Low-Cost Untraceable Authentication Protocols for RFID. In: Proceedings of the Third ACM Conference on Wireless Network Security, WiSec 2010, pp. 55–64. ACM, New York (2010)
16. Paquin, C.: U-Prove Cryptographic Specification V1.1. Technical report, Microsoft (2011)

17. Saeednia, S., Kremer, S., Markowitch, O.: An Efficient Strong Designated Verifier Signature Scheme. In: Lim, J.-I., Lee, D.-H. (eds.) ICISC 2003. LNCS, vol. 2971, pp. 40–54. Springer, Heidelberg (2004)
18. Schnorr, C.-P.: Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
19. Vaudenay, S.: On Privacy Models for RFID. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 68–87. Springer, Heidelberg (2007)

A Security and Privacy Proofs

A.1 Proof of Theorem 1

Proof. (Theorem 1.) Suppose we are given an adversary $\mathcal{A}$ that wins the active impersonation game for the Designated Verifier DL-REP proof, we show how to construct an adversary $\mathcal{B}$ that wins the active impersonation game for the U-Prove selective disclosure scheme. We need to show how $\mathcal{B}$ answers the identification requests from $\mathcal{A}$ using only the U-Prove oracle. Furthermore, we show how the impersonated identification protocol, run by $\mathcal{A}$ against the Designated Verifier DL-REP scheme, is converted by $\mathcal{B}$ into an identification protocol for the U-Prove selective disclosure scheme.

Initially, adversary $\mathcal{B}$ generates a random private key v and sets the public key V to $V = v\sum P_i$. Furthermore, for any disclosed attribute $i \in \mathcal{D}$ adversary $\mathcal{B}$ generates v_i at random and sets $V_i = v_i P_i$, and sends these V_i's together with V to adversary $\mathcal{A}$.

During the first phase $\mathcal{B}$ answers interrogation queries for a tag as follows. First, it queries its own oracle, who sends a commitment A. Adversary $\mathcal{B}$ generates $\alpha_i \in_R \mathbb{Z}_p$ for $i \in \mathcal{D}$ and $\beta \in_R \mathbb{Z}_p$ and sends to $\mathcal{A}$ the values

$$\begin{aligned} A_1 &= A + \sum_{i\in\mathcal{D}} \alpha_i P_i \\ A_2 &= \beta V \\ B_i &= (\alpha_i + \beta)V_i \qquad i \in \mathcal{D}. \end{aligned}$$

Subsequently, $\mathcal{B}$ receives c from $\mathcal{A}$ which it passes along to its oracle. In return it receives r'_i for $i \notin \mathcal{D}$ and x_i for $i \in \mathcal{D}$. It then sends to $\mathcal{A}$ the responses

$$r_i = \begin{cases} r'_i & \text{for } i \notin \mathcal{D} \\ cx_i + \alpha_i + \beta \pmod p & \text{for } i \in \mathcal{D}. \end{cases}$$

For future reference $\mathcal{B}$ will store the tuples $(x_i, x_i P_i)$ for every disclosed attribute. Clearly, this construction is a perfect simulation of the designated verification protocol.

In the second phase adversary $\mathcal{A}$ will impersonate a tag. The goal of adversary $\mathcal{B}$ is to transform this communication such that it in turn impersonates a valid tag for the U-Prove selective disclosure protocol. First, $\mathcal{A}$ will generate two

commitments A_1 and A_2, which are converted by $\mathcal{B}$ into $A = A_1 + v^{-1}A_2$ before sending it to the original U-Prove verifier. The verifier responds with a challenge c, which $\mathcal{B}$ relays unchanged to $\mathcal{A}$. Finally, $\mathcal{A}$ replies with the r_i values. For $i \notin \mathcal{D}$, $\mathcal{B}$ forwards these values to the challenger. Note that they should equal $r_i = cx_i + \alpha_i + \beta$ and are therefore appropriate responses to the commitment A. For the disclosed attributes ($i \in \mathcal{D}$), $\mathcal{B}$ can calculate

$$x_i P_i = I - c^{-1}(\sum_{j \neq i} r_j P_j - A_1 - v^{-1}A_2 + v_i^{-1}B_i).$$

Using its stored tuples, $\mathcal{B}$ can then recover the values x_i before forwarding them to the challenger according to the U-Prove protocol (see Figure 2). This completes the proof.

A.2 Proof of Theorem 2

Proof. (Theorem 2) A transcript in our designated verifier partial DL-REP proof has the form $A_1 = \sum \alpha_i P_i, A_2 = \beta V, (B_i)_{i \in \mathcal{D}}, c, (r_i = cx_i + \alpha_i + \beta)_{i=0}^{l}$. We would like to show that the adversary cannot distinguish between properly constructed r_i's and randomly chosen ones. Following the argument in Bringer et al. [6], we can take out the attribute values x_i. Hence, the adversary has to distinguish instances from the actual distribution

$$D_A^l = \{(A_1^S = \sum_{i=0}^{l} \alpha_i P_i, A_2^S = \beta V, (B_i)_{i \in \mathcal{D}}, (r_i = \alpha_i + \beta)) : \\ \alpha_i, \beta \in_R \mathbb{Z}_p, 0 \leq i \leq l\}$$

from instances from the simulated distribution

$$D_S^l = \{(A_1^S = \sum_{i=0}^{l} \alpha_i P_i, A_2^S = \beta V, (B_i)_{i \in \mathcal{D}}, (r_i)) : \alpha_i, \beta, r_i \in_R \mathbb{Z}_p, 0 \leq i \leq l\}$$

where the r_i's are random.

Suppose we have an oracle for distinguishing between these two distributions, we will use this to decide between the corresponding instances for the Randomized Schnorr scheme, which are in fact instances from D_A^0 or D_S^0. The main idea is that we use the instance $(A_1 := \alpha P, A_2 := \beta V, r)$ we obtain as a challenge, to construct a full instance. This instance can then be solved using our oracle. We construct the other attributes in such a way that $\alpha_0 = \alpha$ and $\alpha_i = \alpha + \gamma_i$ where γ_i is random.

Start by setting $P_0 = P$ and $P_i = p_i P$, with p_i random. Then we construct A_1^D as

$$A_1^D = A_1 + \sum_{i=1}^{l} (p_i A_1 + \gamma_i P_i) = \alpha P_0 + \sum_{i=1}^{l} [(\alpha + \gamma_i) P_i] .$$

Similarly, we set $V = \sum_{i=0}^{l} P_i$, with v_i random and construct

$$A_2^D = A_2 + \sum_{i=1}^{l} p_i A_2 = \beta \sum_{i=0}^{l} P_i.$$

For any disclosed attribute $i \in \mathcal{D}$ choose $v_i \in_R \mathbb{Z}_p$ and set $V_i = v_i P_i$. Then the values B_i are constructed as

$$B_i = v_i(p_i A_1 + \gamma_i P_i) = (\alpha + \gamma_i) V_i.$$

Finally, we set $r_0 = r$ and

$$r_i = r + \gamma_i.$$

If $r = \alpha + \beta$, then clearly all other r_i's are correct as well, and we are in the normal situation. However, if r is random, then all the other values are random as well.[7] This construction yields a valid input to our Designated Partial DL-REP oracle, and can hence be used to break the privacy of the Randomized Schnorr scheme.

[7] While it may appear that the γ_i's are fixed by the construction of A_1^D, this is actually not the case: Nothing binds the value of α itself anymore, and hence, A_1^D is actually a commitment to the γ_i's that hides information theoretically.

T-MATCH: Privacy-Preserving Item Matching for Storage-Only RFID Tags

Kaoutar Elkhiyaoui[1], Erik-Oliver Blass[2], and Refik Molva[1]

[1] Eurecom, Sophia-Antipolis, France
{kaoutar.elkhiyaoui,refik.molva}@eurecom.fr
[2] College of Computer and Information Science,
Northeastern University, Boston, MA 02115
blass@ccs.neu.edu

Abstract. RFID-based tag matching allows a reader R_k to determine whether two tags T_i and T_j store some attributes that jointly fulfill a boolean constraint. The challenge in designing a matching mechanism is tag privacy. While cheap tags are unable to perform any computation, matching has to be achieved without revealing the tags' attributes. In this paper, we present T-MATCH, a protocol for secure and privacy preserving RFID tag matching. T-MATCH involves a pair of tags T_i and T_j, a reader R_k, and a backend server S. To ensure tag privacy against R_k and S, T-MATCH employs a new technique based on secure two-party computation that prevents R_k and S from disclosing tag attributes. For tag privacy against eavesdroppers, each tag T_i in T-MATCH stores an IND-CPA encryption of its attribute. Such an encryption allows R_k to update the state of T_i by merely re-encrypting T_i's ciphertext. T-MATCH targets cheap tags that cannot perform any computation, but are only required to store 150 bytes.

1 Introduction

One prominent application of RFID technology is the automation of safety inspections when transporting hazardous goods such as highly reactive chemicals in supply chains. Here, it is dangerous to place specific, reactive chemicals close to each other, because small leaks can already result in a threat to the life of workers managing these chemicals.

Some recent solutions to enforce safety regulations when storing or transporting chemicals in supply chains rely on equipping each chemical container with an RFID tag that stores information for identifying the chemical in the container as highlighted by the EU project CoBIs [5]. Before two tags are placed next to each other, their tags are wirelessly "scanned" using an RFID reader. Each tag sends its content in cleartext to a server. The server performs chemicals' matching based on a set Ref of *matching references* that it knows beforehand. Each matching reference identifies a pair of chemicals that react. Now, when two reactive chemicals are detected, the server triggers an alarm.

However, the above solution suffers from several shortcomings that may lead to security and privacy threats. The fact that tags transmit their contents in

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 76–95, 2013.

cleartext allows any malicious entity with proper wireless equipment to learn the content of a container, to infer information about reactive chemicals, and finally to track their location.

Consequently, RFID-based protocols for tag matching require a careful design taking into account both the security and the privacy threats to RFID tags and the consequences thereof on the security and safety of users managing matched items.

A privacy preserving RFID-based tag matching must assure that tag matching is performed without disclosing the content of tags. That is, the only information revealed after executing the protocol to *readers in the supply chain* is a bit b indicating whether the tags involved in the protocol execution "match" or not. It must also ensure *location privacy* so as to prevent tracking attacks by eavesdroppers. Ideally, an eavesdropper must not be able to distinguish between tags based on the traces of the matching protocol, in accordance with previous work, this requirement will be called hereafter tag unlinkability.

With respect to security, it is mandatory to ensure that a matching protocol is correct (almost) all the time. Namely, it is required to detect all incompatible items (reactive chemicals). This corresponds to a *completeness* property: the protocol must always trigger an alarm when two reactive chemicals are put next to each other. Moreover, the protocol has to be efficient: an alarm is triggered only when necessary. When a match is detected by the protocol, one can safely derive that the tags involved in the protocol are attached to reactive chemicals. This second requirement corresponds to the *soundness* property of the protocol.

Note that solutions to answer the above security and privacy problems are strongly constrained by the limitations of RFID environment. While tag privacy against eavesdroppers can be achieved by using re-encryption techniques, tag privacy against readers is more difficult to address especially when using cheap RFID read/write only tags unable to perform any computation. Traditional security and privacy solutions based on heavyweight secret matching protocols between two parties , cf. Ateniese et al. [1], Balfanz et al. [3], cannot be implemented in an RFID setting.

Accordingly, we design T-MATCH, a new tag matching protocol that involves tags T_i attached to "containers" (barrels) of chemicals traveling in a supply chain, multiple readers R_k and a back-end server S. T-MATCH targets read/write only tags only featuring storage and no computational capabilities so as to allow for the deployment of such an application with a reasonable cost.

Overview: In T-MATCH, a reader R_k in the supply chain reads out the content of a pair of tags T_i and T_j, cooperates with back-end server S to perform tag matching, and finally outputs the outcome of matching while assuring various privacy properties in the face of curious readers R_k and curious backend server S.

Readers R_k and backend server S are required to evaluate securely a *boolean* function CHECK for any pair of tags T_i and T_j, such that CHECK outputs $b = 1$, if T_i and T_j match. To this effect, each tag T_i in T-MATCH stores a homomorphic IND-CPA encryption Enc of its attribute a_{T_i}. When two tags T_i and T_j are in the range of a reader R_k, reader R_k reads both tags and retrieves the encryptions

$\mathsf{Enc}(a_{T_i})$ and $\mathsf{Enc}(a_{T_j})$ of T_i and T_j's attributes respectively. To protect the privacy of tags, reader R_k re-encrypts the ciphertexts stored into tags T_i and T_j. Now, to evaluate the CHECK function, reader R_k uses the homomorphic property of Enc to compute an encryption $\mathsf{Enc}(f(a_{T_i}, a_{T_j}))$ of a function f of T_i and T_j's attributes. Then, reader R_k and backend server S engage in two party protocol for a *modified* privacy preserving plaintext equality test [9] to check whether $f(a_{T_i}, a_{T_j}) \in \mathsf{Ref}$, where Ref is the set of matching references of backend server S. If so, CHECK outputs $b = 1$; otherwise, CHECK outputs $b = 0$.

To summarize, T-MATCH's major contributions are:

- T-MATCH proposes a novel solution for item matching that targets read/write only tags. A tag T_i in T-MATCH does not perform any computation, it is only required to store a state that is updated at every protocol execution by readers R_k.
- T-MATCH is provably privacy preserving: T-MATCH relies on techniques of secure two-party computation to ensure that neither readers R_k nor backend server S can disclose the content of a tag or learn its attribute.
- T-MATCH is provably secure: readers R_k raise an alarm only when they interact with a pair of matching tags.

2 Preliminaries

In this section, we introduce T-MATCH's problem statement and T-MATCH's entities.

2.1 Problem Statement

A read/write only tag T_i in T-MATCH stores a state that encodes its attribute a_{T_i}. By solely relying on the states of any pair of tags T_i and T_j, a reader R_k in the supply chain has to decide whether tags T_i and T_j match or not.

A first solution to tackle this problem could be encrypting the state of tags. When two tags T_i and T_j are in the range of an authorized reader R_k, reader R_k decrypts the content of tags T_i and T_j. Finally, based on a set of matching references Ref, reader R_k decides whether T_i and T_j match or not.

However, the solution above has two limitations: **first,** if the underlying encryption is not probabilistic, tags will be sending the same ciphertexts whenever queried. This enables any eavesdropper to track tags, and consequently, enables eavesdroppers to violate tag unlinkability. **Second,** it does not ensure tag privacy against readers R_k. The solution relies on disclosing the tags' attributes to readers R_k in the supply chain.

Although the first limitation can be tackled by using probabilistic encryption, the second limitation is difficult to address, as tags cannot perform any computation.

We recall that our main goal is to enable reader R_k to perform tag matching for any pair of tags T_i and T_j while preserving the privacy of tags. That is, at the end of the matching protocol, *a reader R_k only gets the outcome of a boolean function* CHECK *which outputs a bit $b = 1$ if tags T_i and T_j match, otherwise, it outputs $b = 0$.*

A straightforward solution to address the problem above is to use homomorphic encryption. Homomorphic encryption enables readers R_k to compute the encrypted value $\mathsf{Enc}(\textsc{Check}(T_i, T_j))$ using the encrypted value $\mathsf{Enc}(a_{T_i})$ of attribute a_{T_i} stored in tag T_i and the encrypted value $\mathsf{Enc}(a_{T_j})$ of attribute a_{T_j} stored in tag T_j.

However, a limitation of this approach arises when we allow readers to decrypt the ciphertext $\mathsf{Enc}(\textsc{Check}(T_i, T_j))$: if a reader R_k is allowed to decrypt $\mathsf{Enc}(\textsc{Check}(T_i, T_j))$, then by the same means, it can decrypt $\mathsf{Enc}(a_{T_i})$ and $\mathsf{Enc}(a_{T_j})$, leading to the potential disclosure of the tag attributes to readers in the supply chain.

An idea to overcome this limitation consists of preventing readers from decrypting ciphertexts by themselves. This calls for the use of secret sharing techniques [15]. We identify two methods to implement secret sharing: the **first** method relies on distributing secret shares to readers and tags. The idea would be to allow a reader R_k to decrypt only when it reads a pair of tags T_i and T_j that match. However, such a solution requires that tags T_i in the system are either active and able to perform cryptographic operations, or synchronized by readers. The **second** method relies on an additional third-party component that is a backend server S. S possesses the set Ref of matching references. Readers and backend server S hold secret shares of some secret key sk that allows backend server S and any reader R_k to evaluate securely $\textsc{Check}(T_i, T_j)$.

T-MATCH relies on the second method to implement item matching. That is, in addition to readers R_k which read and update the content of tags, T-MATCH involves a backend server S that stores the set Ref of matching references for any pair of attributes that match. Although, this approach requires backend server S to be always online with readers R_k, it remains realistic. We stress that today, even handheld RFID readers can establish continuous connection with backend server S using wireless technologies such as Bluetooth, ZigBee, WiFi or even GSM. Furthermore, having a backend server S allows for using techniques of secure multi-party computation to ensure that at the end of an execution of T-MATCH, readers R_k and backend server S learn at most the output of the CHECK function .

Now, to check whether a pair of tags T_i and T_j match, a reader R_k reads first the encrypted states stored into T_i and T_j, then R_k contacts backend server S in order to securely evaluate the CHECK function for T_i and T_j. The CHECK function has as input the encrypted states of tags T_i and T_j along with the matching references Ref of backend server S. At the end of a T-MATCH's execution, reader R_k gets the output of the CHECK function.

2.2 T-MATCH's Setup

T-MATCH involves the following entities:

- **Tags T_i:** Each tag is attached to an item (container, barrel, ...). A tag T_i is equipped with a re-writable memory storing T_i's current "state" denoted $S^j_{T_i}$. The state $S^j_{T_i}$ encodes and *encrypts* an attribute $a_{T_i} \in \mathbb{A}$, where $\mathbb{A}$ is the set of valid attributes in T-MATCH. We denote $\mathcal{T}$ the set of tags in T-MATCH, and we assume that $|\mathbb{A}| = l$ and $|\mathcal{T}| = n$.
- **Issuer I:** The issuer I initializes tags. It chooses an attribute $a_{T_i} \in \mathbb{A}$, then computes an initial state $S^0_{T_i}$, and finally writes the state $S^0_{T_i}$ into T_i.
- **Readers R_k:** A reader R_k in the supply chain interacts with tags T_i in its vicinity. R_k reads the states $S^{k_i}_{T_i}$ and $S^{k_j}_{T_j}$ stored into tags T_i and T_j respectively by calling the function READ, and updates the states $S^{k_i}_{T_i}$ and $S^{k_j}_{T_j}$ accordingly. Next, R_k writes the new states $S^{k_i+1}_{T_i}$ and $S^{k_j+1}_{T_j}$ into T_i and T_j by calling the function WRITE. Finally, R_k engages in a *two party protocol* with backend server S to compute securely a boolean function CHECK. R_k's input to CHECK is the states $S^{k_i}_{T_i}$, $S^{k_j}_{T_j}$, and its secret share α_{R_k}. If CHECK outputs $b = 1$, then reader R_k raises an alarm meaning that T_i and T_j match. Otherwise, T_i and T_j do not match and reader R_k does nothing. Without loss of generality, we assume that the supply chain comprises η readers R_k.
- **Backend server S:** Backend server S stores a set of ν matching references $\mathsf{Ref} = \{\mathsf{ref}_1, \mathsf{ref}_2, ..., \mathsf{ref}_\nu\}$. Backend server S is required to compute a boolean function CHECK jointly with reader R_k. The input of Backend server S to the CHECK function is its set of matching references Ref and its secret share α_S.

3 Adversary Models

We recall that in secure multiparty computation protocols, two adversary models are identified: *semi-honest* and *malicious* in compliance with the work of Goldreich [6].

- ***Semi-honest model*:** Readers R_k and backend server S are assumed to act according to the protocol with the exception that each party keeps a record of all its computations.
- ***Malicious model*:** An adversary $\mathcal{A} \in \{R_k, S\}$ in this model may act arbitrarily. Adversary $\mathcal{A}$ may **i.)** refuse to participate in the protocol when the protocol is first invoked. $\mathcal{A}$ may as well **ii.)** substitute its local input: this corresponds for instance to a reader R_k providing an input that does not match the states of tags it has just read, or to backend server S submitting a set of bogus matching references as its local input. $\mathcal{A}$ may also **iii.)** abort the protocol before sending its last message.

In the remainder of this paper, we focus on semi-honest adversaries as we believe that in the real world, it is hard for readers R_k and backend server S to deviate

from the protocol arbitrarily without being detected. Note that it is always feasible to verify whether a reader R_k raises an alarm when it should or not. Whereas it is hard to prevent readers R_k and backend server S from keeping records of their previous protocol executions or from eavesdropping on tags in the system.

Accordingly, we assume that readers R_k and backend server S are semi-honest (i.e., they behave in compliance with T-MATCH) and they do not collude against tags in the supply chain. We assume as well that issuer I is honest, meaning that when I initializes a tag, then this tag correctly encodes the attribute of the item to which it is attached.

Now, to formally capture the capabilities of an adversary $\mathcal{A}$ against the security and the privacy of T-MATCH, a challenger $\mathcal{C}$ provides adversary $\mathcal{A}$ with access to the following oracles:

- $\mathcal{O}_{\mathsf{Tag}}(\mathsf{param})$: When queried with a parameter param, the oracle $\mathcal{O}_{\mathsf{Tag}}(\mathsf{param})$ returns a tag based on the value of the parameter chosen by $\mathcal{A}$. For instance, if $\mathsf{param} = a_i \in \mathbb{A}$, then $\mathcal{O}_{\mathsf{Tag}}$ returns a tag that encodes attribute a_i.
- $\mathcal{O}_{\mathsf{Check}}(T_i, T_j)$: When queried with a pair of tags T_i and T_j, the oracle $\mathcal{O}_{\mathsf{Check}}$ returns a bit $b = \textsc{Check}(T_i, T_j)$. If $b = 1$, then this entails that T_i and T_j store a pair of attributes that match; otherwise, they do not.
- $\mathcal{O}_{\mathsf{Flip}}(\mathsf{T}_0, \mathsf{T}_1)$: When queried with two tags T_0 and T_1, $\mathcal{O}_{\mathsf{Flip}}$ flips a fair coin $b \in \{0, 1\}$. If $b = 1$, then $\mathcal{O}_{\mathsf{Flip}}$ returns tag T_1; otherwise, it returns tag T_0.

3.1 Security

In the following, we introduce the security requirements of T-MATCH.

Completeness. Completeness ensures that if two tags T_i and T_j store a pair of matching attributes, then $\textsc{Check}(T_i, T_j)$ outputs $b = 1$.

Definition 1 (Completeness). T-MATCH *is complete* $\Leftrightarrow$ *For any pair of tags* (T_i, T_j) *that store a pair of matching attributes,* $\textsc{Check}(T_i, T_j) = 1$.

Soundness. Soundness assures that if the CHECK function outputs $b = 1$, then this means that the tags T_i and T_j presented to reader R_k encode a pair of attributes a_{T_i} and a_{T_j} that match with an overwhelming probability.

We formalize soundness using a game-based definition as depicted in Algorithm 1 and Algorithm 2. In the learning phase, challenger $\mathcal{C}$ calls the oracle $\mathcal{O}_{\mathsf{Tag}}$ that supplies $\mathcal{A}$ with r tags T_i. $\mathcal{A}$ is allowed to read and write into tags T_i. He can also query the oracle $\mathcal{O}_{\mathsf{Check}}$ with any tag from the set of r tags T_i for a maximum of s times.

In the challenge phase, adversary $\mathcal{A}$ submits two challenge tags T_0 and T_1 to challenger $\mathcal{C}$, who queries the oracle $\mathcal{O}_{\mathsf{Check}}$ with tags T_0 and T_1. Finally, the oracle $\mathcal{O}_{\mathsf{Check}}$ outputs a bit b.

Adversary $\mathcal{A}$ is said to win the soundness game, if **i.)** $b = 1$ and if **ii.)** T_0 and T_1 encode two attributes a_{T_0} and a_{T_1} that do not match.

for $i := 1$ **to** r **do**
  $T_i \leftarrow \mathcal{O}_{\mathsf{Tag}}(\mathsf{param}_i)$;
  for $j := 1$ **to** s **do**
    $S^j_{T_i} = \textsc{Read}(T_i)$;
    $\textsc{Write}(T_i, S'^j_{T_i})$;
    $T_{(i,j)} \leftarrow \mathcal{O}_{\mathsf{Tag}}(\mathsf{param}_{(i,j)})$;
    $S_{T_{(i,j)}} = \textsc{Read}(T_{(i,j)})$;
    $\textsc{Write}(T_{(i,j)}, S'_{T_{(i,j)}})$;
    $b_{(i,j)} \leftarrow \mathcal{O}_{\mathsf{Check}}(T_i, T_{(i,j)})$;
  end
end

Algorithm 1. Learning phase of the soundness game

$(\mathsf{T}_0, \mathsf{T}_1) \leftarrow \mathcal{A}$;
$b \leftarrow \mathcal{O}_{\mathsf{Check}}(\mathsf{T}_0, \mathsf{T}_1)$;

Algorithm 2. Challenge phase of the soundness game

The advantage ϵ of adversary $\mathcal{A}$ in winning the soundness game is defined as:

$$\epsilon = Pr(\mathcal{A} \text{ wins})$$

Definition 2 (Soundness). T-MATCH *is sound,* **iff** *for any adversary* $\mathcal{A}(r, s, \epsilon)$*, the advantage* ϵ *in winning the soundness game is negligible.*

The definition above captures the capabilities of an active adversary $\mathcal{A}$, who in addition to being able to read tags, can re-write their internal states. The adversarial goal of $\mathcal{A}$ is to provide a pair of tags T_0 and T_1 which do not store matching attributes, yet $\textsc{Check}(\mathsf{T}_0, \mathsf{T}_1)$ outputs 1.

3.2 Privacy

T-MATCH is said to be privacy preserving, with respect to tags in the supply chain *if the only information learned by an adversary* $\mathcal{A}$ *after executing* T-MATCH *with a pair of tags* T_i *and* T_j *is the output of* $\textsc{Check}(T_i, T_j)$. That is, adversary $\mathcal{A}$ only learns whether tags T_i and T_j match or not.

Along these lines, we define first T-MATCH's privacy against readers R_k and backend server S, so as to measure information leakage through reader and backend server interaction. Second, we define T-MATCH's privacy against an *outsider adversary* $\mathcal{A} \notin \{R_k, S\}$ to quantify information leakage through the wireless channel between tags and readers R_k in the supply chain.

Privacy against Readers R_k and Backend Server S. In accordance with previous work on secure two-party computation [6], we define privacy of T-MATCH against readers R_k and backend server S in the semi-honest model by considering, first an ideal model in which both parties communicate their inputs to a TTP that computes the output of the CHECK function for reader R_k and backend server S. Then, we consider an execution of T-MATCH which evaluates the CHECK function in the real model without a TTP as depicted in Fig. 1.

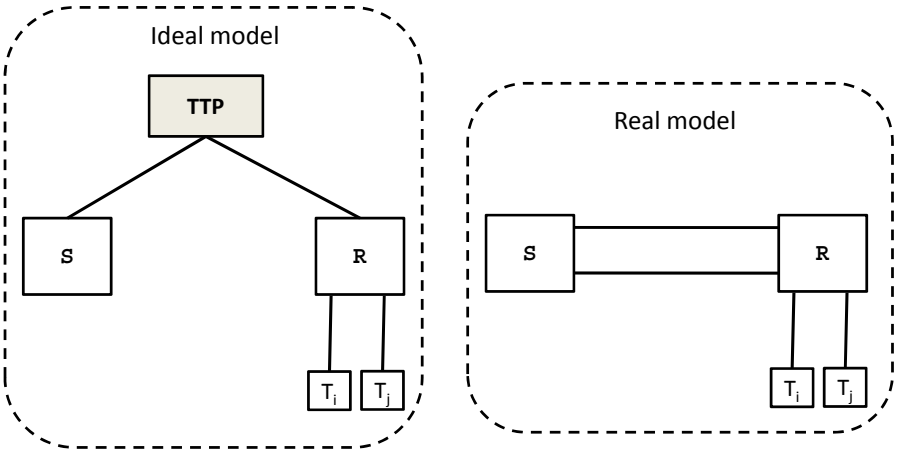


Fig. 1. Computing CHECK in both the ideal model and the real model

T-MATCH is said to be privacy preserving against readers R_k and backend server S, if for every semi-honest behavior of one of the parties (reader R_k or backend server S), the joint view of both parties can be simulated by a computation of the CHECK function in the ideal model, where also one party is semi-honest and the other is honest. That is, T-MATCH does not leak information about the private inputs of readers R_k and backend server S.

Definition 3 (Privacy against readers R_k and backend server S [6]). *Let $\bar{\mathcal{A}} = (\mathcal{A}_1, \mathcal{A}_2)$ be an admissible pair representing adversarial behavior by reader R_k and backend server S in the real model. Such a pair is admissible if at least one party $\mathcal{A}_i$ is honest.*

- *On input pair (X, Y) (X is R_k's input and Y is S's input), let $\mathsf{View}_1 = (X, r, M_1, ..., M_p, \text{CHECK}(X, Y))$ denote the view of reader R_k, where r is the outcome of R_k's internal randomness, and M_i is the i^{th} message that R_k has received.*
- *Let $\mathsf{View}_2 = (Y, r', M'_1, ..., M'_q, \perp)$ denote the view of backend server S, where r' is the outcome of S's internal randomness, and M'_i is the i^{th} message that S has received.*

We denote the joint execution under $\bar{\mathcal{A}}$ in the real model on input pair (X, Y) $\mathsf{Real}_{\bar{\mathcal{A}}}(X, Y)$, and it is defined as $(\mathcal{A}_1(\mathsf{View}_1), \mathcal{A}_2(\mathsf{View}_2))$.

Let $\bar{\mathcal{B}} = (\mathcal{B}_1, \mathcal{B}_2)$ be an admissible pair representing adversarial behavior by reader R_k and backend server S in the ideal model.

We denote the joint execution under $\bar{\mathcal{B}}$ in the ideal model on input pair (X, Y) $\mathsf{Ideal}_{\bar{\mathcal{B}}}(X, Y)$, and it is defined as $(\mathcal{B}_1(X, \text{CHECK}(X, Y)), \mathcal{B}_2(Y, \perp))$.

T-MATCH *is said to be privacy preserving with respect to readers R_k and backend server S in the semi-honest model, if there is a transformation of pairs*

of admissible adversaries $\bar{\mathcal{A}} = (\mathcal{A}_1, \mathcal{A}_2)$ *in the real model, into pairs of admissible adversaries* $\bar{\mathcal{B}} = (\mathcal{B}_1, \mathcal{B}_2)$ *in the ideal model, so that the distributions* $\{\mathsf{Real}_{\bar{\mathcal{B}}}(X, Y)\}_{X,Y}$ *and* $\{\mathsf{Ideal}_{\bar{\mathcal{B}}}(X, Y)\}_{X,Y}$ *are computationally indistinguishable.*

Using the notations of Section 2.2, we indicate that:

- the input X of reader R_k to T-MATCH is defined as its secret share α_{R_k} and the states $S_{T_i}^{k_i}$ and $S_{T_i}^{k_j}$ of tags T_i and T_j respectively;
- the input Y of backend server S to T-MATCH is its set of matching references Ref and its secret share α_S;
- at the end of T-MATCH's execution, only reader R_k gets the bit $b = \textsc{Check}(T_i, T_j)$.

Privacy against Outsiders. Ideally, a privacy preserving protocol for tag matching against an *outsider* adversary $\mathcal{A}$ should provide tag unlinkability. As discussed previously, tag unlinkability is the privacy property that ensures that it is computationally infeasible for an adversary $\mathcal{A}$ to tell two tags T_i and T_j apart.

However, we note that any adversary $\mathcal{A}$ who has access to the output of the CHECK function can mount a trivial attack against tag unlinkability. In fact, to break tag unlinkability for a pair of tags (T_i, T_j), all $\mathcal{A}$ has to do is to run T-MATCH, first with pair of tags (T_i, T_k) and then with pair of tag (T_j, T_k). Next, if $\textsc{Check}(T_i, T_k) \neq \textsc{Check}(T_j, T_k)$, then $\mathcal{A}$ concludes that T_i and T_j encode different attributes, and by the same token, he concludes that T_i and T_j are different tags, breaking hereby tag unlinkability.

Also, it is impossible to ensure *tag unlinkability* against an adversary who monitors all of the tags' interactions. We recall that T-MATCH targets storage only tags, and therewith, relies on readers R_k to update the tags' states. As a result, the state of a tag T_i does not change in between two protocol executions. Accordingly, we relax the definition of tag unlinkability, by assuming that there is at least one unobserved interaction between tag T_i and an honest reader R_k outside the range of adversary $\mathcal{A}$.

Now in accordance with previous work of Juels and Weis [10] and Avoine [2], we use an indistinguishability based definition to formalize tag unlinkability.

In the learning phase as depicted in Algorithm 3, challenger $\mathcal{C}$ provides adversary $\mathcal{A}$ with access to the oracle $\mathcal{O}_{\mathsf{Tag}}$ that $\mathcal{A}$ can query to get a set of r tags which he can read from and write into, and for which he can query the oracle $\mathcal{O}_{\mathsf{Check}}$ for a maximum of s times.

In the challenge phase, cf. Algorithm 4, $\mathcal{A}$ generates two challenge tags T_0 and T_1 that he submits to challenger $\mathcal{C}$. These two tags are read outside the range of adversary $\mathcal{A}$ and submitted to the oracle $\mathcal{O}_{\mathsf{Flip}}$. Next, the oracle $\mathcal{O}_{\mathsf{Flip}}$ supplies $\mathcal{A}$ with tag T_b, $b \in \{0, 1\}$. Finally, $\mathcal{A}$ outputs his guess b' for the value of b.

$\mathcal{A}$ is said to win the tag unlinkability game if $b = b'$.

for $i := 1$ **to** r **do**
  $T_i \leftarrow \mathcal{O}_{\mathsf{Tag}}(\mathsf{param}_i)$;
  for $j := 1$ **to** s **do**
    $S^j_{T_i} = \text{READ}(T_i)$;
    $\text{WRITE}(T_i, S'^j_{T_i})$;
    $T_{(i,j)} \leftarrow \mathcal{O}_{\mathsf{Tag}}(\mathsf{param}_{(i,j)})$;
    $S_{T_{(i,j)}} = \text{READ}(T_{(i,j)})$;
    $\text{WRITE}(T_{(i,j)}, S'_{T_{(i,j)}})$;
    $\mathcal{O}_{\mathsf{Check}}(T_i, T_{(i,j)})$;
  end
end

Algorithm 3. Learning phase of the tag unlinkability game

$(\mathsf{T}_0, \mathsf{T}_1) \leftarrow \mathcal{A}$;
// T_0 and T_1 are read outside the range of $\mathcal{A}$
$\mathsf{T}_b \leftarrow \mathcal{O}_{\mathsf{Flip}}(\mathsf{T}_0, \mathsf{T}_1)$;
$\text{READ}(\mathsf{T}_b)$;
OUTPUT b';

Algorithm 4. Challenge phase of the tag unlinkability game

The advantage ϵ of adversary $\mathcal{A}$ in wining the tag unlinkability game is defined as:

$$\epsilon = Pr(\mathcal{A} \text{ wins}) - \frac{1}{2}$$

Definition 4 (Tag Unlinkability). T-MATCH *ensures tag unlinkability,* **iff** *for any adversary* $\mathcal{A}(r, s, \epsilon)$*, the advantage* ϵ *in winning the tag unlinkability game is negligible.*

Roughly speaking, the above definition of tag unlinkability ensures that if a pair of tags T_i and T_j interact with an honest reader outside the range of a *narrow* adversary[1] $\mathcal{A}$ at least once, then it is computationally infeasible for adversary $\mathcal{A}$ to distinguish between tags T_i and T_j.

4 Protocol

To perform tag matching in T-MATCH, we store into each tag T_i an IND-CPA homomorphic encryption $\mathsf{Enc}(a_{T_i})$ of its attribute a_{T_i}. When reader R_k reads a pair of tags T_i and T_j, it uses the homomorphic property of Enc to compute an encryption $C_{(i,j)}$ of a function f of T_i and T_j's attributes, i.e., $C_{(i,j)} = \mathsf{Enc}(f(a_{T_i}, a_{T_j}))$.

Now, the matching reference of any pair of attributes (a_i, a_j) is computed as $\mathsf{ref}_{(i,j)} = f(a_i, a_j)$. To evaluate the CHECK function, reader R_k and backend server S rely on a two party privacy preserving *plaintext equality test* [9] (PET for short) to decide whether $C_{(i,j)}$ encrypts one of S's matching references or not.

Although it may seem that any IND-CPA homomorphic encryption such as Elgamal or Paillier could suit the privacy and the security requirements of T-MATCH when readers R_k in the supply chain and backend server S are semi-honest, not all of them prevent backend server S from forging new matching

[1] An adversary who does not always access the oracle $\mathcal{O}_{\mathsf{Check}}$ [17].

references from its initial set Ref. We recall that Elgamal is multiplicatively homomorphic and thus the function f is going to be expressed as $f(a_i, a_j) = \psi(a_i)\psi(a_j) = \mathsf{ref}_{(i,j)}$, where ψ is the attribute encoding in T-Match. We note also that Paillier is additively homomorphic, and as a consequence: $f(a_i, a_j) = \psi(a_i) + \psi(a_j) = \mathsf{ref}_{(i,j)}$.

Therefore, neither the use of Elgamal nor Paillier as the underlying encryption technique can thwart backend sever S from forging a new matching reference ref from its set Ref.

To prevent forgery of matching references, we use Boneh-Goh-Nissim (BGN) encryption [4]. In addition to being multiplicatively homomorphic, BGN encryption allows computing an encryption of a bilinear pairing of two plaintexts from their ciphertexts. Consequently, a matching reference of two attributes a_i and a_j in T-Match is computed as: $\mathsf{ref}_{(i,j)} = f(a_i, a_j) = f(a_j, a_i) = e(\psi(a_i), \psi(a_j))$, where ψ is the attribute encoding in T-Match. we conjecture that forging a new matching reference ref from REF is as hard as the bilinear computational Diffie-Hellman problem.

Now, we introduce the definitions and the assumptions that will be used throughout the paper.

4.1 Tools

Bilinear Pairings. Let $\mathbb{G}$ and $\mathbb{G}_T$ be groups, such that $\mathbb{G}$ and $\mathbb{G}_T$ are two cyclic groups of the same finite order N. A pairing e: $\mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear pairing if:

1. e is *bilinear*: $\forall\, x, y \in \mathbb{Z}_N$, $g, h \in \mathbb{G}$, $e(g^x, h^y) = e(g, h)^{xy}$;
2. e is *computable*: there is an efficient algorithm to compute $e(g, h)$ for any $(g, h) \in \mathbb{G}^2$;
3. e is *non-degenerate*: if g is a generator of $\mathbb{G}$, then $e(g, g)$ is a generator of $\mathbb{G}_T$.

T-Match uses the BGN cryptosystem which takes place in subgroups of finite composite order that support symmetric bilinear pairings, as in previous work of Katz et al. [11].

Boneh-Goh-Nissim (BGN) Cryptosystem. We now describe Boneh-Goh-Nissim (BGN) cryptosystem that we employ to encrypt tags' attributes in T-Match.

– **Key generation:** On input of a security parameter τ, the system obtains a tuple $(q_1, q_2, \mathbb{G}, \mathbb{G}_T, e)$ such that:

  1. q_1 and q_2 are two random primes. Typically, $|q_1| = |q_2| = 512$ bits.
  2. $\mathbb{G}$ is a bilinear group of composite order $N = q_1 q_2$.
  3. $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ is a bilinear pairing.

  The system then picks up two random generators $g, u \in \mathbb{G}$ and sets $h_1 = u^{q_2}$. Finally, the system outputs the public key $\mathsf{pk} = (N, \mathbb{G}, \mathbb{G}_T, e, g, h_1)$ and the secret key $\mathsf{sk} = q_1$.

- **Encryption:** The encryption algorithm is defined in both groups $\mathbb{G}$ and $\mathbb{G}_T$.
  - *Encryption in* $\mathbb{G}$: On input of a message $m \in \mathbb{G}$, the encryption algorithm selects a random number $r \in \mathbb{Z}_N$ and computes $c = \mathsf{Enc}_{\mathbb{G}}(m) = mh_1^r$.
  - *Encryption in* $\mathbb{G}_T$: On input of a message $M \in \mathbb{G}_T$, the encryption algorithm picks a random number $r \in \mathbb{Z}_N$ and computes $C = \mathsf{Enc}_{\mathbb{G}_T}(M) = M \cdot e(g, h_1)^r$.
- **Decryption:** Decryption in BGN relies on computing the discrete logarithm in a finite group of order N. Thus, decryption takes $O(\sqrt{N})$ steps, which makes BGN only suitable for encrypting short messages. However in T-MATCH, we do not decrypt any ciphertext C. For completeness purposes, we detail below the decryption algorithm of BGN.
  - *Decryption in* $\mathbb{G}$: On input of a ciphertext $c \in \mathbb{G}$ and secret key $\mathsf{sk} = q_1$, the decryption algorithm computes: $\mathcal{C} = c^{q_1} = m^{q_1} \cdot h_1^{rq_1}$. Since the order of h_1 is q_1, it follows that $\mathcal{C} = m^{q_1}$.
    As g is a generator of $\mathbb{G}$, there exists $x_m \in \mathbb{Z}_N$ such that: $m = \mathsf{Dec}_{\mathbb{G}}(c) = g^{x_m}$, and $x_m = \mathsf{log}_{g^{q_1}}(\mathcal{C})$.
  - *Decryption in* $\mathbb{G}_T$: On input of a ciphertext $C \in \mathbb{G}_T$ and secret key $\mathsf{sk} = q_1$, the decryption algorithm computes: $\mathcal{C} = C^{q_1} = M^{q_1} \cdot e(g, h_1)^{rq_1} = M^{q_1}$, since the order of $e(g, h_1)$ is q_1.
    As $e(g, g)$ is a generator of $\mathbb{G}_T$, then there exists $x_M \in \mathbb{Z}_N$ such that: $M = e(g, g)^{x_M}$. Therefore, $\mathcal{C} = (e(g, g)^{q_1})^{x_M}$ and x_M is computed as $\mathsf{log}_{e(g,g)^{q_1}}(\mathcal{C})$. Finally, $\mathsf{Dec}_{\mathbb{G}_T}(C) = e(g, g)^{x_M} = M$.

The BGN cryptosystem is IND-CPA under the subgroup decision assumption.

Definition 5 (The Subgroup Decision Assumption [4, 12]). *Let $\mathcal{G}$ be a group of order N where $N = q_1 q_2$ is the product of two primes q_1 and q_2. The subgroup decision assumption is said to hold in $\mathcal{G}$, if given a random element u in $\mathcal{G}$, it is computationally hard to decide whether u is in the subgroup of $\mathcal{G}$ of order q_1 or not.*

Moreover, the following homomorphic properties hold:

$$\forall\ m_1, m_2 \in \mathbb{G},\ \mathsf{Enc}_{\mathbb{G}}(m_1)\mathsf{Enc}_{\mathbb{G}}(m_2) = \mathsf{Enc}_{\mathbb{G}}(m_1 m_2)$$

$$e(\mathsf{Enc}_{\mathbb{G}}(m_1), \mathsf{Enc}_{\mathbb{G}}(m_2)) = \mathsf{Enc}_{\mathbb{G}_T}(e(m_1, m_2))$$

Attribute Encoding. Let $\mathbb{G}$ be a group of composite order $N = q_1 q_2$ and $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ be a bilinear pairing.

We denote $\mathbb{G}_1$ and $\mathbb{G}_2$ the subgroups of $\mathbb{G}$ of order q_1 and q_2 respectively.

We also denote $\mathbb{G}_{T_1}$ and $\mathbb{G}_{T_2}$ the subgroups of $\mathbb{G}_T$ of order q_1 and q_2 respectively.

Let g, u be two random generators of $\mathbb{G}$. By construction, $h_1 = u^{q_2}$ is a generator of $\mathbb{G}_1$ and $h_2 = g^{q_1}$ is a generator of $\mathbb{G}_2$.

Let $x_I = q_1 x'_I$ be the issuer's secret key, where x'_I is randomly selected in $\mathbb{Z}_N^*$.

An attribute a_i in T-MATCH is encoded as $\psi(a_i) = H(a_i)^{x_I}$, where $H : \{0,1\}^* \rightarrow \mathbb{G}$ is a cryptographic hash function.

To evaluate H, issuer I can use the algorithm proposed by Icart [8] that hashes into elliptic curves.

We note that

$$\forall a_i \in \mathbb{A},\ \exists x_i \in \mathbb{Z}_N^* \text{ such that: } \psi(a_i) = H(a_i)^{x_I} = (g^{x_i})^{x_I} = g^{x_i x_I}$$
$$= g^{x_i q_1 x_I'} = (g^{q_1})^{x_i x_I'} = h_2^{x_i x_I'} \in \mathbb{G}_2,$$

And accordingly,

$$\forall (a_i, a_j) \in \mathbb{A}^2, e(\psi(a_i), \psi(a_j)) \in \mathbb{G}_{T_2}$$

4.2 T-MATCH Overview

Before presenting a detailed description of T-MATCH, we provide a quick overview on how our matching protocol works.

Each tag T_i stores a state $S_{T_i}^{k_i}$ that consists of a BGN encryption $c_{T_i}^{k_i} = \mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_i})) = \mathsf{Enc}_{\mathbb{G}}(H(a_{T_i})^{x_I})$ of T_i's attribute a_{T_i} (where $H : \{0,1\}^* \rightarrow \mathbb{G}_T$ is a cryptographic hash function, and x_I is the secret key of issuer I) together with a MAC $\sigma_{T_i}^{k_i} = \mathsf{MAC}_K(c_{T_i}^{k_i})$, i.e., $S_{T_i}^{k_i} = (c_{T_i}^{k_i}, \sigma_{T_i}^{k_i})$. Whereas, backend server S stores a set Ref of ν matching references. Each matching reference $\mathsf{ref}_{(i,j)}$ corresponds to two attributes a_i and a_j in $\mathbb{A}$ that match and it is computed as:

$$\mathsf{ref}_{(i,j)} = f(a_i, a_j) = f(a_j, a_i) = e(\psi(a_i), \psi(a_j)) = e(H(a_i)^{x_I}, H(a_j)^{x_I})$$

When two tags T_i and T_j come together in the range of a reader R_k, reader R_k reads the current states $S_{T_i}^{k_i}$ and $S_{T_j}^{k_j}$ of tags T_i and T_j respectively. Then, reader R_k checks whether the keyed MAC stored into tags T_i and T_j are valid or not. If they are, reader R_k computes the bilinear pairing $e(c_{T_i}^{k_i}, c_{T_j}^{k_j})$.

$$C_{(i,j)} = e(c_{T_i}^{k_i}, c_{T_j}^{k_j}) = e(\mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_i})), \mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_j})))$$
$$= \mathsf{Enc}_{\mathbb{G}_T}(e(\psi(a_{T_i}), \psi(a_{T_j})))$$

Next, reader R_k and backend server S engage in a secure two party protocol for *plaintext equality test* (PET) to check whether the underlying plaintext of ciphertext $C_{(i,j)}$ belongs to the set of matching references Ref of backend server S or not. That is, reader R_k and backend server S check whether:

$$\exists\ \mathsf{ref}_p \in \mathsf{Ref},\ C_{(i,j)} = \mathsf{Enc}_{\mathbb{G}_T}(\mathsf{ref}_p)$$

Now, a reader R_k outputs $b = 1$ (i.e., CHECK$(T_i, T_j) = 1$), if the plaintext equality test outputs 1; otherwise, it outputs $b = 0$.

Privacy and Security Overview. To protect the privacy of tags, a tag T_i in T-MATCH stores a BGN encryption of its attribute a_{T_i} and a keyed MAC of the encryption. In each protocol execution, the BGN encryption is re-encrypted by

readers R_k and the MAC is computed accordingly. Now, to protect the privacy of tags that participate in the matching protocol against readers R_k and backend server S, we rely on a *modified* privacy preserving plaintext equality test that is run jointly by reader R_k and backend server S. Moreover, T-MATCH uses shuffling techniques to ensure that the only information leaked at the end of the matching protocol is a bit b that indicates whether the pair of tags participating in the current execution of T-MATCH match or not.

Furthermore, to prevent backend server S from forging new matching references from its set Ref, attributes in T-MATCH are encoded as "signatures" by issuer I, while a matching reference is computed as a bilinear pairing. Finally, T-MATCH relies on a keyed MAC to prevent adversaries from tampering with tags' content without being detected.

4.3 Protocol Description

We now describe in more details how T-MATCH performs tag matching.

System Setup. A trusted third party (TTP) outputs a matching pair of BGN public key $\mathsf{pk} = (N, \mathbb{G}, \mathbb{G}_T, e, g, h_1)$ and secret key $\mathsf{sk} = q_1$, a cryptographic hash function $H : \{0,1\}^* \to \mathbb{G}_T$, a secret key $x_I = q_1 x'_I \bmod N$ where x'_I is selected randomly in $\mathbb{Z}_N^*$, and a MAC key K. The TTP selects randomly a secret share $\alpha_1 \in \mathbb{Z}_N$, then it sets the second secret share to $\alpha_2 = \mathsf{sk} - \alpha_1 = q_1 - \alpha_1 \bmod N$.

Next, the TTP computes the set Ref of matching references. On input of two attributes a_i and a_j that match, the TTP computes $\mathsf{ref}_{(i,j)} = e(\psi(a_i), \psi(a_j)) = e(\psi(a_j), \psi(a_i)) = e(H(a_i)^{x_I}, H(a_j)^{x_I}) \in \mathbb{G}_{T_2}$.

Finally, the TTP supplies

- each reader R_k with its share $\alpha_{R_k} = \alpha_1$ of secret key sk and with the MAC key K;
- backend server S with its share $\alpha_S = \alpha_2$ of secret key sk and with the set of matching references Ref;
- issuer I with the hash function H, secret key $x_I = q_1 x'_I \bmod N$ and the MAC key K.

Tag Initialization. For each new tag T_i, issuer I computes $\psi(a_{T_i}) = H(a_{T_i})^{x_I}$, such that a_{T_i} is the attribute associated with the chemical in the container that T_i will label. Then, using the BGN public key pk, issuer I picks a random number r_i^0 and computes a ciphertext $c_{T_i}^0 = \mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_i})) = \psi(a_{T_i}) h_1^{r_i^0}$. Finally, issuer I computes $\sigma_{T_i}^0 = \mathsf{MAC}_K(c_{T_i}^0)$ and stores into tag T_i the state $S_{T_i}^0 = (c_{T_i}^0, \sigma_{T_i}^0)$.

Tag Matching. We break down the tag matching protocol into three operations that describe **first**, the interaction between tags T_i, T_j and reader R_k, **second**, the interaction between reader R_k and backend server S, and **third** the computation of the output of the CHECK function by reader R_k.

Tag $T_i \leftrightarrow$ **Reader** $R_k \leftrightarrow$ **Tag** T_j**.** Assume there are two tags T_i and T_j in the range of reader R_k. Tags T_i and T_j store states $S_{T_i}^{k_i} = (c_{T_i}^{k_i}, \sigma_{T_i}^{k_i})$ and $S_{T_j}^{k_j} = (c_{T_j}^{k_j}, \sigma_{T_j}^{k_j})$ respectively.

Reader R_k first reads out the tags T_i and T_j and checks whether $\sigma_{T_i}^{k_i} = \mathsf{MAC}_K(c_{T_i}^{k_i})$ and $\sigma_{T_j}^{k_j} = \mathsf{MAC}_K(c_{T_j}^{k_j})$ or not. If not, reader R_k updates the states of tags T_i and T_j and aborts the protocol. Otherwise, it updates the states of tags T_i and T_j and continues the execution of the protocol.

Now to update the state of tag T_i participating in the protocol, reader R_k proceeds as follows.

- If $\sigma_{T_i}^{k_i} = \mathsf{MAC}_K(c_{T_i}^{k_i})$, then reader R_k picks a random numbers r_i' and re-encrypts the ciphertexts $c_{T_i}^{k_i}$ to obtain new BGN ciphertext $c_{T_i}^{k_i+1} = c_{T_i}^{k_i} h_1^{r_i'}$. Then, it computes $\sigma_{T_i}^{k_i+1} = \mathsf{MAC}_K(c_{T_i}^{k_i+1})$. Finally, reader R_k writes the new state $S_{T_i}^{k_i+1} = (c_{T_i}^{k_i+1}, \sigma_{T_i}^{k_i+1})$ into tag T_i.
- If $\sigma_{T_i}^{k_i} \neq \mathsf{MAC}_K(c_{T_i}^{k_i})$, then reader R_k picks two random strings (st_1, st_2) and stores them into tag T_i.

Reader R_k then computes the BGN ciphertext $C_{(i,j)} = e(c_{T_i}^{k_i}, c_{T_j}^{k_j}) \in \mathbb{G}_T$.

Without loss of generality, we assume that $c_{T_i}^{k_i} = \mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_i})) = \psi(a_{T_i}) h_1^{r_i}$ and $c_{T_j}^{k_j} = \mathsf{Enc}_{\mathbb{G}}(\psi(a_{T_j})) = \psi(a_{T_j}) h_1^{r_j}$, $r_i, r_j \in \mathbb{Z}_N$. By bilinearity of e:

$$\begin{aligned} C_{(i,j)} &= e(c_{T_i}^{k_i}, c_{T_j}^{k_j}) = e(\psi(a_{T_i}) h_1^{r_i}, \psi(a_{T_j}) h_1^{r_j}) \\ &= e(\psi(a_{T_i}), \psi(a_{T_j}) h_1^{r_j}) \cdot e(h_1^{r_i}, \psi(a_{T_j}) h_1^{r_j}) \\ &= e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot e(\psi(a_{T_i}), h_1^{r_j}) \cdot e(h_1^{r_i}, \psi(a_{T_j})) \cdot e(h_1^{r_i}, h_1^{r_j}) \end{aligned}$$

We recall that:

- $h_1 = u^{q_2}$ where u is a generator of $\mathbb{G}$, and that there exist $x \in \mathbb{Z}_N$ such that $h_1 = g^x$;
- $\psi(a_{T_i})$ and $\psi(a_{T_j})$ are elements of $\mathbb{G}_2$ and that $h_2 = g^{q_1}$ is generator of $\mathbb{G}_2$. As a result, there exist $x_i,\ x_j \in \mathbb{Z}_N$ such that $\psi(a_{T_i}) = h_2^{x_i} = g^{q_1 x_i}$ and $\psi(a_{T_j}) = h_2^{x_j} = g^{q_1 x_j}$.

$$\begin{aligned} C_{(i,j)} &= e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot e(g^{q_1 x_i}, u^{q_2 r_j}) \cdot e(u^{q_2 r_i}, g^{q_1 x_j}) \cdot e(g^{x r_i}, h_1^{r_j}) \\ &= e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot e(g^{x_i}, u^{r_j})^{q_1 q_2} \cdot e(u^{r_i}, g^{x_j})^{q_1 q_2} \cdot e(g, h_1)^{x r_i r_j} \\ &= e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot \underbrace{e(g^{x_i}, u^{r_j})^N}_{1} \cdot \underbrace{e(u^{r_i}, g^{x_j})^N}_{1} \cdot e(g, h_1)^{x r_i r_j} \\ &= e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot e(g, h_1)^R \end{aligned}$$

where $R = x r_i r_j$ is distributed in $\mathbb{Z}_N$, thus:

$$C_{(i,j)} = \mathsf{Enc}_{\mathbb{G}_T}(e(\psi(a_{T_i}), \psi(a_{T_j})))$$

This directly follows from the homomorphic property of BGN as illustrated in Section 4.1.

Reader $R_k \leftrightarrow$ Backend Server S. Reader R_k then sends ciphertext $C_{(i,j)}$ to backend server S.

Without loss of generality, we assume that $\mathsf{Ref} = \{\mathsf{ref}_1, \mathsf{ref}_2, ..., \mathsf{ref}_\nu\}$, and that for all $\mathsf{ref}_p \in \mathsf{Ref}$, there exist a_i and a_j in $\mathbb{A}$, such that $\mathsf{ref}_p = e(\psi(a_i), \psi(a_j))$.

Upon receiving ciphertext $C_{(i,j)}$ from reader R_k, backend server S proceeds as follows:

- It picks ν random numbers $r_p \in \mathbb{Z}_N^*$, and computes ν ciphertexts $C_p = \left(\frac{C_{(i,j)}}{\mathsf{ref}_p}\right)^{r_p}$, for all p in $\{1, 2, ..., \nu\}$.
- On input of its secret share α_2 and ciphertexts C_p, backend server S computes $M'_p = (M_{(1,p)}, M_{(2,p)}) = (C_p, C_p^{\alpha_2})$. Next, backend server S shuffles M'_p.
  We note that by shuffling messages M'_p, T-MATCH prevents semi-honest readers R_k from telling whether two pairs of matching tags satisfy the same matching reference or not.
- Finally, backend server S sends M'_p to reader R_k.

The Output of the CHECK Function. When receiving M'_p from backend server S, reader R_k uses its secret share α_1 and computes:

$$
\begin{aligned}
M_p = {M_{(1,p)}}^{\alpha_1} \cdot M_{(2,p)} = C_p^{\alpha_1} \cdot C_p^{\alpha_2} = C_p^{\alpha_1+\alpha_2} = C_p^{q_1} &= \left(\left(\frac{C_{(i,j)}}{\mathsf{ref}_p}\right)^{r_p}\right)^{q_1} \\
&= \left(\left(\frac{e(\psi(a_{T_i}), \psi(a_{T_j})) \cdot e(g, h_1)^R}{\mathsf{ref}_p}\right)^{r_p}\right)^{q_1} \\
&= \left(\frac{e(\psi(a_{T_i}), \psi(a_{T_j}))}{\mathsf{ref}_p}\right)^{q_1 r_p} \cdot e(g, h_1)^{q_1 R r_p} \\
&= \left(\frac{e(\psi(a_{T_i}), \psi(a_{T_j}))}{\mathsf{ref}_p}\right)^{q_1 r_p}
\end{aligned}
$$

Note that if T_i and T_j match then there exists a matching reference $\mathsf{ref}_p \in \mathsf{Ref}$ such that: $e(\psi(a_{T_i}), \psi(a_{T_j})) = \mathsf{ref}_p$. That is:

$$
\exists\ p \in \{1, \nu\} \text{ such that: } M_p = \left(\frac{e(\psi(a_{T_i}), \psi(a_{T_j}))}{\mathsf{ref}_p}\right)^{q_1 r_p} = 1
$$

Consequently, if there exists $p \in \{1, 2, ..., \nu\}$ such that $M_p = 1$, then reader R_k outputs $b = 1$ meaning that T_i and T_j match. Otherwise, R_k outputs $b = 0$, i.e., T_i and T_j do not match.

5 Discussion

Due to limited space, the formal proofs of T-MATCH's security and privacy can be found in an extended version of this paper. Here, we present a quick overview of their main ideas and rationale.

Security. To prove that T-MATCH is secure against semi-honest adversaries, we rely on the security of MAC and the security of the hash function H. The security of MAC ensures that an adversary $\mathcal{A}$ who does not have the secret key K cannot create a new tag T_i that does not encode a valid attribute and which can be accepted by readers R_k. Thus, to break the security of T-MATCH, adversary $\mathcal{A}$ has to use tags that encode valid attributes (i.e., tags that were issued by issuer I and updated by readers R_k). Now, the security of the hash function H ensures that for any pair of attributes $\{a_i, a_j\} \neq \{a_p, a_q\} \subset \mathbb{A}$, $e(\psi(a_i), \psi(a_j)) \neq e(\psi(a_p), \psi(a_q))$. Consequently, if the CHECK function outputs $b = 1$ for a pair of tags T_i and T_j, then this implies that tags T_i and T_j encode *valid* attributes that match.

Privacy. T-MATCH ensures the privacy of tags T_i. First, a tag T_i in T-MATCH stores a state $S_{T_i} = (c_{T_i}, \sigma_{T_i})$, where c_{T_i} is a BGN encryption of T_i's attribute, while $\sigma_{T_i} = \mathsf{MAC}_K(c_{T_i})$. The state S_{T_i} is updated after each read by readers R_k. Thanks to the IND-CPA property of BGN, an adversary $\mathcal{A}$ who does not monitor all of T_i's interactions nor does he observe the output of the CHECK function will be unable to link the interactions of tag T_i.

Second, by using secret sharing techniques, neither readers R_k nor backend server S can disclose the encoded attribute stored into T_i unless they collude and perform a threshold decryption. Moreover, since backend server S randomizes the ciphertexts $C_p = \frac{C_{(i,j)}}{\mathsf{ref}_p}$, for all $\mathsf{ref}_p \in \mathsf{Ref}$, and shuffles the messages $M'_p = (C_p, C_p^{\alpha_2})$, it follows that at the end of an execution of T-MATCH, the only information a semi-honest reader R_k learns is the output of the CHECK function.

From Semi-honest Adversaries to Malicious Adversaries. As established by Goldreich [6], a semi-honest behavior can be enforced in the malicious model as long as trapdoor permutations exist. The idea is to **1.)** use commitment schemes to force each party to commit to their local inputs and to generate random numbers that are uniformly distributed. In the case of T-MATCH, each reader R_k commits to its secret share α_1 and the states of tags it has read, while backend server S commits to its secret share α_2, its set of matching references Ref and the randomness it uses to compute the messages M'_p. Then, **2.)** zero knowledge proofs are used to ensure that the messages exchanged between reader R_k and backend server S are protocol compliant.

While the above techniques enforce semi-honest behavior between readers R_k and backend server S, they do not enforce semi-honest behavior with respect to tags participating in the protocol, since tags in T-MATCH are storage only and do not feature any computational capabilities. However, as discussed earlier, such attacks can be detected with human inspection.

Also, we conjecture that as long as readers R_k and backend server S do not collude against tags, the only information they can learn at the end of the execution of T-MATCH is the outcome of the CHECK function. Still, we note that observing the output of the CHECK function over multiple protocol sessions allows any adversary to infer information about tags. This cannot be circumvented as it is inherent to the nature of tag matching protocols.

Table 1. Evaluation of memory and computation in T-MATCH

	Tag	Reader R_k	Backend server S
Memory	1184 bits	pk, K, α_1	pk, α_2, Ref
Exponentiation in $\mathbb{G}_T$ $\lvert\mathbb{G}_T\rvert = 2048$ bits	–	ν	2ν
Exponentiation in $\mathbb{G}$ $\lvert\mathbb{G}\rvert = 1024$ bits	–	2	–
MAC	–	2	–
Bilinear pairing	–	1	–
Shuffle	–	–	1

6 Evaluation

T-MATCH targets read/write only tags that do no feature any computational capabilities. A tag in T-MATCH is required to store a BGN ciphertext in $\mathbb{G}$ ($|\mathbb{G}| = 1024$ bits) and a MAC of size 160 bits, totaling a storage of 1184 bits.

We believe that T-MATCH can be deployed using current ISO 18000-3 HF tags, such as UPM RFID HF RaceTrack tags [16] that feature up to 8 Kbits of memory.

In each execution of T-MATCH, reader R_k reads two tags T_i and T_j and updates their states as follows: it re-encrypts the BGN ciphertexts c_{T_i} and c_{T_j} of tags T_i and T_j respectively, then computes the MAC of the re-encrypted ciphertexts. This amounts to computing two exponentiations in $\mathbb{G}$ and two keyed hash functions.

To evaluate the CHECK function, reader R_k computes a bilinear pairing $C_{(i,j)} = e(c_{T_i}, c_{T_j}) \in \mathbb{G}_T$ such that $|\mathbb{G}_T| = 2048$ bits. Then, reader R_k initiates a two round protocol for plaintext equality test with backend server S by sending the ciphertext $C_{(i,j)}$.

Upon receiving ciphertext $C_{(i,j)}$, backend server S performs 2ν exponentiations in $\mathbb{G}_T$, where ν is the number of matching references in Ref, and obtains ν messages M'_p. Next, backend server S shuffles the messages M'_p and sends them to reader R_k.

Finally, when reader R_k receives the messages M'_p, it performs ν exponentiations in $\mathbb{G}_T$ and outputs the outcome of the CHECK function.

7 Related Work

T-MATCH shows similarities to secret handshake and secret matching protocols. Nevertheless, traditional solutions for secure and privacy preserving secret matching between two parties as proposed by Ateniese et al. [1], Balfanz et al. [3] cannot be implemented in cheap RFID tags. These solutions require the computation of bilinear pairings which cannot be performed by current RFID tags.

Boneh et al. [4] propose a protocol that allows the public evaluation of 2-DNF formula on boolean variables by relying on the BGN encryption. The protocol

proposed in [4] can be slightly modified to implement tag matching. However in this case, tags are required to store $O(l)$ ciphertexts of size 1024 bits where l is the number of attributes in the system – rendering such an approach unrealistic.

Another approach to evaluate the CHECK function is attribute based encryption see Goyal et al. [7], Pirretti et al. [13], Sahai and Waters [14]. The idea is to associate each attribute a_i in the system with some secret component of some private key sk. When two tags T_i and T_j that match come together, the secret key sk can be reconstructed. The reconstruction of a correct secret key sk enables reader R_k to decrypt some ciphertext C for which it knows the underlying plaintext M. The tag matching is verified by checking whether $\mathsf{Dec}_{\mathsf{sk}}(C) = M$ or not. Though, the use of attribute based encryption can allow reader R_k to evaluate the CHECK function by itself without a backend server S, it requires either cryptographic operations on tags or their synchronization.

8 Conclusion

RFID tag based matching is required by many real-world supply-chain applications. Matching however, raises new security and privacy concerns. T-MATCH tackles these challenges and provides secure and privacy preserving item matching suited for resource restricted tags that are unable to perform any computation. T-MATCH evaluates, in a privacy preserving manner, a function CHECK that on the input of two tags T_i and T_j outputs a bit b indicating whether T_i and T_j match or not. T-MATCH is provably secure and privacy preserving under standard assumptions: security of MAC and hash functions, and the subgroup decision assumption. Finally, designed for read/write only tags, T-MATCH requires tags to store only 150 bytes.

References

[1] Ateniese, G., Kirsch, J., Blanton, M.: Secret Handshakes with Dynamic and Fuzzy Matching. In: Proceedings of the Network and Distributed System Security Symposium, NDSS. The Internet Society (2007)

[2] Avoine, G.: Adversarial Model for Radio Frequency Identification. Cryptology ePrint Archive, Report 2005/049 (2005), http://eprint.iacr.org/2005/049.pdf

[3] Balfanz, D., Durfee, G., Shankar, N., Smetters, D., Staddon, J., Wong, H.C.: Secret Handshakes from Pairing-Based Key Agreements. In: Proceedings of the 2003 IEEE Symposium on Security and Privacy, SP 2003, p. 180. IEEE Computer Society, Los Alamitos (2003) ISBN 0-7695-1940-7

[4] Boneh, D., Goh, E.-J., Nissim, K.: Evaluating 2-DNF Formulas on Ciphertexts. In: Kilian, J. (ed.) TCC 2005. LNCS, vol. 3378, pp. 325–341. Springer, Heidelberg (2005)

[5] Cobis Consortium. Collaborative Business Items: Chemical drums use-case (2007), http://www.cobis-online.de/files/live.stream.wvx

[6] Goldreich, O.: Foundations of Cryptography. Basic Applications, vol. 2. Cambridge University Press, New York (2004) ISBN 0521830842

[7] Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, pp. 89–98. ACM, New York (2006) ISBN 1-59593-518-5

[8] Icart, T.: How to Hash into Elliptic Curves. In: Halevi, S. (ed.) CRYPTO 2009. LNCS, vol. 5677, pp. 303–316. Springer, Heidelberg (2009)

[9] Jakobsson, M., Juels, A.: Mix and Match: Secure Function Evaluation via Ciphertexts. In: Okamoto, T. (ed.) ASIACRYPT 2000. LNCS, vol. 1976, pp. 162–177. Springer, Heidelberg (2000)

[10] Juels, A., Weis, S.A.: Defining Strong Privacy for RFID. In: PerCom Workshops, White Plains, USA, pp. 342–347 (2007) ISBN 978-0-7695-2788-8

[11] Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)

[12] Okamoto, T., Uchiyama, S.: A New Public-Key Cryptosystem as Secure as Factoring. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 308–318. Springer, Heidelberg (1998)

[13] Pirretti, M., Traynor, P., McDaniel, P., Waters, B.: Secure attribute-based systems. In: Proceedings of the 13th ACM Conference on Computer and Communications Security, CCS 2006, pp. 99–112. ACM, New York (2006) ISBN 1-59593-518-5

[14] Sahai, A., Waters, B.: Fuzzy Identity-Based Encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer, Heidelberg (2005)

[15] Shamir, A.: How to share a secret. Commun. ACM 22, 612–613 (1979) ISSN 0001-0782

[16] UPM RFID Technology. UPM RFID HF RaceTrack RFID Tag (2011), http://www.rfidtags.com/upm-rfid-racetrack-rfid-tag

[17] Vaudenay, S.: On Privacy Models for RFID. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 68–87. Springer, Heidelberg (2007)

Private Yoking Proofs: Attacks, Models and New Provable Constructions*

Jens Hermans and Roel Peeters

KU Leuven, ESAT/SCD - COSIC & IBBT
Kasteelpark Arenberg 10/2446, 3001 HEVERLEE, Belgium
firstname.lastname@esat.kuleuven.be

Abstract. We present two attacks on the security of the private grouping proof by Batina *et al.* [1]. We introduce the first formal models for yoking proofs. One model incorporates the aspect *time*, ensuring that the grouping proofs were generated at a specific time. A more general variant only provides a proof that tags were together at some time. Based on these models we propose two new protocols to generate sound yoking proofs that can trivially be extended to multiple parties and that attain narrow-strong privacy.

1 Introduction

Juels [9] introduced the concept of yoking[1] proofs, also referred to as grouping proofs. These proofs allow the reader to claim afterwards (i.e. off-line) to a trusted party, that two RFID tags were scanned roughly at the same time and communicated to each other. Tags do not contain clocks and cannot communicate to each other directly, they communicate via the potentially untrusted reader. Note that these proofs give no guarantee that the RFID tags were physically close to each other, although close timing makes it harder for an adversary to obtain a yoking proof from two tags that are far apart. At the same time tag privacy should be considered: apart from the trusted party that is able to verify the yoking proof, no information should be gained on the tags' identities. Several papers have proposed constructions to generate yoking proofs, also generalising the setting to groupings of more than two tags.

Most proposed proof systems [3,5,9,12,13,14] are based on symmetric cryptographic primitives. Lee *et al.* [11] and Hein *et al.* [7] showed that it is also possible to deploy public key cryptography on RFID tags, more specifically Elliptic Curve Cryptography. Towards tag privacy, symmetric cryptographic solutions are not scalable and only provide some basic privacy protection. Vaudenay [16] showed that public key cryptography is necessary to provide strong privacy guarantees for the tags such that no identifiable information leaks from the messages sent by the tags. Thus far, only Batina *et al.* [1] proposed two yoking proof systems

* Joking with Yoking: Two Protocols in Front of a Circus :-)
[1] From the verb *to yoke*, meaning *to join together*.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 96–108, 2013.

that are based on public key cryptography. We will show two separate attacks on the security of their proposed protocols to generate yoking proofs.

One of the crucial aspects for grouping proofs is timing. Since a grouping proof is verified off-line by definition only a trusted party can assure the time the grouping proof took place. It's insufficient to simply submit the finished proof to the trusted party after finishing, since this does not prevent delaying the submission of the proof. The trusted party should actively participate in the protocol to avoid replaying and delaying. We present two security models: one that ensures timed grouping proofs, with trusted third party and one for non-timed grouping proofs without trusted third party. In the later case the proof only guarantees that the tags participated in a grouping proof without specifying any time or order.

Outline. Section 2 presents two attacks on the Batina et al. In Sect. 3 we introduce the privacy and security model used throughout this paper. In Sect. 4 and Sect. 5 our new yoking proofs are proposed and their security and privacy is proven.

2 Attacks

Batina *et al.* [1] proposed two protocols to generate grouping proofs, one with colluding tag prevention and a basic one. Figure 1 describes the one with colluding tag prevention. The basic protocol, without colluding tag prevention, can be obtained by setting $r_s = 1$ in the protocol from Fig. 1. The proposed protocols build upon an authentication protocol, EC-RAC [10] for which the security is claimed to be related to the security of the Schnorr identification protocol [15].

We will now show how an adversary can break the security of these protocols, i.e. the adversary can generate a valid grouping proof that T_a and T_b were scanned together.

2.1 First Attack

For authentication protocols, the temporal order of the messages is crucial. Authentication protocols consist of three stages: commit, exam and response. If the value of the exam is known before the prover needs to provide the commitment to its randomness (which is used later on for the response, the prover can construct a crooked proof). This can easily be shown for tag T_b. We are only interested in the value of the exam $\alpha = \texttt{xcoord}(r_s T_{a1})$, where $\texttt{xcoord}(P)$ returns the x-coordinate of the point $P = (x_P, y_P)$. Given this tag's public key $S_b = s_b P$, the adversary can construct a valid response $T_{b,1} = rP - \alpha S_b$, $T_{b,2} = rY$ for $r \in_R \mathbb{Z}_l$. One can argue whether or not public keys of tags are known to the adversary, since the claimed privacy of the protocol implies that the adversary cannot learn the public key of an RFID tag from the exchanged messages, but we can definitely conclude that the value r_s, chosen by a genuine reader, does not provide any protection against colluding tags.

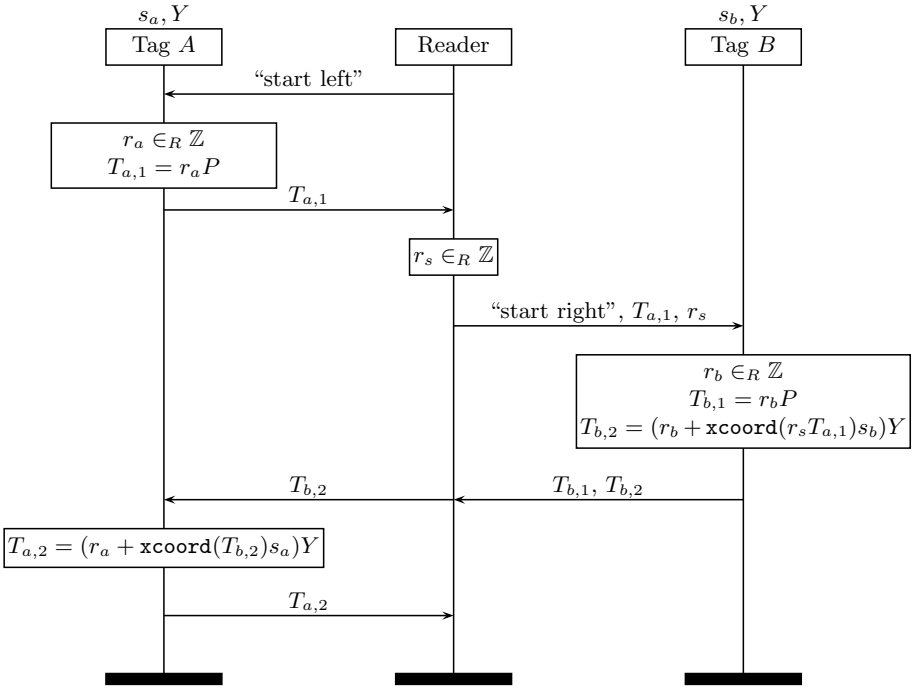


Fig. 1. Two-party grouping-proof protocol with colluding tag prevention, proposed by Batina *et al.* [1]

This weakness can be mitigated by requiring that the tag T_b first has to send the commitment $T_{b,1}$ before being presented with the exam. However, the resulting proof, presented by the (potentially untrusted) reader to the verifier, contains no verifiable information on the temporal ordering of the messages, still allowing this attack.

2.2 Second Attack

For the second attack, no knowledge of public keys of the tags is required. In the first phase the adversary needs to collect a tuple (α, T_1, T_2) for which the following relations hold: $T_1 = rP$ and $T_2 = (r + \alpha s)Y$, for s the secret key of the target tag and r an unknown random number. To collect this tuple one can eavesdrop on the protocol with honest tags: $(\texttt{xcoord}(T_{b,2}), T_{a,1}, T_{a,2})$ and $(\texttt{xcoord}(r_s T_{a1}), T_{b,1}, T_{b,2})$. Since there is no reader authentication (and the reader can be untrusted), one can also query the tags actively to collect this attack tuple.

In the second stage one can trick a genuine reader to accept $T'_{b,1}, T'_{b,2}$ as coming from the target tag, for which the attacker only has a tuple (α, T_1, T_2).

This means that one can generate arbitrary yoking proofs with respect to tag T_b. Let $\beta = \texttt{xcoord}(r'_s T'_{a1})$, then $T'_{b,1}$ and $T'_{b,2}$ are computed as follows:

$$T'_{b,1} = \gamma T_1 + \delta P \qquad T'_{b,2} = \gamma T_2 + \delta Y \quad \text{for} \quad \delta \in_R \mathbb{Z}_l \quad \text{and} \quad \gamma = \frac{\beta}{\alpha}.$$

Again, this attack is independent on the value of r_s.

3 Privacy and Security Model

In this paper we will use the privacy model from Hermans et al. [8]. We will also use the oracles defined in this privacy model for the security games.

3.1 Privacy Model of Hermans et al.

The intuition behind the RFID privacy model is that privacy is guaranteed if an adversary cannot distinguish with which one of two RFID tags (of its choosing), he is interacting through a set of oracles. A brief overview of these oracles is given in App. A.

Privacy is defined as a distinguishability game between a challenger and the adversary. This game is defined as follows. First the challenger picks a random challenge bit b and then sets up the system $\mathcal{S}$ with a security parameter k. Next, the adversary $\mathcal{A}$ can use a subset (depending on the privacy notion) of the following oracles to interact with the system: $\texttt{CreateTag}(ID)$, $\texttt{DrawTag}(T_i, T_j)$, $\texttt{Free}(vtag)_b$, $\texttt{SendTag}(vtag, m)_b$, $\texttt{SendReader}(\pi, m)$, $\texttt{Result}(\pi)$ and $\texttt{Corrupt}(T_i)$.

By using the `DrawTag` oracle the adversary can arbitrarily select which tags to interact with. Based upon the challenge bit b the system that the challenger presents to the adversary will behave as either the *left* tags T_i or the *right* tags T_j. After $\mathcal{A}$ called the oracles, it outputs a guess bit g.

In this paper the $\texttt{Result}(\pi)$ is not used, since the grouping proofs are validated off-line at a later stage. For the full privacy definition we refer the reader to [8].

For the protocols that require a trusted third party (TTP), we define the $\texttt{SendTTP}(m) \rightarrow m'$ oracle, to send a message m to the TTP and receive the reply m'.

Privacy Notions. All adversaries presented in this paper are *narrow strong* adversaries, which are allowed to use all the oracles available except the `Result` oracle.

We also define X^* privacy notion variants, where X refers to the basic privacy notion and $*$ to the notion that arises when the corruption abilities of the adversary are further restricted with respect to the `Corrupt` oracle. The restricted `Corrupt` oracle will only return the non-volatile state of the tag. This restriction allows to exclude trivial privacy attacks on multi-pass protocols, that require the tag to store some information in volatile memory during the protocol run.

3.2 Grouping Proof

A grouping proof protocol has the following two properties: correctness, soundness. Correctness and soundness are necessary to establish the security of the protocol.

A function $f : \mathbb{N} \to \mathbb{R}$ is called *polynomial* in the security parameter $k \in \mathbb{Z}$ if $f(k) = O(k^n)$, with $n \in \mathbb{N}$. It is called *negligible* if, for every $c \in \mathbb{N}$ there exists an integer k_c such that $f(k) \leq k^{-c}$ for all $k > k_c$.

Definition 1. *Correctness. A scheme is correct if a legitimate grouping proof is rejected with negligible probability and all tags involved are identified correctly with overwhelming probability.*

We make a distinction between timed grouping proofs and non-timed grouping proofs. For a timed grouping proof, the time at which the proof was generated is recorded and can be verified afterwards. For a protocol to achieve timed grouping proof soundness a trusted third party is required to provide timestamps.

Definition 2 (Timed Grouping Proof Soundness). *In the first phase of the soundness game the adversary may interact with all tags. After the first phase ends the challenger notes the current time t_1. In the second phase the adversary can also interact with all tags, except for one tag $T_c \in S$, where $S \subset \mathcal{T}$ is the set of tags for which a grouping proof is produced by the adversary. This tag T_c should also remain uncorrupted during the entire game. The adversary outputs a candidate grouping proof σ at the end of the second phase. A grouping proof scheme is sound if no polynomially bounded strong adversary, with non-negligible probability, is able to produce a valid grouping proof for a set of tags S with time $t_2 > t_1$.*

The above definition ensures that even if all tags but one participating in the yoking protocol collude it remains impossible to construct a valid grouping proof without cooperation of all tags.

A non-timed proof is restricted to proving that the tags in question were together and completed the protocol. One cannot in any way determine from the yoking proof at what time this happened. As such, once a proof is produced it can be reused without limits.

Definition 3 (Non-Timed Grouping Proof Soundness). *In the first phase of the soundness game the adversary may interact with all tags, except T_a. This also implies that corrupting T_a is impossible in the first phase.*

In the second phase the adversary cannot interact with any tag except T_a. The adversary outputs a candidate grouping proof σ at the end of the second phase. A grouping proof scheme is sound if no polynomially bounded strong adversary, with non-negligible probability, is able to produce a valid grouping proof for the group of tags $S = \{T_a, T_b, \ldots\}$, even when allowed to corrupt T_a in the second phase.

During the entire game T_b cannot be corrupted.

By splitting the soundness game in two phases we ensure that at least two of the tags in the grouping proof cannot perform a yoking protocol together. In the first phase T_a cannot be used, but T_b can, while in the second phase only T_a can be used.

4 Yoking Proof with Trusted Party

Figure 2 presents our new protocol, which is based on the Randomised Schnorr protocol [4] to ensure soundness as well as tag privacy. The exam e is generated by the trusted time stamping authority (TTSA) after receiving the tags' commitments $R_{a1}, R_{a2}, R_{b1}, R_{b2}$. This ensures the proper ordering of the messages in the authentication protocol, necessary to avoid crooked proofs. Given the exam, each tag generates a response s_a, s_b. The TTSA finally signs all messages and the timestamp provided the final values s_a, s_b arrive before the session with the TTSA times out. The signature is returned to the reader, who stores the full grouping proof σ for later verification.

Note that neither the reader, nor the TTSA are able to learn the identity of the tags. The grouping proof can only be checked by the verifier with secret key y. The proof is verified as follows:

- $\texttt{verify}(s_{TTSA})$;
- $X_a = e^{-1}(s_a P - R_{a1} - y^{-1} R_{a2})$;
- $X_b = e^{-1}(s_b P - R_{b1} - y^{-1} R_{b2})$.

The public keys X_a, X_b can be checked in the database of the verifier. This ensures that tag T_a and tag T_b were scanned together at time ts.

The main cost for each tag is two scalar-EC point multiplications. The most complex operation, the signature, is performed by the TTSA.

Our protocol can easily be extended to multiple tags, at no additional cost for the RFID tags.

4.1 Security and Privacy

Grouping Proof Soundness. The soundness of the grouping proof is based on the one more discrete logarithm (OMDL) assumption, which was introduced by Bellare *et al.* [2]. Let P be a generator of a group $\mathbb{G}_\ell$ of order ℓ. Let $\mathcal{O}_1$ be an oracle that returns random elements $A_i = a_i P$ of $\mathbb{G}_\ell$. Let $\mathcal{O}_2(\cdot)$ be an oracle that returns the discrete logarithm of a given input base P. The OMDL problem is to return the discrete logarithms for each of the elements obtained from the m queries to $\mathcal{O}_1$, while making strictly less than m queries to $\mathcal{O}_2(\cdot)$.

Theorem 1. *The protocol from Fig. 2 is timed grouping proof sound under the OMDL assumption and the existential unforgeability of the signature scheme used by the TTSA.*

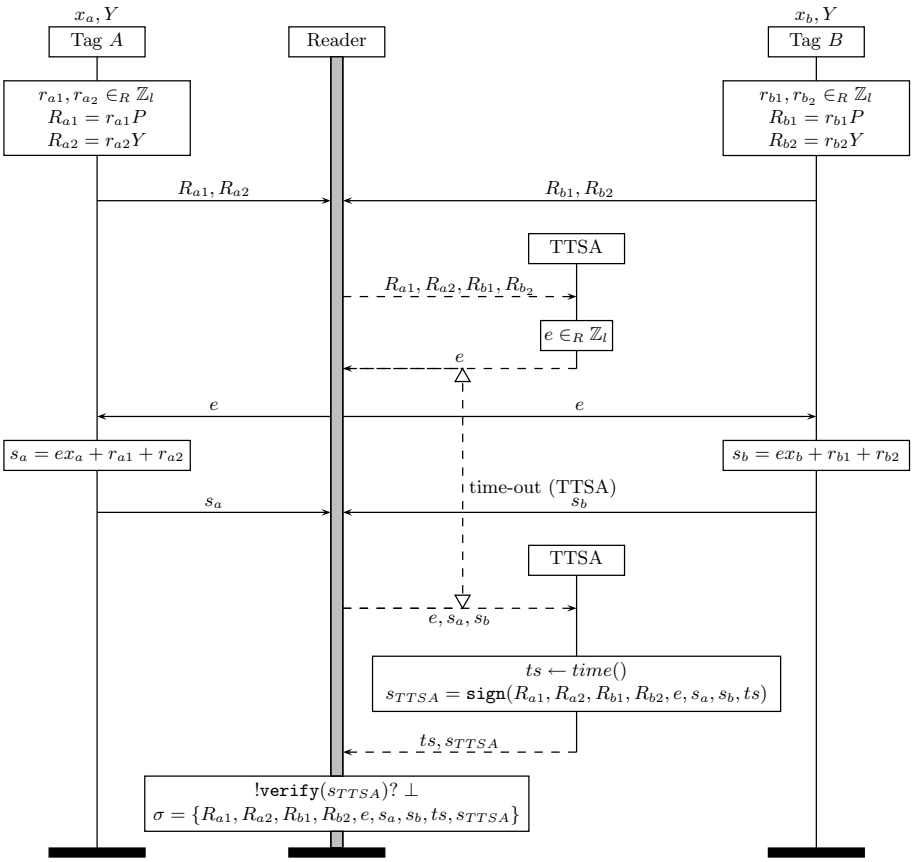


Fig. 2. Two-Party Grouping-Proof Protocol with Timestamp

Proof. Assume an adversary $\mathcal{A}$ that forges the timed grouping proof.

We now construct an adversary $\mathcal{B}$ that breaks the unforgeability of the signature σ, or an adversary $\mathcal{B}$' that breaks the OMDL.

If $\mathcal{A}$ produces a σ with timestamp $t_2 > t_1$ this implies that either it communicated at time t_2 with the TTSA to produce σ or that $\mathcal{A}$ forged the signature. In the latter case we can easily use $\mathcal{A}$ to break the existential unforgeability of the signature scheme.

From now on we can assume that the messages $R_{a1}, R_{a2}, R_{b1}, R_{b2}, e,\ s_A,\ s_B$ where faithfully exchanged with the TTSA around time t_2 using the `SendTTS` oracle.

Let $X_A = \mathcal{O}_1()$. In the first phase $\mathcal{B}$' simulates the i'th pair of `SendTag` queries to tag T_a as follows:

- First $\mathtt{SendTag}() \rightarrow R_{a1,i}, R_{a2,i}$: $R_{a1,i} = \mathcal{O}_1()$, $r_{a2,i} \in_R \mathbb{Z}_l, R_{a2,i} = r_{a2,i}Y$
- Second First $\mathtt{SendTag}(e) \rightarrow s_A$: $s_{A,i} = \mathcal{O}_2(e_i X_A + R_{a1,i}) + r_{a2,i}$

In the second phase, the adversary $\mathcal{A}$ calls `SendTTS` with $R_{a1}, R_{a2}, R_{b1}, R_{b2}$. $\mathcal{B}$' simulates `SendTTS` by generating a random e after which $\mathcal{A}$ will call `SendTTS` with s_A and s_B. Upon receiving these, $\mathcal{B}$' rewinds $\mathcal{A}$ until the moment it calls `SendTTS` with $R_{a1}, R_{a2}, R_{b1}, R_{b2}$ and sends back a fresh e' after which $\mathcal{A}$ will send new s'_A and s'_B to the TTSA. $\mathcal{B}$' can now recover $x_A = (s'_A - s_A)/(e' - e)$, and returns $x_A, \{s_{A,i} - e_i x_A - r_{a2,i}\}_i$ to the OMDL challenger, thereby solving the OMDL problem. □

Privacy. The privacy of the protocol is based on the decisional Diffie Hellman (DDH) assumption. Let P be a generator of a group $\mathbb{G}_\ell$ of order ℓ. Let $a, b, r \in_R \mathbb{Z}_l$ and $A = aP, B = bP$. The DDH assumption states that is hard to distinguish between $(A, B, C = abP)$ and $(A, B, C = rP)$.

Theorem 2. *The protocol from Fig. 2 is narrow strong* private under the DDH assumption.*

Note that the protocol uses randomized Schnorr, which has been proven narrow strong private in [4]. Below we give a modified proof for the [8] model, using a standard hybrid argument [6,17].

Proof. For simplicity we will only consider a single execution of the protocol. A full proof can be obtained by using a standard hybrid argument.

Assume an adversary $\mathcal{A}$ that breaks narrow strong privacy. We will create a adversary $\mathcal{B}$ that breaks DDH (with $A = aP, B = bP$ and $C = abP$ or $C = rP$) which executes $\mathcal{A}$. $\mathcal{B}$ sets $Y = B$ at the beginning, chooses a random bit b and simulates the `SendTag` oracles for a single protocol run to $\mathcal{A}$ as follows:

- First `SendTag`($vtag$): select $r \in_R \mathbb{Z}_l$ and return $R_{a1} = r'P - A, R_{a2} = C$
- Second `SendTag`($vtag, e$): $s_A = ex_i + r'$ where x_i is either the secret key of tag T_i or T_j, depending on the tags passed to the `DrawTag` that generated $vtag$ and the random bit b.

At the end of the game $\mathcal{A}$ outputs a guess bit g. $\mathcal{B}$ outputs $(b == g)$ as output to the DDH challenger.

In case of a real DDH instance (i.e. $C = abP$) the simulation perfectly follows the real protocol, hence it follows that $\Pr[\mathcal{B}\,\text{wins}]_{realddh} = \Pr[\mathcal{A}\,\text{wins}]$. In case of a random DDH instance (i.e. $C = rP$) $\mathcal{A}$ only obtains randomized, independent data and as such $\Pr[\mathcal{B}\,\text{wins}]_{randomddh} = \frac{1}{2}$.

It follows that $\mathbf{Adv}_{\mathcal{B}} = \frac{1}{2}\mathbf{Adv}_{\mathcal{A}}$. □

5 Yoking Proof without Trusted Parties

In case no trusted parties are available we have to rely on some form of signature (or MAC) for validation of the grouping proof. We cannot rely on authentication protocols since ordering of messages is not guaranteed when validation takes place off-line.

Figure 3 shows the proposed protocol to generate a yoking proof. In the first round, both tags T_a and T_b generate a one-time key pair for signing, with public key R_a and private key r_a. In the second round, both tags MAC both public keys with their permanent private key x_a. Note that one can also use a signature scheme instead of a MAC. In the final round, both tags sign the MAC's s_a, s_b using their one-time signing key.

One possible instantiation of the signature scheme is a Schnorr signature [15], which requires ECC and a hash function. The hash function can also be reused for the MAC function (or the MAC can be replaced with a signature). However, MAC functions and Schnorr signatures do not guarantee privacy. In Appendix B we show how to make privacy preserving signatures, which can replace the MAC function to ensure narrow strong privacy. Note that the final signature does not need to be privacy preserving as the signing key is freshly generated for every protocol run.

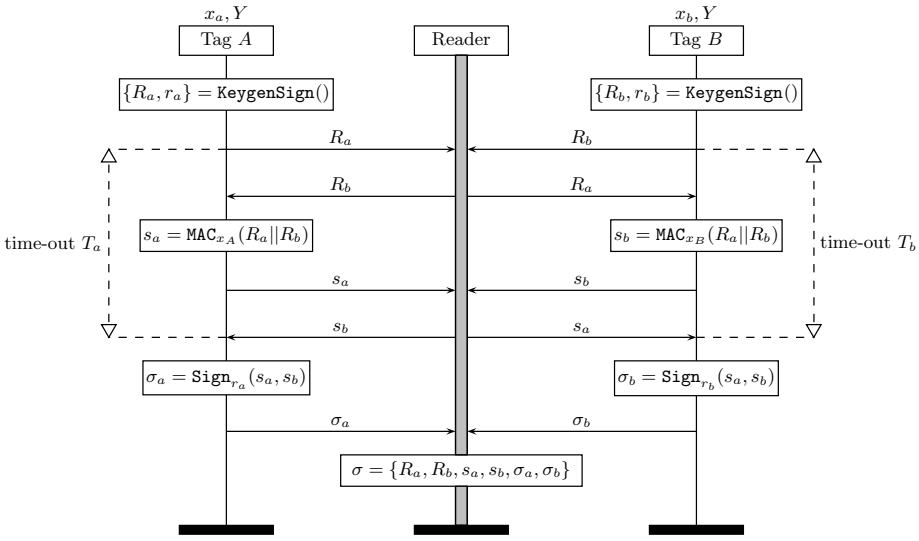


Fig. 3. Two-party grouping-proof protocol without trusted party

Our protocol can easily be extended to multiple tags, at the cost of additional communication. The computational overhead will remain small, since the number of signatures (and MACs) a tag needs to compute are independent of the number of tags in the grouping proof. However, since the messages that need to be signed (or on which the MAC algorithm needs to be deployed) increase in size, the computational effort will slightly raise.

5.1 Security Proof

Theorem 3. *The protocol from Fig. 3 is non-timed grouping proof sound under the existential unforgeability of the MAC and the (one-time) signature scheme.*

Proof. Assume an adversary $\mathcal{A}$ that breaks the non-timed grouping proof soundness. We will use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that breaks either the existential unforgeability of the MAC or the signature scheme.

$\mathcal{B}$ runs $\mathcal{A}$ internally and simulates the grouping proof challenger to $\mathcal{A}$. At the start of the grouping proof game $\mathcal{B}$ sets $X_b = \texttt{KeygenMAC}$. During the first phase of the grouping proof challenge $\mathcal{B}$ simulates the `SendTag` oracle of T_b to $\mathcal{A}$ as follows:

- First $\texttt{SendTag}() \rightarrow R_b$: return $R_b = \texttt{KeygenSign}$.
- Second $\texttt{SendTag}(R_a) \rightarrow s_b$: return $s_b = \texttt{MAC}(R_a||R_b)$
- Third $\texttt{SendTag}(s_a) \rightarrow \sigma_b$: return $\sigma_b = \texttt{Sign}(s_a, s_b)$

All other oracle queries are simulated according to the protocol specification. In the second phase of the grouping proof game, $\mathcal{B}$ generates a random x_a and passes this to $\mathcal{A}$. At the end of the game $\mathcal{A}$ outputs a $\sigma = \{R_a, R_b, s_a, s_b, \sigma_a, \sigma_b\}$.

By assumption, σ is a valid grouping proof, implying that s_b is a valid MAC. If s_b was not requested during the first phase through the `MAC` oracle with $R_a||R_b$, this implies that s_b is a valid forgery and $\mathcal{B}$ breaks the existential unforgeability of the MAC scheme.

If, on the one hand, it was requested through a `MAC` oracle call, the definition of the simulation above by $\mathcal{B}$ to $\mathcal{A}$ implies that there also was a call to `KeygenSign`, which yielded the specified R_b. Since σ is a valid grouping proof, σ_b is valid signature on s_a, s_b using the private key matching to the public key R_b. If σ_b was not requested during the first phase through the `Sign` oracle, σ_b is a valid forgery and $\mathcal{B}$ breaks the existential unforgeability of the one-time signature scheme.

If, on the other hand, σ_b was requested through a `Sign` oracle this implies that the full grouping proof presented by $\mathcal{A}$ took place in the first phase of the grouping proof challenge. This is impossible however, since x_a was only generated after the first phase. □

6 Conclusion

In this paper, we presented two attacks on the security of the yoking proofs as proposed by Batina *et al.* [1]. To ensure privacy of the RFID tags that take part in the protocol to generate a grouping proof, one should move away from the symmetric key cryptographic building blocks in favour of public key cryptography. Not only will this provide us with scalability at the verifier side, RFID tags will also have stronger privacy guarantees, i.e. narrow strong privacy. This paper introduced the first formal models of the security of yoking proofs. In the first model, time is taken into account, since for most use cases one is not only interested in two RFID tags being scanned together, but also when these tags were

scanned together. In the second, we consider how to build a grouping proof without trusted third party. We provide for each model a protocol, for which both security and privacy are proven. Our proposed protocol with trusted timestamp authority is also the first one for which the verifier can upon verification of the yoking proof be absolutely sure that the tags were scanned at this point in time.

Acknowledgements. The authors would like to thank Fréderik Vercauteren for his valuable suggestion and interesting discussions. Additionally we appreciate the comments received from the anonymous reviewers.

This work was supported in part by the Research Council K.U.Leuven: GOA TENSE (GOA/11/007), by the IAP Programme P6/26 BCRYPT of the Belgian State (Belgian Science Policy) and by the European Commission through the ICT programme under contract ICT-2007-216676 ECRYPT II. In addition, this work was supported by the Flemish Government, IWT SBO MobCom and IWT Tetra EVENT. Jens Hermans is a research assistant, sponsored by the Fund for Scientific Research - Flanders (FWO).

References

1. Batina, L., Lee, Y.K., Seys, S., Singelée, D., Verbauwhede, I.: Extending ECC-Based RFID Authentication Protocols to Privacy-Preserving Multi-Party Grouping Proofs. Journal of Personal and Ubiquitous Computing 16(3), 323–335 (2012)
2. Bellare, M., Namprempre, C., Pointcheval, D., Semanko, M.: The One-More-RSA-Inversion Problems and the Security of Chaums Blind Signature Scheme. Journal of Cryptology 16, 185–215 (2003)
3. Bolotnyy, L., Robins, G.: Generalized "Yoking-Proofs" for a Group of RFID Tags. In: Annual International Conference on Mobile and Ubiquitous Systems (MOBIQUITOUS 2006), pp. 1–4 (2006)
4. Bringer, J., Chabanne, H., Icart, T.: Cryptanalysis of EC-RAC, a RFID Identification Protocol. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 149–161. Springer, Heidelberg (2008)
5. Burmester, M., de Medeiros, B., Motta, R.: Provably Secure Grouping-Proofs for RFID Tags. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 176–190. Springer, Heidelberg (2008)
6. Goldreich, O.: Foundations of Cryptography. Basic Tools, vol. 1. Cambridge University Press (2001)
7. Hein, D., Wolkerstorfer, J., Felber, N.: ECC Is Ready for RFID – A Proof in Silicon. In: Avanzi, R.M., Keliher, L., Sica, F. (eds.) SAC 2008. LNCS, vol. 5381, pp. 401–413. Springer, Heidelberg (2009)
8. Hermans, J., Pashalidis, A., Vercauteren, F., Preneel, B.: A New RFID Privacy Model. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 568–587. Springer, Heidelberg (2011)
9. Juels, A.: "Yoking-Proofs" for RFID Tags. In: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops (PERCOMW 2004), pp. 138–143. IEEE Computer Society (2004)
10. Lee, Y.K., Batina, L., Singelée, D., Verbauwhede, I.: Low-Cost Untraceable Authentication Protocols for RFID (extended version). In: Wetzel, S., Rotaru, C.N., Stajano, F. (eds.) Proceedings of the 3rd ACM Conference on Wireless Network Security (WiSec 2010), pp. 55–64. ACM (2010)

11. Lee, Y.K., Sakiyama, K., Batina, L., Verbauwhede, I.: Elliptic Curve Based Security Processor for RFID. IEEE Transactions on Computer 57(11), 1514–1527 (2008)
12. Peris-Lopez, P., Hernandez-Castro, J., Estevez-Tapiador, J., Ribagorda, A.: Solving the Simultaneous Scanning Problem Anonymously: Clumping Proofs for RFID Tags. In: The Proceedings of the 3rd International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU 2007). IEEE Computer Society Press (2007)
13. Piramuthu, S.: On Existence Proofs for Multiple RFID Tags. In: Proceedings of the 2nd International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU 2006), pp. 317–320. IEEE, IEEE Computer Society Press (2006)
14. Saito, J., Sakurai, K.: Grouping Proof for RFID Tags. In: 19th International Conference on Advanced Information Networking and Applications (AINA 2005), pp. 621–624. IEEE Computer Society (2005)
15. Schnorr, C.P.: Efficient Identification and Signatures for Smart Cards. In: Brassard, G. (ed.) CRYPTO 1989. LNCS, vol. 435, pp. 239–252. Springer, Heidelberg (1990)
16. Vaudenay, S.: On Privacy Models for RFID. In: Kurosawa, K. (ed.) ASIACRYPT 2007. LNCS, vol. 4833, pp. 68–87. Springer, Heidelberg (2007)
17. Yao, A.C.-C.: Theory and applications of trapdoor functions (extended abstract). In: FOCS, pp. 80–91 (1982)

A Oracles Model Hermans et al.

The model of Hermans et al. [8] defines the following oracles for the privacy game:

- $\texttt{CreateTag}(ID) \rightarrow T_i$: on input a tag identifier ID, this oracle creates a tag with the given identifier and corresponding secrets, and registers the new tag with the reader. A reference T_i to the new tag is returned. Note that this does not reject duplicate IDs.
- $\texttt{Launch}() \rightarrow \pi$: this oracle launches a new protocol run on the reader R_j, according to the protocol specification. It returns a session identifier π, generated by the reader.
- $\texttt{DrawTag}(T_i, T_j) \rightarrow vtag$: on input a pair of tag references, this oracle generates a virtual tag reference, as a monotonic counter, $vtag$ and stores the triple $(vtag, T_i, T_j)$ in a table $\mathcal{D}$. Depending on the value of b, $vtag$ either refers to T_i or T_j. If T_i is already references as the left-side tag in $\mathcal{D}$ or T_j as the right-side tag, then this oracle also returns $\perp$ and adds no entry to $\mathcal{D}$. Otherwise, it returns $vtag$.
- $\texttt{Free}(vtag)_b$: on input $vtag$, this oracle retrieves the triple $(vtag, T_i, T_j)$ from the table $\mathcal{D}$. If $b = 0$, it resets the tag T_i. Otherwise, it resets the tag T_j. Then it removes the entry $(vtag, T_i, T_j)$ from $\mathcal{D}$. When a tag is reset, its volatile memory is erased. The non-volatile memory, which contains the state S, is preserved.
- $\texttt{SendTag}(vtag, m)_b \rightarrow m'$: on input $vtag$, this oracle retrieves the triple $(vtag, T_i, T_j)$ from the table $\mathcal{D}$ and sends the message m to either T_i (if

$b = 0$) or T_j (if $b = 1$). It returns the reply from the tag (m'). If the above triple is not found in $\mathcal{D}$, it returns $\perp$.
- `SendReader`$(\pi, m) \rightarrow m'$: on input π, m this oracle sends the message m to the reader in session π and returns the reply m' from the reader (if any) is returned by the oracle.
- `Result`(π): on input π, this oracle returns a bit indicating whether or not the reader accepted session π as a protocol run that resulted in successful authentication of a tag. If the session with identifier π is not finished yet, or there exists no session with identifier π, $\perp$ is returned.
- `Corrupt`(T_i): on input a tag reference T_i, this oracle returns the complete internal state of T_i. Note that the adversary is not given control over T_i.

B Privacy Preserving Signatures

To obtain privacy preserving signatures with identification we make a slight modification to the Schnorr signature scheme [15]. The original Schnorr signature scheme works as follows:

- $r \in_R \mathbb{Z}_l$
- $e = H(M||rP)$, $s = ex + r$
- Output s, e.

In the modified scheme, rY, instead of e is provided together with s.

- $r \in_R \mathbb{Z}_l$
- $e = H(M||rP)$, $s = ex + r$
- Output s, rY.

The verifier can retrieve $rP = y^{-1}(rY)$ and as such compute e. By computing $e^{-1}(sP - rP) = X$, s is verified. By checking the database for a registered public key X, one obtains both identification and verification of the signature at the same time, provided the number of tags remains significantly lower than ℓ.

Privacy of this modified scheme can be shown under the DDH assumption. Existential unforgeability follows in the same way as for the Schnorr signature when the verifier is provided with y.

Privacy Preserving Payments on Computational RFID Devices with Application in Intelligent Transportation Systems*

Gesine Hinterwälder[1], Christof Paar[1,2], and Wayne P. Burleson[1]

[1] Department of Electrical and Computer Engineering,
University of Massachusetts Amherst
{hinterwalder,burleson}@ecs.umass.edu
[2] Horst Görtz Institute for IT Security, Ruhr-University Bochum, Germany
christof.paar@rub.de

Abstract. Electronic cash is a suitable solution for payment systems, in which the user's identity should not be revealed during the payment. This is for example the case for public transportation payment systems. One electronic cash scheme, efficient during the spending phase, was proposed by Brands'. This scheme, as all privacy-preserving payment schemes, is based on public-key cryptography. However, payment devices used in those systems need to be cheap and low-power, which restricts their computational performance. These two points conflict with the need for payments to be executed quickly, in order to avoid delays at the entrance points of the system. In this work we demonstrate that using sophisticated implementation techniques, it is possible to realize full-size e-cash schemes even on inexpensive payment tokens. We present a full implementation of Brands' offline cash scheme for the UMass Moo, a computational RFID-token. The spending protocol, which is the time critical part in transportation payment systems can be executed in 13 ms. This meets real-world application requirements. The reloading of the card, which is less time critical, as it is conducted offline, is time consuming. We discuss solutions to this problem.

Keywords: Electronic Cash, Elliptic Curves, Privacy-preserving payments, Computational RFID, Transportation payment systems.

1 Introduction

The majority of electronic payment systems in use, e.g., credit-card transactions secured via SSL/TLS, dedicated schemes à la PayPal, or electronic bank transfers (which are popular in Europe for e-business) provide a reasonable level of security, but are not anonymous. They leave an electronic trail which can potentially be exploited and abused. *Electronic cash*, or *e-cash*, refers to technologies

* This work is supported by the NSF under CNS-0964641. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 109–122, 2013.

which allow for secure payments that at the same time preserve the privacy of users. Even though some e-cash solutions were developed in the 1990s, they have not been deployed on a large scale. We speculate that one reason for this development is that Internet-based e-business does not require a high level of anonymity. After all, many consumers are also quite comfortable with buying goods in stores with (decisively non-anonymous) credit cards. However, the situation is different in intelligent transportation systems. We focus on a rather concrete case of public transportation systems. For various reasons — including reduced environmental impact and reduced traffic congestion — public transportation has drawn increased attention in recent years. Electronic payments can provide throughput and convenience advantages in public transportation. Furthermore they help to reduce payment collection costs, enable dynamic pricing, and facilitate maintenance of a payment system. Additionally, the use of electronic payments enables easy collection of meaningful data about customer behavior. This helps to improve a system and tailor it to customer needs. However, anonymity is also highly desirable since revealing the user's identity impacts her current location privacy and potentially allows to derive fine grain patterns of movement and habits. Thus, electronic cash is a highly attractive technology.

One electronic cash scheme, which is efficient during the spending phase, is the one due to Brands. However, as all privacy-preserving payment schemes, Brands' scheme requires the execution of one or more computationally expensive public-key operations. This conflicts with the set-up of public transportation payment systems. Even though the use of devices such as smart phones is possible, it is often desirable to have (extremely) inexpensive payment tokens. This greatly increases the number of users that can participate and also provides ease-of-use. Inexpensive tokens can provide an experience similar to traditional paper tickets, while greatly adding convenience. Another central requirement for payment tokens is that they are contactless because contact-based cards conflict with the need for high throughput, especially during rush hours. In the work at hand we demonstrate that through optimized implementation techniques we can combine the conflicting features of high computational requirements with inexpensive contactless RFID tokens.

The e-cash concept describes interaction between three entities: the Bank $\mathcal{B}$, the User $\mathcal{U}$, and the Shop $\mathcal{S}$. $\mathcal{B}$ issues electronic coins to $\mathcal{U}$. An electronic coin is a piece of data blindly signed by $\mathcal{B}$. In this data the serial number of the coin as well as the account number of $\mathcal{U}$ is encoded. $\mathcal{U}$ can prove ownership of a valid coin to $\mathcal{S}$, while keeping her identity hidden. This can happen in an offline scenario, where $\mathcal{S}$ is not connected to a database of $\mathcal{B}$. $\mathcal{S}$ can later deposit the coin to his bank account. $\mathcal{B}$ can check if the coin has been spent before and whether $\mathcal{S}$ tried to deposit the same coin twice or whether $\mathcal{U}$ spent the same coin twice. In case it is identified that $\mathcal{U}$ has spent a coin twice, her identity can be revealed. An additional feature of some e-cash schemes, including Brands' scheme, is that user information can be encoded into a coin. That way some information about the user can be provided, while other information remains private. This information could for example be the age of a user. In transportation payment systems this

feature can be used to gather meaningful data about the system and to allow fare price adaption. For example to allow cheaper rides for students.

We present an implementation of Brands' payment protocol for a constrained RFID token, the *Moo* device [21]. The Moo is a computational RFID tag designed at the University of Massachusetts Amherst. It operates in the UHF band with a center frequency of 915 MHz and is passively powered, i.e., it harvests its energy from the RF field presented by the RFID reader that it communicates with. The communication between the Moo and an RFID reader is based on the EPC Class 1 Generation 2 protocol. The Moo can communicate over a distance of up to several meters. Even though it is an experimental device, it is a close approximation of platforms that could be provided at low-cost in future payment tokens. Based on market prices of some contactless smartcards, it can be assumed that if mass-produced, the price for a Moo will be in the range of a few dollars. The Moo integrates the MSP430F2618, an ultra-low power MCU from Texas Instruments. The MSP430F2618 has a 16-bit RISC processor, 8 KB of RAM and 116 KB of flash memory. Beneficial for our implementation is the multiply-and-accumulate instruction, featured by this microcontroller ([12], [13]). This instruction supports a 16x16 bit multiplication and accumulates the 32-bit results. The use in our implementation will be further described in Section 3.1.

The most time critical parts of the scheme are the withdrawal and the spending step. Spending should be executable in a couple of hundreds of milliseconds in order to avoid delays for the customer due to congestions. The withdrawal is a little less time critical, but should also not take longer than a couple of seconds. As mentioned above the payment device will integrate cheap hardware, whereas the devices executing the payment on the transportation authorities side can be equipped with powerful hardware, e.g., a 32-bit ARM CPU or even a PC-like platform. The challenge at hand is thus to realize the public-key intensive protocol on the token. Hence, we put our focus on the users withdrawal and spending step.

1.1 Related Work

The application domain of public transportation payment systems as an area of cryptographic research interest had been suggested by [11]. It differs from other application areas in that fare collection cost needs to be kept very low, while allowing for sufficiently secure payments. The authors propose the use of recent advances in anonymous credentials and e-cash systems, which can detect fraud while maintaining the anonymity of the honest user.

In [20] electronic payments in public transport are discussed from a systems perspective. A system model is described and its security is evaluated based on several trust models. The argumentation is based on the idea that public key cryptography cannot be executed on low power devices such as RFID devices. We disagree with this idea. We think that using an efficient protocol, combined with new ideas of implementing public key cryptosystems will enable the use of electronic cash for public transportation payments.

An implementation of a payment scheme that is based on Brands' offline cash scheme is shown in [5]. A Sharp-Zaurus5600 PDA is used as the target platform. This is quite a powerful platform compared to what can be expected as payment devices for public transport ticketing. Instead we use a passively powered device that approximates cheap payment tokens.

There have been implementations of anonymous credentials on Java Cards as for example [1] and [2]. The authors tailor the protocol to the platform since no direct access to the integrated hardware is possible. Instead we implement and evaluate a full-size e-cash scheme and avoid any changes to the protocol.

An implementation of an ECC framework on the Intel WISP, the predecessor of the UMass Moo, has been shown in [17]. Our results are not limited to an implementation of the mathematical framework, instead we show how a privacy-preserving payment protocol can be fully implemented.

Our Contribution. We implemented Brands' e-cash scheme for an ultra-low power microcontroller from Texas Instruments, integrated on the Moo computational RFID, namely the MSP430F2618. The scheme is implemented in C using the IAR Embedded Workbench for TI MSP430 compiler. Time-critical functions have been speeded up using assembly language. Leaving further options for optimizations, this implementation proves that it is possible to implement e-cash schemes on ultra-low power devices. Our approach is different than other implementation proposals of privacy preserving payment protocols, in that we do not tailor a protocol to available hardware. We do not compromise security or privacy in order to achieve high performance, but try to give an idea, which hardware would be necessary in order to achieve high security and privacy standards, while satisfying real-time application requirements for transportation payment systems, which are in the order of hundreds of milliseconds.

Organization. The paper is structured as follows: In Section 2 we give an overview of the protocol from an implementation perspective. Here we concentrate on the parts necessary for a discussion of the implementation. In Section 3 we describe the implementation and the methods we used to achieve an acceptable performance. We present and evaluate our results in Section 4 and conclude with Section 5.

2 Brands' Untraceable Offline Cash Scheme

In [4] Stefan Brands proposed an untraceable offline cash scheme. The scheme includes multiple protocols between the three entities user $\mathcal{U}$, bank $\mathcal{B}$ and shop $\mathcal{S}$. The first protocol registers $\mathcal{U}$ to $\mathcal{B}$ and sets up an account. $\mathcal{B}$ generates an account number and stores it together with some identifying information of $\mathcal{U}$. The second protocol, which is called the withdrawal protocol, handles $\mathcal{U}$'s withdrawal of electronic coins from $\mathcal{B}$. After the execution of this protocol, $\mathcal{U}$ knows a valid representation of a coin, that is blindly signed by $\mathcal{B}$ and cannot be duplicated by $\mathcal{U}$. The third protocol describes the spending scheme between $\mathcal{U}$ and $\mathcal{S}$. $\mathcal{S}$ checks whether the received data represents a valid coin and obtains

some data from $\mathcal{U}$ that it needs later on to deposit the coin to $\mathcal{B}$. The last protocol describes how $\mathcal{S}$ can deposit coins to $\mathcal{B}$ that it previously received from $\mathcal{U}$. For public transportation payment systems, $\mathcal{B}$ and $\mathcal{S}$ belong to the same entity, namely the transportation authority. We will however differentiate between them, as they might represent different hardware.

Allowing payments to be verified in an offline fashion is highly desirable for public transportation payment systems, where the hardware accepting the payment ($\mathcal{S}$) might not be connected to the database of the transportation authority ($\mathcal{B}$) at all times. However, the method of comparing the received data to previously received data, to check whether a coin has been spent before, is not applicable in this scenario. Instead Brands' scheme reveals a crime after the fact. The received coin is compared to previously received coins in the database. In case a user spends the same coin twice, her identity is revealed. Yet when using the system correctly, the user's identity remains hidden.

The implementation focuses on the withdrawal and the spending part, as those are the time-critical parts that have to be executed frequently and quickly, to not impair customer convenience. While the execution of the withdrawal protocol is more efficient on $\mathcal{B}$'s side, i.e. only two exponentiations need to be executed on $\mathcal{B}$'s side while twelve need to be executed on $\mathcal{U}$'s side, the spending protocol is extremely efficient on $\mathcal{U}$'s side, i.e. seven exponentiations need to be executed on $\mathcal{B}$'s and no exponentiation has to be executed on $\mathcal{U}$'s side. In the following the parts of the withdrawal and spending protocol of Brands' offline cash scheme, that are important for a discussion of the implementation, will be described. For a full description of the scheme the reader is referred to [4].

Brands scheme is based on a combination of a primitive that he calls restrictive blind signatures and the representation problem in groups of prime order. On setup of the system, $\mathcal{B}$ decides on a group G_q of prime order and two hash functions $\mathcal{H}, \mathcal{H}_0$. It picks three elements g, g_1, g_2 from G_q, chooses a secret key $x \in_{\mathcal{R}} \mathbb{Z}_q^*$, and calculates the value $h = g^x$. When opening an account, the user picks a secret key u_1 and calculates $I = g_1^{u_1}$, which is stored by the bank as the users account number. Before withdrawal the user first needs to proof ownership of his account number to the bank. Then for each coin the withdrawal protocol shown in Table 1 is executed between the user $\mathcal{U}$ and the bank $\mathcal{B}$ [4].

A coin in Brands' protocol consists of the tuple $A, B, \text{sign}(A, B)$, where $(A, B) \in G_q \times G_q$ and $\text{sign}(A, B)$ consists of the elements $(z', a', b', r') \in G_q \times G_q \times G_q \times \mathbb{Z}_q$. $A, B, \text{sign}(A, B)$ is a valid coin, if

$$g^{r'} = h^{\mathcal{H}(A,B,z',a',b')} a' \quad \text{and} \quad A^{r'} = z'^{\mathcal{H}(A,B,z',a,b')} b' \tag{1}$$

holds. After the execution of the withdrawal protocol, the user knows a representation of that coin, which he will have to prove to $\mathcal{S}$, when spending the coin. The spending protocol between the user $\mathcal{U}$ and the shop $\mathcal{S}$ is shown in Table 2. Here I_S is the identifying information of the shop. The verification of $\text{sign}(A, B)$ is done by checking whether Equation 1 holds.

Table 1. Brands' withdrawal protocol

$\mathcal{U}$		$\mathcal{B}$
		$w \in_{\mathcal{R}} \mathbb{Z}_q$
		$a \leftarrow g^w$
$s \in_{\mathcal{R}} \mathbb{Z}_q^*$	$\xleftarrow{a,b}$	$b \leftarrow (Ig_2)^w$
$A \leftarrow (Ig_2)^s$		
$z' \leftarrow z^s$		
$x_1, x_2, u, v \in_{\mathcal{R}} \mathbb{Z}_q$		
$B \leftarrow g_1^{x_1} g_2^{x_2}$		
$a' \leftarrow a^u g^v$		
$b' \leftarrow b^{su} A^v$		
$c' \leftarrow \mathcal{H}(A, B, z', a', b')$		
$c \leftarrow c'/u \mod q$	$\xrightarrow{c}$	
	$\xleftarrow{r}$	$r \leftarrow cx + w \mod q$
$g^r \stackrel{?}{=} h^c a$		
$(Ig_2)^r \stackrel{?}{=} z^c b$		
$r' \leftarrow ru + v \mod q$		

Table 2. Brands' spending protocol

$\mathcal{U}$		$\mathcal{S}$
	$\xrightarrow{A,B,sign(A,B)}$	$A \stackrel{?}{\neq} 1$
$r_1 \leftarrow d(u_1 s) + x_1 \mod q$	$\xleftarrow{d}$	$d \leftarrow \mathcal{H}_0(A, B, I_{\mathcal{S}}, \text{date/time})$
$r_2 \leftarrow ds + x_2 \mod q$	$\xrightarrow{(r_1,r_2)}$	
		Verify sign(A, B)
		$g_1^{r_1} g_2^{r_2} \stackrel{?}{=} A^d B$

In the following section the implementation of Brands' offline cash scheme [4] will be described.

3 Implementation

We base the scheme on Elliptic Curve Cryptography (ECC), since it is the most efficient asymmetric cryptography scheme, compared to others as for example Discrete Logarithm (DL) based systems. This is due to the fact that, in contrast to other public-key cryptosystems, the run-time of known attacks on ECC-based cryptosystems grows exponentially with the bit length of the curve, i.e. the same security level can be achieved with much shorter key lengths [19]. Thus less calculations are necessary and less data needs to be communicated between the protocol partners. This makes the use of ECC desirable on platforms that are constrained in power and such in computational performance.

We use an elliptic curve in short Weierstrass representation. The points of the elliptic curve over the prime field $\mathbb{F}_p$, i.e. the points that satisfy the equation

$$E : y^2 = x^3 + ax + b, \tag{2}$$

where $a, b \in \mathbb{F}_p$, such that $4a^3 + 27b^2 \neq 0 \mod p$, and the point at infinity form an additive abelian group. On the set of these points, two operations are defined, namely point addition and point doubling. Repeated execution of the point addition is called scalar multiplication $\boldsymbol{Q}= k[\boldsymbol{P}]$, where the point $\boldsymbol{P}= (x, y)$ is multiplied with the scalar k. The Elliptic Curve Discrete Logarithm Problem (ECDLP) states that given two points $\boldsymbol{P}$ and $\boldsymbol{Q}$, it is hard to solve the equation $\boldsymbol{Q}= k[\boldsymbol{P}]$ for the scalar k. The security of ECC-based cryptosystems is based on this observation. The point multiplication is the core operation of ECC, whereas in DL-based systems it is the exponentiation. For the implementation of the protocols presented in Table 1 and 2 an exponentiation in G_q is equivalent to a point multiplication on the elliptic curve E. Similarly a multiplication in G_q is equivalent to a point addition on E.

The choice of the elliptic curve needs to be tailored to the scheme that is implemented. The level of security that is achieved is determined by the effort that is necessary to break the system. As we target a micro-payment system, we conclude from [3] that basing the system on a 160-bit curve, presents sufficient security, i.e. the efforts necessary for breaking the system are high compared to the benefits that would be gained from it. To further increase security, the keys could be used for limited time only, i.e. a ticket could expire after a certain amount of time. We base our implementation on a standardized 160-bit prime curve suggested by [18], namely secp160r1. This curve is based on a special prime, which allows for a very efficient implementation of the reduction, as will further be discussed in Section 3.1. The group spanned by the points of E that can be reached from the base point G, form a cyclic subgroup, suitable for an implementation of Brands' offline cash scheme, as the order of this group is prime.

3.1 $GF(p)$ Framework

We use a curve defined over the prime field $\mathbb{F}_p$ with $p = 2^{160} - 2^{31} - 1$. Hence a $GF(p)$ framework had to be implemented that the ECC framework could be based on. Since the MSP430F2618 has a 16-bit CPU, an element in $\mathbb{F}_p$ is represented as an array of ten words. We put our focus on the optimization of multiplication and squaring in $\mathbb{F}_p$, as those will be used extensively in the implementation of the point multiplication in the ECC-framework.

Both functions implement two steps. In the first step the hybrid multiplication algorithm with $d = 2$ as proposed in [9] is used to calculate a double-sized array, as the result of a multi-precision multiplication or multi-precision squaring of the input arrays. This can be efficiently implemented, making use of the multiply-and-accumulate instruction of the MSP430F2618. To achieve better performance, we implemented these functions in assembly. Further the resulting

array is reduced making use of what is explained as fast reduction of NIST-primes in [10].

To illustrate the implementation of the reduction step, let us assume the calculation of $e = c^2 \mod p$. The number c is stored as an array of ten 16-bit words. This can be expressed as

$$\begin{aligned} c =& c_9 2^{144} + c_8 2^{128} + c_7 2^{112} + c_6 2^{96} + c_5 2^{80} + c_4 2^{64} \\ &+ c_3 2^{48} + c_2 2^{32} + c_1 2^{16} + c_0, \end{aligned}$$

where the coefficient c_i represents the integer that is stored in the i-th element of the array. The result of this squaring step, which we are going to call d, will be an array of 20 elements, which can be represented as

$$\begin{aligned} d =& d_{19} 2^{304} + d_{18} 2^{288} + d_{17} 2^{272} + d_{16} 2^{256} + d_{15} 2^{240} + d_{14} 2^{224} + d_{13} 2^{208} \\ &+ d_{12} 2^{192} + d_{11} 2^{176} + d_{10} 2^{160} + d_9 2^{144} + d_8 2^{128} + d_7 2^{112} + d_6 2^{96} \\ &+ d_5 2^{80} + d_4 2^{64} + d_3 2^{48} + d_2 2^{32} + d_1 2^{16} + d_0. \end{aligned}$$

The reduction step, which leads to the final result $e = c^2 \mod p$ calculates

$$\begin{aligned} e =& [d_9 + d_{19} + (d_{17} >> 1) + (d_{18} << 15)] 2^{144} \\ &+ [d_8 + d_{18} + (d_{16} >> 1) + (d_{17} << 15)] 2^{128} \\ &+ [d_7 + d_{17} + (d_{15} >> 1) + (d_{16} << 15)] 2^{112} \\ &+ [d_6 + d_{16} + (d_{14} >> 1) + (d_{15} << 15)] 2^{96} \\ &+ [d_5 + d_{15} + (d_{13} >> 1) + (d_{14} << 15)] 2^{80} \\ &+ [d_4 + d_{14} + (d_{12} >> 1) + (d_{13} << 15)] 2^{64} \\ &+ [d_3 + d_{13} + (d_{11} >> 1) + (d_{12} << 15) + (d_{19} >> 2)] 2^{48} \\ &+ [d_2 + d_{12} + (d_{10} >> 1) + (d_{11} << 15) + (d_{18} >> 2) \\ &+ ((d_{19} << 14) \ \& \ \mathrm{C000}_{16})] 2^{32} \\ &+ [d_1 + d_{11} + (d_{10} << 15) + (d_{10} << 5) + (d_{19} >> 1) \\ &+ ((d_{18} << 14) \ \& \ 8000_{16})] 2^{16} \\ &+ d_0 + d_{10} + (d_{19} << 15) + (d_{18} >> 1), \end{aligned}$$

where $>>$ stands for a logical right shift and $<<$ a logical left shift of the coefficient, and & stands for a bitwise AND.

3.2 ECC Framework

Points on elliptic curves can be represented in various coordinate systems. When choosing affine coordinates, an inversion **I** in the underlying prime field of the curve is needed to calculate a point doubling or a point addition. An inversion over a prime field is often the most time critical function of an implementation. If the modular multiplication **M** in the underlying prime field can be implemented

much faster than the inversion **I**, which is the case in our implementation, as can be seen in Table 3, it is beneficial to use projective coordinates, since then the modular inversion is exchanged for multiple modular multiplications **M** and squarings **S**.

In the presented implementation Jacobian coordinates were used. Each coordinate represents an element in the underlying field $GF(p)$. Whereas a point in affine coordinates is represented by two coordinates ($\boldsymbol{P}= (x, y)$), the representation in Jacobian coordinates requires a third coordinate ($\boldsymbol{P}= (X, Y, Z)$). The point $\boldsymbol{P}$ has multiple representations in Jacobian coordinates, namely as many as possible coordinates Z. The point (X, Y, Z) in Jacobian coordinates is equivalent to the point $(x, y) = (X/Z^2, Y/Z^3)$ in affine coordinates.

In [14] Meloni proposed an optimization for the point addition in Jacobian coordinates for the case, when two points share the same Z-coordinate, the so called co-Z addition. This method requires less multiplications and squarings, namely only $5\mathbf{M}+2\mathbf{S}+7\mathbf{A}$ instead of $12\mathbf{M}+4\mathbf{S}+7\mathbf{A}$, which is required for the regular Jacobian point addition[19]. The co-Z addition can be calculated as:

$$X_3 = D-(B+C), \quad Y_3 = (Y_2-Y_1)(B-X_3)-E \quad \text{and} \quad Z_3 = Z(X_2-X1), \tag{3}$$

where $A = (X_2-X_1)^2$, $B = X_1A$, $C = X_2A$, $D = (Y_2-Y_1)^2$ and $E = Y_1(C-B)$.

An algorithm for point multiplication, secure against side-channel attacks, is the Montgomery powering ladder proposed in [16]. The algorithm requires a point addition followed by a point doubling for every loop iteration. In contrary the double-and-add algorithm requires a point doubling in every iteration, but a point addition only, if the current bit in the scalar is a one. This leaks out information about the secret key [7].

Generally the Z-coordinate of the resulting point of a point addition $\boldsymbol{P}+\boldsymbol{Q}$ has a different value than the Z-coordinate of the original points $\boldsymbol{P}$ and $\boldsymbol{Q}$, which makes repeated use of the co-Z addition impossible. However the Z-coordinate of the original point $\boldsymbol{P}$ can be updated, such that $\boldsymbol{P}$ shares the same Z-coordinate with $\boldsymbol{P}+\boldsymbol{Q}$, as has been suggested by Meloni [14]. This is done using

$$B = X_1A = X_1(X_2 - X_1)^2 = x_1Z_3^2 \quad \text{and} \quad E = Y_1(X_2 - X_1)^3 = y_1Z_3^3 \tag{4}$$

from Equation 3 and updating the point $\boldsymbol{P}$ to (E, B, Z_3) [19]. Furthermore $\boldsymbol{P}-\boldsymbol{Q}$ can be obtained simultaneously to $\boldsymbol{P}+\boldsymbol{Q}$, such that both results share the same Z-coordinate, by [19]:

$$X_3' = F - (B + C) \quad \text{and} \quad Y_3' = (Y_1 + Y_2)(X_3' - B) - E \tag{5}$$

where the variables are defined as in 3 and $F = (Y_1 + Y_2)^2$. In [19] the addition, where $\boldsymbol{P}+\boldsymbol{Q}$ and $\boldsymbol{P}-\boldsymbol{Q}$ is calculated simultaneously is called co-Z conjugate point addition. It requires $6\mathbf{M}+3\mathbf{S}+16\mathbf{A}$.

The co-Z addition and the conjugate co-Z addition can be used to implement the Montgomery powering ladder. The loop iteration of the Montgomery powering ladder requires the calculation of a point addition followed by a point doubling, i.e. $\boldsymbol{Q}=\boldsymbol{P}+\boldsymbol{Q}$ and $\boldsymbol{P}= 2\boldsymbol{P}$. This can be accomplished by calculating

the co-Z conjugate point addition followed by the co-Z point addition, since this leads to $\boldsymbol{P}+\boldsymbol{Q}+\boldsymbol{P}-\boldsymbol{Q}=2\boldsymbol{P}$ and $\boldsymbol{P}+\boldsymbol{Q}$.

As can be seen from 3 and 5, the Z-coordinate is not required for the calculation of the X-coordinate and the Y-coordinate, which further saves multiplications. Yet, the Z-coordinate is needed to transform the resulting point back to affine coordinates, at the end of the algorithm. If a point is given in affine (x, y) and in Jacobian (X, Y, Z) coordinates, where the Z-coordinate of the Jacobian representation is unknown, the Z-coordinate of the Jacobian representation can be calculated as:

$$Z = xY/yX \tag{6}$$

The interested reader is referred to [19] for detailed explanations, how this can be used to recover the Z-coordinate of the resulting point at the end of the Montgomery ladder. The idea is that after each iteration the difference between the points stored in $\boldsymbol{P}$ and $\boldsymbol{Q}$ is the input point $\boldsymbol{P}$. Together with the affine representation of $\boldsymbol{P}$, the Z-coordinate can be recovered. These ideas combined can be used to achieve an efficient algorithm for point multiplication, which is shown as Algorithm 9 in [19].

3.3 $GF(q)$ Framework

In comparison to the ECC functions $GF(q)$ functions are much less computationally intensive. Hence the performance of the $GF(q)$ framework is not as critical. The order of the chosen curve is prime. However the prime is not of a form suitable to use special reduction techniques. If this is not the case, reduction can be very time consuming. We implemented Barett reduction as presented in [15].

3.4 Hash Function

During the setup of Brands' offline cash scheme the bank decides on two hash functions $\mathcal{H}$ and $\mathcal{H}_0$, with [4]:

$$\mathcal{H}: G_q \times G_q \times G_q \times G_q \times G_q \rightarrow \mathbb{Z}_q^*$$
$$\mathcal{H}_0: G_q \times G_q \times \text{SHOP-ID} \times \text{DATE/TIME} \rightarrow \mathbb{Z}_q$$

Hence for our implementation a hash function is needed that takes as input several points on the curve and calculates as output an element in $\mathbb{Z}_q^*$, where q is the order of the basepoint $\boldsymbol{G}$ of the chosen curve. To ensure that the output of the hash function lies in $\mathbb{Z}_q^*$, we seek for a 160-bit output instead of 161 bits, which is the bit-length of q for our case. As a coordinate on the elliptic curve is 160 bit long, we seek for a hash function that takes 160-bit inputs and produces a 160-bit output.

We implemented the block cipher based hash function AES-hash [6]. Inputs are hashed blockwise, by using the intermediate result as the input and the next input block as the key to the block cipher. The block ciphers output is XORed with the intermediate result. We used the proposed block cipher Rijndael [8] for our implementation, since this can be implemented for different key and block lengths.

4 Results

We used the IAR Embedded Workbench for MSP430 v 5.40 for our implementation. In this section the simulation results will first be presented. Together with this presentation an evaluation of the applicability of the implementation to transportation payment systems will be given.

In Table 3 timings for the implementation of the different frameworks and the hash functions are given. We put our focus on the optimization of the time critical parts of the implementation, which are the $GF(p)$ and the ECC frameworks, leaving further options for optimization of the $GF(q)$ framework and the hash functions. The results mirror the design considerations presented in this paper. As expected using a special prime leads to a very efficient reduction, and such to a fast execution of the modular multiplication and squaring. Even though we used the binary algorithm for inversion in $\mathbb{F}_p$, which is Algorithm 2.22 in [10], the execution time of an inversion is much longer than the execution time of a multiplication. The results show that a point multiplication is by far the most time-consuming computation.

The results have been simulated for two frequencies, which are 4 MHz and 16 MHz. This implementation is designed for the UMass Moo, which operates at 4 MHz. Nonetheless the MSP430F2618 can run at a maximum frequency of 16 MHz. The results at 16 MHz represent execution times that would be possible, if the platform supports an operating frequency of 16 MHz. This can be achieved by providing the platform with a power-source in addition to the power it gets from harvesting the RF-field. In the presented scheme the computationally challenging part is the withdrawal part, whereas spending is very efficient. A device can be imagined that could be operated in contact and contactless manner. When in contact manner it could be provided with enough energy to operate it at 16 MHz.

Table 3. Timings of implementation of $GF(p)$, $GF(q)$, and ECC framework, and the Hash functions $\mathcal{H}$ and $\mathcal{H}_0$

Function	Cycle count	Execution time @16 MHz in miliseconds	Execution time @4 MHz in miliseconds
Reduction in $\mathbb{F}_p$	384	0.024	0.096
Multiplication in $\mathbb{F}_p$	2,266	0.142	0.567
Squaring in $\mathbb{F}_p$	1,678	0.105	0.420
Inversion in $\mathbb{F}_p$	190,294	11.9	47.6
Reduction in $\mathbb{F}_q$	11,648	0.728	2.91
Multiplication in $\mathbb{F}_q$	17,101	1.07	4.27
Rijndael160	10,785	0.67	2.69
$\mathcal{H}_0$	44,031	2.75	11
$\mathcal{H}$	54,958	3.43	13.7
Point addition	196,059	12.3	49
Point multiplication	6,312,785	395	1,578

Table 4 shows the execution times of the user side of Brands' scheme. For the spending step we aim for an execution time of a couple of hundreds of milliseconds. This can be achieved on the UMass Moo. The withdrawal step is less time critical, but should also not take longer than a couple of seconds. A device would be favourable that could operate in contactless and contact manner. The spending step is very time critical. Hence, an execution in contactless mode would be favourable. However, when withdrawing coins, the device could be connected to a machine, which allows for the use of more powerful hardware, or additional hardware accelerators.

Table 4. Timing results of the user sides' implementation of Brands' offline cash protocol on the MSP4302618

Cycle count Withdrawal	77,292,874
Cycle count Spending	52050
Execution time Withdrawal in seconds @ 4 MHz	19.3
Execution time Spending in seconds @ 4 MHz	0.013
Execution time Withdrawal in seconds @ 16 MHz	4.83
Execution time Spending in seconds @ 16 MHz	0.0033

Table 5. Code size of the implementation of Brands' protocol on the MSP430F2618

	CODE in bytes	CONST in bytes	DATA in bytes
GF(p) framework	5,912	228	20
GF(q) framework	2,308	158	22
ECC framework	3,088	120	20
Hash function	2,578	308	256
Protocol User	560	-	-

5 Conclusion

Electronic cash combines the benefits of representing a secure payment scheme, while allowing for a similar level of privacy as physical cash does. This makes the use of e-cash schemes especially suitable for payment systems, where the user's identity should remain hidden. In this paper we analysed the applicability of one specific e-cash scheme to public transportation payment systems. These payment systems additionally require fast payment execution times, while using extremely cheap payment devices. Since e-cash schemes are based on public-key cryptography, they have long been too computationally intensive, to be suitable for an application domain, where payments are executed on extremely cheap hardware. In this paper a full implementation of Brands' offline cash scheme for a computational RFID-tag has been presented. Spending is considered the time-sensitive operation as this happens during the actual transportation, e.g.,

at a turnstile during rush hour. The achieved spending timings meet real-world application requirements even for high throughput transportation situations. The withdrawal of coins could be done at home, where the user connects the payment token to his personal computer, and leaves it there for the charging period. In that case the withdrawal would not be very time-critical. Yet, there are several advantages of the withdrawal taking place at charging stations located near the entrance points of the public transportation system. In that case, the withdrawal (even of several coins) needs to be executable in a couple of seconds. Yet, during charging the payment device could be connected to the charging station, which supplies the passive tag with sufficient energy to power additional hardware accelerators.

Acknowledgements. We thank Andy Rupp and Andrés Molina-Markham for their input to this work.

References

1. Batina, L., Hoepman, J.-H., Jacobs, B., Mostowski, W., Vullers, P.: Developing Efficient Blinded Attribute Certificates on Smart Cards via Pairings. In: Gollmann, D., Lanet, J.-L., Iguchi-Cartigny, J. (eds.) CARDIS 2010. LNCS, vol. 6035, pp. 209–222. Springer, Heidelberg (2010)
2. Bichsel, P., Camenisch, J., Groß, T., Shoup, V.: Anonymous credentials on a standard java card. In: ACM Conference on Computer and Communications Security, pp. 600–610 (2009)
3. Bos, J.W., Kaihara, M.E., Kleinjung, T., Lenstra, A.K., Montgomery, P.L.: On the Security of 1024-bit RSA and 160-bit Elliptic Curve Cryptography. IACR Cryptology ePrint Archive, 2009:389 (2009)
4. Brands, S.: Untraceable Off-line Cash in Wallets with Observers (Extended Abstract). In: Stinson, D.R. (ed.) CRYPTO 1993. LNCS, vol. 773, pp. 302–318. Springer, Heidelberg (1994)
5. Clemente-Cuervo, E., Rodríguez-Henríquez, F., Arroyo, D.O., Ertaul, L.: A PDA Implementation of an Off-line e-Cash Protocol. In: Security and Management, pp. 452–458 (2007)
6. Cohen, B.: AES-hash (2001), http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/aes-hash/aeshash.pdf
7. Coron, J.-S.: Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In: Koç, Ç.K., Paar, C. (eds.) CHES 1999. LNCS, vol. 1717, pp. 292–302. Springer, Heidelberg (1999)
8. Daemen, J., Rijmen, V.: The Rijndael Block Cipher (1999), http://ftp.csci.csusb.edu/ykarant/courses/w2005/csci531/papers/Rijndael.pdf
9. Gura, N., Patel, A., Wander, A., Eberle, H., Shantz, S.C.: Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In: Joye, M., Quisquater, J.-J. (eds.) CHES 2004. LNCS, vol. 3156, pp. 119–132. Springer, Heidelberg (2004)
10. Hankerson, D., Menezes, A.J., Vanstone, S.: Guide to Elliptic Curve Cryptography. Springer-Verlag New York, Inc., Secaucus (2003)
11. Heydt-Benjamin, T.S., Chae, H.-J., Defend, B., Fu, K.: Privacy for Public Transportation. In: Danezis, G., Golle, P. (eds.) PET 2006. LNCS, vol. 4258, pp. 1–19. Springer, Heidelberg (2006)

12. T. I. Incorporated. MSP430x2xx Family User's Guide (Rev. H) (2011)
13. T.I. Incorporated. MSP43F261x Mixed Signal Microcontroller (2011)
14. Meloni, N.: New Point Addition Formulae for ECC Applications. In: Carlet, C., Sunar, B. (eds.) WAIFI 2007. LNCS, vol. 4547, pp. 189–201. Springer, Heidelberg (2007)
15. Menezes, A., van Oorschot, P.C., Vanstone, S.A.: Handbook of Applied Cryptography. CRC Press (1996)
16. Montgomery, P.L.: Speeding the Pollard and Elliptic Curve Methods of Factorization. In: Mathematics of Computation, pp. 243–264 (1987)
17. Pendl, C., Pelnar, M., Hutter, M.: Elliptic Curve Cryptography on the WISP UHF RFID Tag. In: Juels, A., Paar, C. (eds.) RFIDSec 2011. LNCS, vol. 7055, pp. 32–47. Springer, Heidelberg (2012)
18. Certicom Research. SEC 2: Recommended elliptic curve domain parameters. In: Standards for Efficient Cryptography (2000), http://www.secg.org/download/aid-386/sec2-final.pdf
19. Rivain, M.: Fast and Regular Algorithms for Scalar Multiplication over Elliptic Curves. IACR Cryptology ePrint Archive, 2011:338 (2011)
20. Sadeghi, A.-R., Visconti, I., Wachsmann, C.: User Privacy in Transport Systems Based on RFID E-Tickets. In: PiLBA (2008)
21. Zhang, H., Gummeson, J., Ransford, B., Fu, K.: Moo: A Batteryless Computational RFID and Sensing Platform (2011), https://web.cs.umass.edu/publication/docs/2011/UM-CS-2011-020.pdf

Weakening ePassports through Bad Implementations

Luigi Sportiello

European Commission
Joint Research Centre
Via Enrico Fermi 2749, 21027 Ispra(VA), Italy
luigi.sportiello@jrc.ec.europa.eu

Abstract. Different countries issue an electronic passport embedding a contactless chip that stores the holder data (ePassport). To prevent unauthorized reading of the sensitive information present on such chip an access control mechanism based on symmetric cryptography, the Basic Access Control (BAC), has been introduced. In this work we present the flaws we have found out in some implementations of the software hosted on ePassport chips and how BAC is affected. In particular we show how it is possible to discern the different software versions used on the chip over time through some their peculiar fingerprints. This information can be used to shrink the BAC keys space making the protocol weaker. In addition, we show the presence of a defective function to exchange random material during the BAC procedure that opens a door for a hypothetical MITM attack. The results of this paper could be exploited as a first guide for reviewing and refining existing ePassport implementations.

Keywords: ePassport, Basic Access Control, Key Space, Man in the Middle.

1 Introduction

The passport is the international travel document used for people identification at border crossings [1]. Such document represents an important tool for preventing that unauthorized people and criminals freely move among countries. For this reason, over the years, a lot of effort has been put into the development of passports featured by an ever higher trustworthiness, for instance adopting new papers, studying ink-based techniques and adopting holograms to make harder the production of fake documents and easier their detection.

With this aim in mind and the idea to strengthen the connection between the document and its bearer, during the last decade also a chip has been included in the document, developing the so called "electronic passport" (ePassport), which can store biometric data along with the usual holder personal information [2]. To prevent chip data alteration and duplication, solutions based on digital signatures and authentication mechanisms have been introduced, so as to increase further the document trustworthiness. Then, at border controls, the matching

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 123–136, 2013.

between the possible biometric data contained in the chip and the subject carrying the document could be verified, so strengthening the passport-holder link.

Specifically, the electronic component of ePassports is a contactless chip embedded in the document cover. The contactless communication choice was supported by the idea of an easy and fast reading of the digital data and to avoid the common wear and tear effect of contact chips. Anyway, since its introduction, precisely for the contactless nature of the adopted solution, ePassport has raised several disputes. In particular, the possibility for traceability of individuals and their identity information leakage are under indictment [3]. Indeed, without a proper control mechanism, personal data could be for instance silently got through contactless accesses to the chips without the consent of the relative holders, or their movements could be recored interrogating the chips at different places, so in general exposing people to privacy threats and identity theft risks. To contrast such threats some solutions have been introduced, among them an access control mechanism for regulating the chip content reading based on symmetric cryptography called Basic Access Control. Unfortunately some issues have arisen for such mechanism, in particular concerning the strength of the adopted keys, which have made the system weaker in some circumstances.

In this work we highlight further weaknesses of the adopted access control mechanism, in particular in consequence of bad implementations of the software hosted on the chip. Specifically, through a fingerprinting of the different software versions present on board over time, we are able to reduce the space of the keys used in the Basic Access Control. In addition, we have found out an implementation flaw that harms such access control mechanism enabling Man in the Middle attacks. With this work we intend to point out some possible ePassport shortcomings that could be seen by the issuing countries as a cue for checking, and in case refining, their ePassport implementations, as such document is mandatory for citizens and the cost for a bad implementation is represented by a privacy risk for them.

The paper is organized as follows. Section 2 is for presenting the ePassport architecture. In Section 3 and 4 we point out some shortcomings we have found in ePassport implementations and how they weaken the adopted access control mechanism. In Section 5 we comment our results, while Section 6 is for conclusions.

2 ePassport Architecture

The electronic component of an ePassport is represented by a contactless smart card, which is basically composed of a chip connected to a coil. The chip is essentially a mini-computer featured by a microprocessor, peripherals (e.g., coprocessors, random number generators), volatile/non-volatile memories and an I/O interface for the communication with a smart card reader. Such readers rely on a RF field and by induction through the coil power the chip and communicate with it.

Smart card chips are typically managed through an operating system and data are organized according to a file system, which reflects a tree architecture with

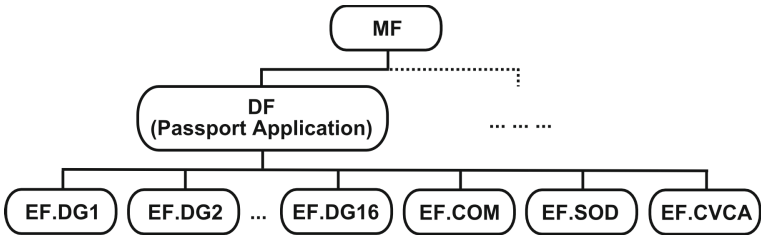


Fig. 1. ePassport Logical Data Structure

directories and files, respectively called Dedicated Files (DFs) and Elementary Files (EFs) [4]. Different applications can be installed on the chip: each of them is featured by an Application Identifier (AID) and is related to a specific portion of the file system. In particular the root directory is called Master File (MF) and each application is identified by a DF, logically placed below MF, that contains the files for that specific application. Each file, including directory files, has a 2-byte File Identifier (FID) which can be used to select that specific file. DFs also have an additional identifier called DF Name, which is at most 16 byte long. In particular, for those DFs representing an application, the relative DF Name is set equal to the corresponding application AID.

The ePassport case [2] is depicted in Fig. 1. Currently only the passport application is usually installed on the chip, but others may be added in the future by the issuer entities for different purposes. The standardized Logical Data Structure (LDS) for ePassports contemplates the presence of different Data Group files (EF.DGn), many of them are optional, that store the holder data and cryptographic material used in some protocols, the EF.SOD file containing the LDS contents hash representations used in a chip authenticity verification process, the EF.COM file carrying some application information and listing the actually present Data Groups, and the EF.CVCA file, presents only if the chip adopts a particular authentication procedure specific for certain Data Groups [5].

The interaction between a reader and a smart card is carried out through messages called Application Protocol Data Unit (APDU) [6]. The communication takes place in a master-slave mode: the reader sends APDU commands and the chip replies with APDU responses. Their format is presented below

Command:	CLA	INS	P1	P2	Lc	Data	Le

Response:	Data	SW1	SW2

where CLA are class bytes, INS denotes the instruction to process, P1 and P2 are command parameters, Lc is the length of the sent Data, Le represents the expected length of Data in the response, and SW1-SW2 are status bytes returning information regarding the launched command. A passport application has to be able to deal with at least SELECT and READ BINARY commands received from a reader, respectively used to select files and read their content.

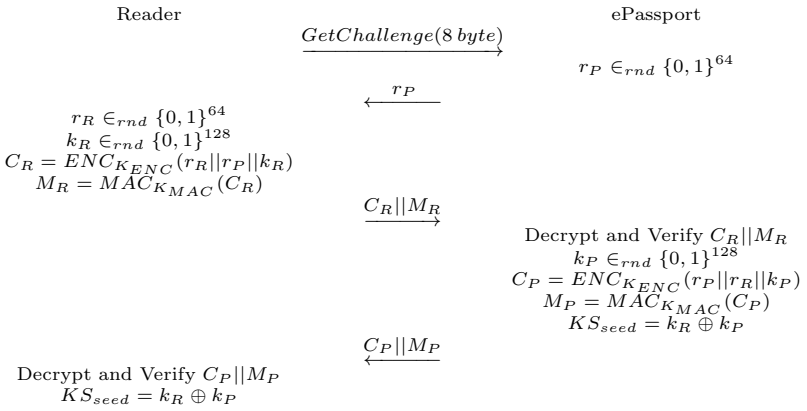


Fig. 2. Basic Access Control protocol. ENC is a Triple-DES in CBC mode with zero IV, while MAC is a cryptographic checksum [2]

Due to the contactless nature of the solution adopted for ePassports, unauthorized subjects may stealthily attempt an access to the chip exploiting the before-mentioned command pair to read the holder information, that is a data skimming could take place. In addition, also over-the-air eavesdropping could be possible, reading the commands-responses transmitted during a reader-passport interaction. As this pose a privacy threat for all those citizens bearing an ePassport, a mechanism to regulate the chip access and encrypt the communication, called Basic Access Control (BAC), has been introduced.

BAC is a cryptographic protocol that allows to authenticate a reader towards an ePassport and establish a common secret between the two for deriving communication session keys. At the base of the protocol there is a pair of secret cryptographic keys, K_{ENC} and K_{MAC}, stored in the chip. For a successful protocol execution, also the reader has to be aware of such keys. In particular, they are derived from a set of data present on the document: Passport Number (PN), holder Date of Birth (DB) and document Date of Expire (DE). Such data are usually printed on an internal page of the passport in the form of an optically readable stripe called Machine Readable Zone (MRZ). Thus, only when the holder physically provides the document, for instance to a control officer, the MRZ can be read, the relative keys computed and data in the ePassport accessed. To this end, ePassport inspection systems are usually equipped with a smart card reader as well as with a scanner to acquire the MRZ, on which the secret keys are derived and then used to engage the BAC protocol. The secret key derivation follows a deterministic scheme: PN||DB||DE (and their check digits, see [1] for the relative computations) is processed through SHA-1 keeping only the most significant 16 bytes (K_{seed}), then K_{ENC} and K_{MAC} are respectively derived as the most significant 16 bytes of SHA-1(K_{seed}||00000001) and SHA-1(K_{seed}||00000002).

Table 1. Basic Italian ePassport BAC keys entropy along with BAC keys entropy of the collected documents exploiting their revealed PN-DE linearity

PN and DE knowledge	DB guessing range		
	50 years	10 years	2 years
Issuing in working days	49.76	47.43	45.11
ePassports since 26/10/06	48.83	46.50	44.18
New booklet since 20/05/10	46.79	44.46	42.14
PN-DE linearity	44.71	42.38	40.06

The BAC protocol carried out between reader and ePassport is shown in Fig. 2. The two parties exchange some random material on which at the end a common secret (KS_{seed}) is derived. K_{ENC} and K_{MAC} are respectively used to encrypt the sent data, relying on a Triple-DES in CBC mode with zero IV, and compute a relative MAC, see [2] for details. If wrong keys are used the procedure is aborted and access denied. From the agreed KS_{seed} a pair of session keys are obtained to secure the channel, encrypting messages and computing the relative MACs [2]. In order that a BAC procedure can take place, other two commands have to be manageable by the enforced passport application on the chip: GET CHALLENGE is used at the beginning to ask for random material from the chip, while with the MUTUAL AUTHENTICATE command $C_R||M_R$ is send to the chip and $C_P||M_P$ is returned in the response.

3 Shrinking the BAC Keys Space

As explained, BAC secret keys, on which the protocol strength is based, are directly derived from PN||DB||DE. As stated by ICAO, the entropy of such a combination is at most 56 bits for a 10-year valid ePassport [2]. This makes BAC natively weak, considering the common minimum acceptable entropy of 80 bits for keys used in a symmetric cryptosystem like the one adopted in ePassports. In addition, as reported in the same ICAO document, entropy can be in concrete even lower, considering that the age of the bearer could be guessed and analysis of the Passport Number may reveal particular structures and/or relations with the Date of Expire according to the document issuing scheme. Studies on Belgian [7], Dutch [8], German [9] and United States [3] passports have confirmed that specific PN structures, e.g., based on geographical data, and PN-DE relations, e.g., a linearity between the two, may exist and effectively reduce the BAC keys entropy.

With this work we highlight that it is possible to shrink further the BAC keys entropy apart from the already known and abovementioned shortcomings. In particular, given an ePassport, it could be possible to extract some "side information" regarding its implementation that could be posed in relation with PN and DE.

For practical reasons we focused on Italian ePassports. Their PN is featured by 2 characters, equal to AA or YA, followed by 7 digits, so providing $log_2(2 \cdot 10^7) =$

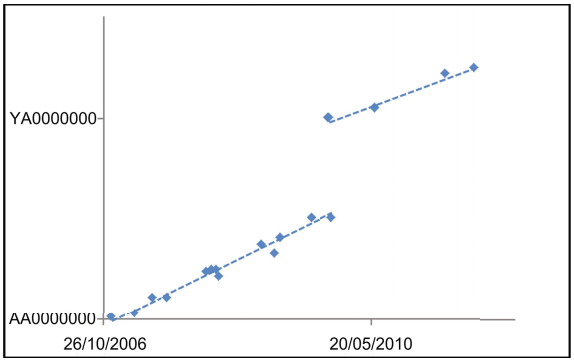


Fig. 3. Date of Issue-PN linearity in the ePassport collection

24.25 bits of entropy. Such documents have a 10-year validity and are issued during working days, thus the DE entropy is $log_2(10 \cdot 365.25 \cdot 5/7) = 11.35$ bits. The actual entropy is shown in the first row of Table 1 for different guesses on the holder age. It is reasonable to assume that the Date of Birth can be almost always correctly guessed in a 50-year range, but in most cases a 10-year range should be actually assumable, while in case some personal information is grabbed, e.g., eavesdropping a call staying in queue or in train, the guessing may be further refined (in our examples we suppose a 2-year range).

So, according to the reported data, without any particular analysis, the entropy is already quite low, below 50 bits. In addition, some further available public information can be exploited. It is known that the Italian ePassport issuing started on 26/10/06 [10], as consequence, considering the working days until the end of January 2012, the time we are writing this work, the DE entropy contribution becomes 10.42 bits. Along with such date, also a PN-DE relation is available: passport booklet has been renewed and the new ones have been issued since 20/05/10 with the passport numbering starting from YA0370001 [11]. Being conservative and assuming a passport numbering in the range (AA00...0:AA99...9)+ (YA00...0:YA0370000) for the first period (first booklet version), and a range (YA0370001:YA99...9) for the second period (second booklet version), the combined PN||DE entropy contribution results equal to 32.63 bits. Thus, considering both the additional information, the BAC keys entropy is 45 bits on average without having performed any specific analysis.

3.1 Data Collection

After the basic observations on document numbering and issuing dates we moved to the analysis of possible PN-DE relations. In particular we were able to collect 22 Italian ePassports, whose relative data revealed that a sort of linearity between PN and DE exists as shown in Fig. 3 (considering the constant 10-year offset between Date of Issue and DE). This should be attributed to the adopted

issuing scheme, namely PN is a sort of serial number roughly increased over time, a problem common also to ePassports of other countries [7] [8].

We grouped our collected PN-Date of Issue pairs according to their relative numbering format, that is one group of PN-Date of Issue pairs where the relative PN started with AA and another one for those pairs featured by a YA in PN. For the two data group we computed the relative linear regressions that are shown in Fig. 3, with the maximum resultant residual that was $\delta \cong 572000$. According to such analysis based on our collected data, given an issue date, the relative Passport Number can be estimated in a $\pm\delta$ interval, equivalent to $log_2(2 \cdot 572000) = 20.13$ bits of entropy. Remembering that the Italian ePassport issuing started at the end of 2006, so limiting the DE entropy contribution to 10.42 bits, the resulting BAC keys entropy is the one presented in the last row of Table 1. Note that to be conservative we took the maximum residual of the two computed linear regressions, anyway a more precise δ per numbering format would be possible for reducing further the key space. On the other hand we have to point that the two numbering formats were slightly overlapped over time, but we neglected this aspect for its minimal impact in the entropy computation. In case a larger passport collection shows a wider overlap, this would turn out in an increased entropy but specific to the overlapped numbering region, leaving for instance unaffected the entropy level of the new released passports.

3.2 ePassport Version Fingerprinting

ePassports may differ each other for the specific hardware and/or software used to implement them. For instance the authors of [12] exploited the responses of the passport application to some specific commands to determine the relative nationality: the idea was that ePassports from different countries present different implementations providing different responses for a given command. In a similar way, we have suspected that, given a country, the relative issued ePassports can present implementation differences over time. This could be for instance the case of an added Data Group or the development of new functionalities. If such differences exist and can be detected, they could be used to identify different ePassport "versions", which could be potentially put in relation with PN and DE.

With this idea in mind we launched on each ePassport in our collection a set of 250 commands and command combinations, varying in several ways the relative fields and analysing the relative responses (we used RFIDIOt as base for our software development [13]). We encountered a broad set of differences that allowed to identify 5 different ePassport versions in our set. We point out that many differences were overlapped, in the sense that they occurred for the same version transition, so for space reason we report here a selected and minimum subset of differences, which are enough for discerning the 5 different versions. The resulting ePassport fingerprinting is shown in Fig. 4 where

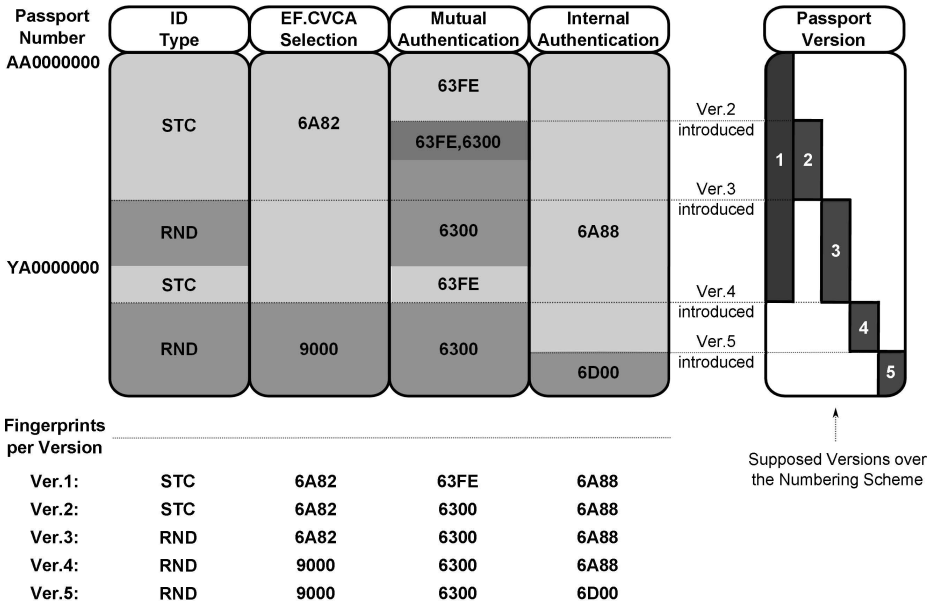


Fig. 4. ePassport versions revealed on the collected documents according to their fingerprints (chip ID type and returned status bytes SW1-SW2 for some specific commands). An intermission appears along the passport numbering for the Ver.1 features, that we ascribed to the lack of some document samples in our set, so for this reason a unified Ver.1 interval was considered. Similarly the Ver.3 interval was considered expanded up to the beginning of the Ver.4 interval.

- *ID Type* represents the returned ID type by the chip at the beginning of the communication;
- *EF.CVCA Selection* denotes the SELECT command 00|A4|02|00|02|011C|00 for the EF.CVCA file selection specifying the relative FID in the Data field, after a chip reset and ePassport application selection;
- *Mutual Authentication* represents a wrong BAC session, so a GET CHALLENGE command 00|84|00|00|08 that is followed by a MUTUAL AUTHENTICATE 00|82|00|00|28|$\{00\}^{40}$|28 featured by a wrong returned value in the Data field, after a chip reset and ePassport application selection;
- *Internal Authentication* denotes an INTERNAL AUTHENTICATE command 00|88|00|00|08|$\{00\}^{8}$|00 launched after a chip reset;

with all the command fields expressed as hexadecimal values and the application selection achieved through the 00|A4|04|00|07|A0000002471001|00 SELECT command containing the passport AID in the Data field. For all the three command sequences presented above, the status bytes SW1-SW2 returned by the last launched instruction in the sequence are exploited to discriminate the passport versions and the relative values are shown in the figure. With regard to the ID type, it has to be pointed out that the chip returns an identification

Table 2. Collected ePassport BAC keys entropy with version fingerprinting

ePassport version	DB guessing range					
				(using PN-DE linearity)		
	50 years	10 years	2 years	50 years	10 years	2 years
Ver.1	46.51	44.18	41.86	44.15	41.82	39.50
Ver.2	44.12	41.79	39.47	42.73	40.40	38.08
Ver.3	42.57	40.24	37.92	42.31	39.98	37.66
Ver.4	41.52	39.19	36.87	(41.52)	(39.19)	(36.87)
Ver.5	42.82	40.49	38.17	42.43	40.10	37.78

number at the beginning of the reader-chip interaction. It is advisable to have it random (RND) so that traceability threats for passport holders are prevented, anyway for the first issued documents it was static (STC), providing a further information to discern different ePassport versions.

The resulting ePassport versions in relation with the passport numbering are outlined in Fig. 4, where only the Version 1 appears to be overlapped to others. The reason is that its peculiar fingerprints appear for the first numbering part and then in a second stage. We suspected that such a kind of document could have been issued contemporaneously with other versions for some logistic reason. A larger document collection would refine the analysis, anyway to be conservative we have kept an extended PN range for it. For similar reasons we have kept a longer PN range for the Version 2 and 3. For each version, along with the shown PN range, we have also found a relative Date of Issue – so a DE – range. As consequence each version was featured by a reduced set of PN||DE combinations and the corresponding BAC keys entropy is presented on the left side of Table 2. On the right side of the same table we present the entropy in case that also the data linearity parameter δ computed in Section 3.1 is exploited within each version (for the Version 4 there is no advantage). As evident, fingerprinting affects further the keys entropy. Note that we have chosen not to show the identified PN and DE boundaries relative to the different versions as they were derived on collected documents by volunteers. Furthermore with this work we mainly aim at pointing out the potential of the idea, which is in principle applicable to ePassports issued by any country.

That being so, fingerprinting a document it could be possible to detect its version and take advantage of a reduced key space. As easily imaginable this could improve a skimming attack, first fingerprinting the document and then trying different keys according to the detected PN||DE combination space. Concerning this we have to point out that, from the Version 3 on, an access attempt counter has been set up in the Italian ePassports. After a certain number of attempts the authentication procedure is slowed down until a successful attempt occurs, thereby practically impairing the attack.

The fingerprinting flaw could be also used to enhance eavesdropping attacks. For instance in a queue for border controls, in train or at a hotel reception an ePassport fingerprinting could take place and then, at the time of chip

interrogation, the communication is recorded (it is supposed that distances up to several meters can potentially be reached [14]). The acquired data, the fingerprints, revealing the ePassport keys space, and the encrypted messages, could be sent to a (maybe remote) cracker machine to reveal the holder information. Such a machine could be a COPACOBANA, which is able to test $\approx 2^{28}$ BAC keys per second on a recorded communication [9]. According to Table 2 the worst case is associated to an ePassport whose keys entropy is ≈ 46 bits, so requiring ≈ 3 days for a break. Anyway, ePassport versions presenting a smaller key space exist, for instance in scenarios featured by keys entropy of ≈ 36 bits the break would be achieved almost in real time, requiring only ≈ 4 minutes.

As further remarks, we point out that whenever a new "version" is issued, right after its issuing it immediately becomes the weakest one, as the relative PN and DE ranges result strictly limited. In addition, following further analysis, a more precise PN-DE linearity per ePassport version may turn out.

4 Enabling Man in the Middle Attacks

In order to scrutinize the BAC protocol security we moved to the analysis of the generated random numbers, which are at the base of the process trustworthiness. In particular for several ePassports in our collection we generated some relative random data through the GET CHALLENGE command and then we tested them for randomness through the NIST Statistical Test Suite [15]. For each examined ePassport we collected 55 series of 1MB, which underwent all the 15 tests in the suite with their parameters set to the default values except for the Block Frequency test where M was equal to 20000. Differently from [16], no weaknesses arose in our experiments, so the random number generator appeared to be sound.

In spite of the above-mentioned good results, working on the random number generation a further issue appeared. As explained in Section 2, for a complete BAC session the commands GET CHALLENGE and MUTUAL AUTHENTICATE have to be launched in sequence. If the MUTUAL AUTHENTICATE is run without having before received the random data r_P through the GET CHALLENGE command an error should occur. Furthermore, according to the designed protocol (Fig. 2), the returned r_P has to be 8 bytes in length. Unluckily this last constraint could be bypassed for the ePassports in our collection. Indeed, we found out that if a GET CHALLENGE command is run specifying to return less then 8 bytes of data, the subsequent MUTUAL AUTHENTICATE command is not stopped. As an 8 bytes r_P is necessary in the BAC protocol, we were curious to figure out how the "reduced" r_P was managed during the execution of the MUTUAL AUTHENTICATE instruction. After some trials we discovered that the "reduced" r_P was simply padded with zeros to make it 8 bytes long. This obviously represents an implementation error and it appears to be common to all the ePassport versions we have identified, even for recently issued ePassports present in the collection. It is however important not to generalize this result beyond the limited set of collected ePassports. A wider study would be needed to assess if all the Italian ePassports are actually affected.

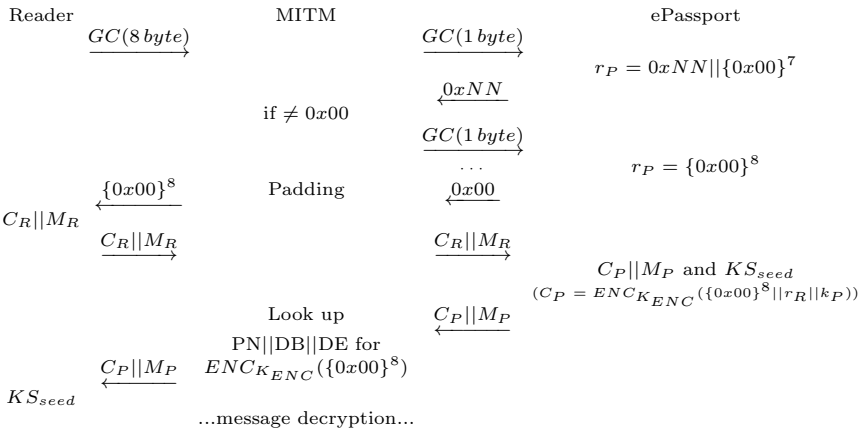


Fig. 5. ePassport Man in the Middle attack due to a bad implementation

Such a flaw opens the way for an attack. Indeed, BAC is designed to also prevent unnoticed alterations of communication between a reader and an ePassport, like for instance a Man in the Middle Attack (MITM). But, unluckily, such an error allows a MITM as presented in Fig. 5. The man in the middle intercepts the reader-chip communication and repeatedly (there is no limit on GET CHALLENGE executions) asks for 1 byte of random data (the minimum size allowed) to the chip until zero ($0x00$) is returned (it required 1 s on average with our set up featured by a SCM SDI010 reader and a 424Kbps bidirectional communication rate). Now the man in the middle knows that the chip internally has r_P=$\{0x00\}^8$, so it simply returns to the reader the same value. From this stage on the man in the middle simply forwards and records the messages between the two. However, when $C_P||M_P$ is returned by the chip, he knows that the first 8 bytes are equal to $ENC_{K_{ENC}}(\{0x00\}^8)$: finding the right K_{ENC} the whole communication is decrypted. Indeed K_{ENC} is simply derived from PN||DB||DE which regulates the overall security, so it is simply necessary to find the right combination giving that key. But as explained PN||DB||DE tends to present a limited entropy, so it would be possible to pre-compute and store $ENC_{K_{ENC}}(\{0x00\}^8)$ for all the possible combinations, then using such data as a look up table. Considering for instance the entropy values presented in Table 2, the pre-computed data would amount to some TBs, which could be stored and made available through a remote NAS. Note that in such a scenario, holder data would be always got in real time.

Although such an attack is hard to apply in practice, we decided to design it in order to stress as perfect implementations are hard to achieve and more attention should be devoted to the application development process, as even minor lacks of attention, like the one just presented, risk opening serious threats.

5 Discussion and Countermeasures

The intrinsic BAC issues linked to the low entropy of its fields are well known in the literature, but in this paper we have highlighted how additional issues arise when some side information can be got from the application implementation. Indeed the ePassport fingerprinting represents an effective way for shrinking the BAC keys space as long as the BAC secret parameter is represented by a MRZ including sequential data. This can be largely ascribed to the implementation specifications [2] that do not cover the overall ePassport application and consequently several details are up to the issuer countries. For instance there is not any recommendation in case of unexpected commands from a reader, so each country implements its own solution that could be changed over time revealing "versions". Another way to inadvertently disclose "time information" is represented by the addition of Data Groups from a certain ePassport generation on, whose presence/absence can be used to denote a timeline. We have verified that it is possible to discern between presence/absence of some Data Groups in ePassports issued by some countries before the BAC procedure takes place. Other similar examples are possible. All this kind of "side information" should be masked from the outside, at least until the BAC protocol is completed. In support of this statement, we have to consider that even in case the BAC strength is improved randomizing the Passport Numbers so that the relative MRZ entropy is increased, as done by Germany, if a fingerprinting remains available it would continue to affect the Date of Expire entropy, as DE is linear over time. For similar reasons, lack of comprehensive specifications/tests, the introduction of the random generation flaw has been possible.

That being so particular attention should be spent to the ePassport implementation and we hope that the findings presented in this work can represent a cue to push the issuing countries into checking, and in case improving, their ePassport implementation process. Indeed we point out as some implementation differences have also been detected in a couple of ePassports from Germany and Belgium, so we remark that our results may concern ePassports from different countries and not only the Italian ones, which have been chosen for simple practical reasons.

More in general, the introduction of refined and more comprehensive specifications and tests would be auspicable, so as to force all ePassports to reply in a uniform way regardless of their country and issuing time until the BAC procedure is completed. To improve further the implementations, the introduction of automatic testing techniques based on models, as suggested in [17], could represent a good way to check the developed passport applications and in case detect undesired and unforeseen hidden behaviours. This could be beneficial not only for BAC but also for the other ePassport mechanisms not mentioned in this paper. In addition, until the MRZ is not re-designed as to provide an acceptable entropy level, an access attempt counter, like the one adopted in the Italian ePassports, could be considered as additional deterrent against skimming attacks. In the meanwhile, citizens could adopt shielding covers to erect an

additional barrier on the access way to their documents, as shields in ePassport covers are not adopted by almost all the issuing countries.

During the last few years a new protocol called PACE [5] has been proposed as access mechanism for the ePassports and it should replace BAC in the future. Thanks to a Diffie-Hellman key agreement it provides a good forward secrecy de facto preventing attacks based on eavesdropping and offline decryption. Anyway for a correct protocol execution reader and chip start from a shared common secret, which can be, among others, again the MRZ. If a fingerprinting remains possible for shrinking the MRZ entropy, in principle a skimming attack could be enhanced, in particular thinking about the case of new issued versions that become the weakest ones right after their introduction. That being so, care about the implementation should be taken also when this protocol will be adopted.

We point out that ePassport nationality detection [12], according to our results, could result a bit more complicated but still possible. Indeed, given a command we have shown that different responses can take place for the documents issued by a country over time, making a bit harder to identify peculiar fingerprints for a nationality that do not change across different ePassport versions.

In the end we point out that our results can be exploited for implementation considerations of another electronic document, the eDriving License [18]. Indeed the access is regulated according to BAP (Basic Access Protection), that could be configured to work exactly as BAC, and where the shared secret of the protocol can be represented by a combination of existing fields in MRZ-style, so risking to suffer for the same problems.

6 Conclusions

In this work we have scrutinized the access control mechanism adopted in ePassports from a different point of view, in particular verifying if some defects in the passport application implementation can affect the BAC security. We have found out that different implementation details allow to discern among different software versions installed on the chip over time, which can be put in relation with MRZ data and so exploited for a more accurate guessing of the BAC keys. As result, attacks to the system would require a reduced amount of time. In addition, we have detected an implementation flaw that affects part of the BAC protocol opening the way for a hypothetical Man in the Middle attack. This work could be interpreted by the issuing countries as a cue for checking their ePassport implementations and fixing possible flaws, as citizen privacy is involved. As future work we will assess the other security mechanisms adopted in ePassports and evaluate how flaws in such devices affect citizen data protection and privacy.

References

1. International Civil Aviation Organization: Machine Readable Travel Documents, Part 1, 6th edn., vol. 1 (2006)
2. International Civil Aviation Organization: Machine Readable Travel Documents, Part 1, 6th edn., vol. 2 (2006)

3. Juels, A., Molnar, D., Wagner, D.: Security and privacy issues in e-passports. In: Proceedings of the IEEE First International Conference on Security and Privacy for Emerging Areas in Communications Networks, pp. 74–88 (2005)
4. Rankl, W., Effing, W.: Smart Card Handbook, 3rd edn. Wiley (2003)
5. BSI: Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE) and Restricted Identification (RI), Ver. 2.05 (2010)
6. ISO/IEC 7816: Identification Cards – Integrated Circuit Cards – Part 4: Organization, Security and Commands for Interchange (2005)
7. Avoine, G., Kalach, K., Quisquater, J.J.: ePassport: Securing International Contacts with Contactless Chips. In: Proceedings of the 12th International Conference on Financial Cryptograpy and Data Security, pp, pp. 141–155 (2008)
8. Hoepman, J.-H., Hubbers, E., Jacobs, B., Oostdijk, M., Schreur, R.W.: Crossing Borders: Security and Privacy Issues of the European e-Passport. In: Yoshiura, H., Sakurai, K., Rannenberg, K., Murayama, Y., Kawamura, S.-I. (eds.) IWSEC 2006. LNCS, vol. 4266, pp. 152–167. Springer, Heidelberg (2006)
9. Liu, Y., Kasper, T., Lemke-Rust, K., Paar, C.: E-Passport: Cracking Basic Access Control Keys. In: Meersman, R., Tari, Z. (eds.) OTM 2007, Part II. LNCS, vol. 4804, pp. 1531–1547. Springer, Heidelberg (2007)
10. Polizia di Stato: Il Passaporto per Entrare negli Stati Uniti d'America (2012), http://poliziadistato.it/articolo/1090/
11. Polizia di Stato: Note Tecniche Nuovo Passaporto (2012), http://img.poliziadistato.it/docs/note_tecniche.pdf
12. Richter, H., Mostowski, W., Poll, E.: Fingerprinting passports. In: NLUUG Spring Conference on Security, pp, pp. 21–30 (2008)
13. Laurie, A.: RFIDIOt (2012), http://rfidiot.org/
14. Carluccio, D., Lemke-Rust, K., Paar, C., Sadeghi, A.-R.: E-Passport: The Global Traceability Or How to Feel Like a UPS Package. In: Lee, J.K., Yi, O., Yung, M. (eds.) WISA 2006. LNCS, vol. 4298, pp. 391–404. Springer, Heidelberg (2007)
15. NIST: Random Number Generation (2012), http://csrc.nist.gov/groups/ST/toolkit/rng/index.html
16. Auletta, V., Blundo, C., De Caro, A., De Cristofaro, E., Persiano, G., Visconti, I.: Increasing Privacy Threats in the Cyberspace: the Case of Italian e-Passports. In: Proceedings of the 14th International Conference on Financial Cryptograpy and Data Security, pp. 94–104 (2010)
17. Mostowski, W., Poll, E., Schmaltz, J., Tretmans, J., Wichers Schreur, R.: Model-Based Testing of Electronic Passports. In: Alpuente, M., Cook, B., Joubert, C. (eds.) FMICS 2009. LNCS, vol. 5825, pp. 207–209. Springer, Heidelberg (2009)
18. ISO/IEC: Information Technology – Personal Identification – ISO-Compliant Driving Licence – Part 3: Access Control, Authentication and Integrity Validation (2009)

Never Trust a Bunny*

Daniel J. Bernstein[1] and Tanja Lange[2]

[1] Department of Computer Science
University of Illinois at Chicago, Chicago, IL 60607–7053, USA
djb@cr.yp.to
[2] Department of Mathematics and Computer Science
Technische Universiteit Eindhoven, P.O. Box 513, 5600 MB Eindhoven,
The Netherlands
tanja@hyperelliptic.org

Abstract. "Lapin" is a new RFID authentication protocol proposed at FSE 2012. "Ring-LPN" (Ring-Learning-Parity-with-Noise) is a new computational problem proposed in the same paper; there is a proof relating the security of Lapin to the difficulty of Ring-LPN. This paper presents an attack against Ring-LPN-512 and Lapin-512. The attack is not practical but nevertheless violates specific security claims in the FSE 2012 paper.

Keywords: Authentication, RFID, LPN, Ring-LPN, attacks, Lapin, bunnies.

1 Introduction

Lapin [15] is a lightweight "RFID authentication" system introduced at FSE 2012 a week before the submission of this paper. The system is claimed in [15, Section 1.1] to be "provably secure against active attacks". This claim is qualified elsewhere in [15]: attacking Lapin-n is provably as difficult as attacking an n-bit "Ring-LPN" problem introduced in the same paper [15].

This proof begs the question of how secure Ring-LPN-n is. Ring-LPN-512 is claimed in [15, Section 3.1] to "require 2^{77} memory (and thus at least that much time) to solve when given access to approximately as many samples" according to the "analysis" of [22].

A closer look shows that the attack in [22] begins with nearly 2^{76} 513-bit vectors consuming nearly 2^{82} bytes of memory, performs a series of 6 sorting steps to compute 2^{63} reduced 75-bit vectors, and concludes with a similarly expensive Walsh transform. Note that [22] is an analysis of *some* attacks; there is no proof that better attacks do not exist against Ring-LPN-512, or against Lapin-512.

* The Wolf: "I knew it! Never trust a bunny!" Twitchy: "Never trust a bunny!" [9].

** This work was supported by the National Science Foundation under grant 1018836, and by the European Commission under Contract ICT-2007-216676 ECRYPT II. Permanent ID of this document: 78fc70fd7aa0cbd24ffe73a426567577. Date: 2012.06.21.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 137–148, 2013.

Contributions of This Paper. We present an attack that discovers the Ring-LPN-512 secret using $<2^{56}$ bytes of memory, $<2^{38}$ queries, and $<2^{98}$ easily vectorized bit operations. Running the attack twice discovers the Lapin-512 secret, allowing the attacker to clone a Lapin-512 RFID tag.

We do not claim that this Ring-LPN-512 attack is feasible. However, it is obviously much closer to feasibility than the attack in [22], and in particular breaks solidly through the "2^{77} memory" barrier claimed in [15] while remaining competitive in bit operations. It breaks even more solidly through the "approximately $[2^{77}]$ samples" barrier claimed in [15].

The concrete proposal in [15] is actually Lapin-532 rather than Lapin-512. Our attack scales naturally to any Ring-LPN-n, and allows many different tradeoffs between memory, queries, and bit operations.

Varying the Error Fraction. LPN, Ring-LPN, and Lapin have an implicit "error fraction" τ. The specific Lapin-n protocols and Ring-LPN-n problems discussed above have $\tau = 1/8$, but smaller values and larger values have appeared in other proposals: for example, the "reducible" Lapin variant in [15, Section 5.1] has $\tau = 1/6$, while [22, Section 5.2] recommends $\tau = 1/20$. We state and analyze our attack with τ as a variable.

It is stated in [19] that "the best algorithms [attacking LPN] take time $2^{\ell/\log \ell}$" for any "constant" $\tau > 0$. In Section 4 we consider the case $n = 1024$, $\tau = 1/20$, and give an attack on Ring-LPN taking far fewer than $2^{1024/\lg 1024} = 2^{102.4}$ bit operations, while using $<2^{21}$ bytes of memory and $<2^{64}$ queries. A variant of the attack uses more bit operations, but still fewer than $2^{102.4}$ bit operations, $<2^{21}$ bytes of memory, and just 10 Ring-LPN queries. This variant also works for LPN, using just 5120 queries.

We emphasize that τ plays an important role in the cost of these attacks, and that security estimates must take τ into account. Note that increasing τ slows down protocols.

Previous Work. HB [16], HB$^+$ [18], HB^{++} [6], HB-MP [23], HB* [8], Trusted-HB [5], and HB$^\sharp$ [13] were each broken within one year of being proposed. See, e.g., [12].

Each of these RFID authentication protocols claimed security on the basis of the alleged difficulty of various LPN problems. The protocols were broken in two different ways. First, in many cases the protocol structure allowed attacks that were simpler than breaking those LPN problems. Second, in many cases the LPN problems turned out to be weaker than claimed.

Our attack follows the second line of work. We combine and refine LPN attack ideas previously published by Blum, Kalai, Wasserman, Levieil, Fouque, and Kirchner. See [3], [22], and [20].

We note that some of the previous attacks have the advantage of provability. Our algorithm — like the number-field sieve for attacking RSA, the rho method for attacking ECC, etc. — is only analyzed heuristically.

Notes on Low-Memory LPN Attacks. It is often claimed that memory is the main bottleneck in breaking LPN: consider the "require 2^{77} memory" quote

above, or the quote "It takes 2^{46} bytes of memory to solve a LPN problem with $k = 256$ [and $\tau = 1/8$]" from [22, Section 5.2].

This claim is obviously violated by *extremely slow* attacks: a glance at the definition of LPN shows that a brute-force search for the LPN secret takes negligible memory. What is less obvious is that the claim is also violated by *fast* attacks, although perhaps not quite the fastest possible attacks. This was already visible from Kirchner's paper [20] (and not from earlier papers such as [22]); we introduce an improvement in the Walsh-transform step, but the critical LPN attack idea is due to Kirchner.

Notes on LPN vs. Ring LPN. [15, Section 3.1] says that the most efficient known attacks against "irreducible" cases of (q-query) Ring-LPN are simply attacks against (nq-query) LPN. However, one of the major steps in our Ring-LPN attack saves some time by taking advantage of the ring structure of Ring-LPN, suggesting that Ring-LPN is not as hard as LPN. This savings applies to both the "reducible" and "irreducible" cases; it is lost only when the number of queries is very small.

Consequences for RFID Security. We do not mean to suggest that LPN and Ring-LPN are incapable of reaching a 2^{80} security level. LPN has always been able to dodge attacks by increasing its parameters, the same way that RSA has always been able to dodge improved factorization algorithms. However, we are skeptical of the idea that LPN, Ring-LPN, HB, Lapin, etc. can provide the same security as a block cipher within the same RFID cost constraints.

Lapin-532 is advertised in [15, Keywords] as being suitable for "RFID authentication" and in [15, Abstract] as having "10 times smaller code size" than AES. Code size is claimed in [15, Section 5] to be "the most precious resource once the run-time constraints are fulfilled ... For instance, the WISP, a computational RFID tag, has only 8 kBytes of program memory."

This comparison is puzzling for several reasons. First, the AES implementation cited in [15] is clearly very far from optimal. Second, Lapin-532 was designed for 2^{80} security, while AES was designed for 2^{128} security. Third, AES is certainly not the state of the art in lightweight block-cipher design. Fourth, Lapin-532 obviously needs much more RAM than a block cipher, and bytes of RAM are inherently more expensive than bytes of ROM; note that the MSP430F2132 in the current WISP4.1 has only 512 bytes of RAM.

What [15] actually reports for Lapin-532 is 195000 cycles using 459 bytes of code on an AVR ATmega163 smart-card CPU. For comparison, [10] reports

- 128-bit AES: 4600 AVR cycles, 1659 bytes of code, 32 bytes of RAM;
- 128-bit NOEKEON: 23500 AVR cycles, 364 bytes of code, 32 bytes of RAM;
- 128-bit HIGHT: 19500 AVR cycles, 402 bytes of code, 32 bytes of RAM;

and so on. Lapin uses more code than (e.g.) NOEKEON, uses more RAM, uses more cycles, uses more communication, and provides much less security against known attacks. The cost evaluation in [15, Section 5.3] also omits the cost of generating random bits, stating that this cost is "independent of the underlying cryptographic functions"; but cipher-based protocols actually require far fewer

random bits than Lapin. There is also no evidence that switching platforms (from AVR to MSP430 or to an ASIC) would make Lapin competitive.

Notes on Provable Security. The literature describes several ways for an RFID tag to securely authenticate itself using a secret AES key shared with the RFID reader; see, e.g., [11]. Attacking any of these AES-based authentication protocols is provably as difficult as attacking the standard "PRP" security notion for AES: distinguishing AES (with a secret key) from a uniform random permutation. Of course, AES can also be replaced with other block ciphers, as discussed above.

We see three reasons that the security proofs for these AES-based authentication systems are more satisfactory than the security proof for Lapin. First, the Lapin proof is limited to "active" attacks (obtaining information from the tag and then attempting to fool a reader) while the AES proofs cover a wider class of "man-in-the-middle" attacks (interacting with the tag and the reader in parallel); the "man-in-the-middle" attack in [15, Appendix] suggests that Lapin is actually less secure than Ring-LPN. Second, the AES proofs are based on a problem that seems difficult after extensive cryptanalysis, namely the AES PRP-distinguishing problem, while the Lapin proof is based on the obviously much less mature Ring-LPN problem. Third, even if there is no further progress in attacks, the quantitative security-performance tradeoff will be considerably worse for Ring-LPN than for AES.

We are puzzled by the claims in [15] that Lapin is "provably secure" while AES-based protocols are " 'merely' computationally secure with respect to known attacks". There are certainly reasons to avoid describing the AES-based protocols as "provably secure", but all of the same reasons also apply to Lapin. The AES proofs begin with an unproven hypothesis, namely that the AES PRP-distinguishing problem is difficult, but the Lapin proofs also begin with an unproven hypothesis, namely that the Ring-LPN problem is difficult. There could be better attacks against the AES PRP-distinguishing problem and the AES-based protocols, but there also could be better attacks against the Ring-LPN problem and the Lapin protocol. We are unaware of any reason to prefer Ring-LPN over AES as a foundation for security.

Acknowledgments. Thanks to the anonymous referees for their comments. We acknowledge one referee in particular for the random-bits comment above.

2 LPN, Ring-LPN, and Lapin

The learning parity with noise (LPN) problem was introduced as a basis of cryptographic authentication schemes by Hopper and Blum in [16]. Here is the basic protocol they describe: The parameters are $m, n \in \mathbf{Z}$ and $\tau, \tau' \in \mathbf{Q}$ with $0 < \tau \leq \tau' < 1/2$. The authentication tag and the reader share an n-bit secret $s \in \mathbf{F}_2^n$. To authenticate the tag to the reader the reader sends an n-bit challenge c to the tag; the tag computes the dot product $y = c \cdot s$, puts $e = 0$ with probability $1 - \tau$ and $e = 1$ with probability τ, and responds with $t = y + e$. The reader computes $e = t + c \cdot s$. The tag is accepted if after m repetitions of

the basic protocol the number of repetitions with $e \neq 0$ is no larger than $\tau' m$. Obviously one can merge the m executions into a single step by changing the challenge c to an $m \times n$ matrix C and requesting the tag to compute $Cs + e$ for some error vector $e \in \mathbf{F}_2^m$ in which each bit of e is set with probability τ.

LPN refers to learning the secret s under the noise e; the distinguishing version is given a sequence of pairs (c, b) and should distinguish whether it comes from $(c, c \cdot s + e)$ or (c, b) for random b. When HB was proposed, Håstad had shown in [14] that LPN is NP-hard and the best known attack at that time, due to Blum, Kalai and Wasserman [2], took $2^{\Omega(n/\log n)}$ challenge-response pairs and time $2^{\Omega(n/\log n)}$. However, in the above protocol an active attacker impersonating the reader can determine s bit-by-bit by setting c to be the ith unit vector several times until bit i of s is known with sufficiently high probability. Similarly, many subsequent HB variants have been broken because of the way that they use LPN, rather than because LPN is much easier than thought.

The past 10 years have seen various changes to the HB protocol which differ in the number of passes and in how the tags introduce extra randomness to avoid the basic attack; also some progress on solving the computational LPN problem has been made. At FSE 2012, Heyse et al. [15] presented a 2-pass authentication protocol based on the Eurocrypt 2011 2-pass protocol by Kiltz et al. [19] and some modifications for more efficient implementation. We first describe a matrix variant of their protocol and then state their changes to reduce communication and computation complexity.

The tag and the reader share 2 secret vectors $s, s' \in \mathbf{F}_2^n$. For each run of the protocol, the reader generates an $n \times n$ challenge matrix C, and the tag generates a random $n \times n$ matrix R. Challenged with C, the tag replies with $(R, R(Cs + s') + e)$. The reader deduces e and accepts the tag as authentic if R is invertible and the Hamming weight of e is smaller than $n\tau'$.

The scheme described so far is based on the LPN problem. The authors of [15] introduce a new problem called the "Ring-LPN" problem. Take the ring $\mathbf{F}_2[x]/f$ for some polynomial f of degree n and embed n-bit vectors $(s_0, s_1, \ldots, s_{n-1})$ as ring elements $\sum s_i x^i$. The Ring-LPN problem is to reconstruct s given pairs $(r, rs + e)$, where the computations take place modulo f and the bits of e are set with probability τ. In the decision version the distribution of $(r, rs + e)$ should be distinguished from (r, y) for random y's in $\mathbf{F}_2[x]/f$.

Using a ring has the advantage that the product of two vectors, interpreted as a ring element, gives another vector. Hence the $n \times n$ matrices C and R can be replaced by ring elements c and r, reducing the communication cost from $2n^2 + n$ to $3n$ and reducing the need for random bits. Furthermore, the ring operations can be implemented more efficiently. To reduce the communication complexity further, c is derived from a shorter bit string by some deterministic function $\pi : \{0,1\}^\lambda \to \mathbf{F}_2[x]/f$. In the Lapin-$n$ scheme the answers from the tag have the form $(r, z) = (r, r(s\pi(c) + s') + e)$. The reader verifies that r is invertible and that $z + r(s\pi(c) + s')$ has Hamming weight less than $n\tau'$.

To relate the Ring-LPN version to the LPN version note that R can be obtained from r as the matrix which for $0 \leq i \leq n-1$ has the coefficients of

$rx^i \bmod f$ in column i, starting with the coefficient of x^0. This way the embedding of Rs into the ring gives $rs \bmod f$.

For the attacks we note that the reader controls c and thereby $\pi(c)$. If an attacker can solve the Ring-LPN problem, i.e. find s from observing pairs $(r_j, r_j s + e_j)$, then he can break the Lapin scheme by running the attack twice, once with c_1 and once with c_2 (for which $(\pi(c_1) + \pi(c_2))$ is invertible): The first attack reveals $s_1 = s\pi(c_1) + s'$ and the second one $s_2 = s\pi(c_2) + s'$ which give $s = (s_1 + s_2)/(\pi(c_1) + \pi(c_2))$ and $s' = (s_1\pi(c_2) + s_2\pi(c_1))/(\pi(c_1) + \pi(c_2))$.

The authors of [15] present 2 versions of Lapin-n which differ in whether or not f is irreducible; the attacks presented in the following section are independent of the properties of f, so we skip the review of these considerations.

3 The Attack

This section states an attack against Ring-LPN. The attack has several parameters introduced in this section and optimized in the next section.

Initial Queries. Query the Ring-LPN oracle repeatedly, obtaining a sequence $(r_1, r_1 s + e_1), (r_2, r_2 s + e_2), \ldots, (r_q, r_q s + e_q)$. Here q, the number of queries, is an attack parameter. As in the previous section, s is the oracle's secret; each r_j is a uniform random ring element; each bit of each e_j is set with probability τ; and all of the random choices made here are independent. Define $\delta = 1 - 2\tau$.

Targeting e_i. Choose $i \in \{1, \ldots, q\}$. The remaining steps of the attack hope that r_i is an invertible ring element (which is overwhelmingly likely) and that e_i does not have many bits set (which has a noticeable chance, depending on various parameters).

Compute $1/r_i$ in the ring; if this fails, stop. For each $j \neq i$, compute r_j/r_i and then $(r_j/r_i)(r_i s + e_i) + (r_j s + e_j) = (r_j/r_i)e_i + e_j$.

This computation transforms the original Ring-LPN oracle for s into a Ring-LPN oracle for e_i: the sequence of pairs $(r_j/r_i, (r_j/r_i)e_i + e_j)$ has exactly the same distribution as a sequence of Ring-LPN outputs for e_i. This transformation is useful because searching through the likely possibilities for e_i is much faster than searching through the possibilities for s, especially when τ is small.

The idea of this transformation was introduced by Kirchner in [20, Section 4.3.2] in the context of LPN. Starting from LPN outputs $(r_1, r_1 \cdot s + e_1), \ldots, (r_q, r_q \cdot s + e_q)$, Kirchner selects a target set T of n indices i, hoping that the n rows r_i for $i \in T$ form an invertible $n \times n$ matrix (probability about 30%) and that the n noise bits e_i for $i \in T$ do not have many bits set. Kirchner then applies the inverse of the matrix to replace the LPN oracle for s with an LPN oracle for these n bits e_i.

Our Ring-LPN variant is simpler and, more importantly, faster: ring multiplication is faster than matrix multiplication. We comment, however, that for small q the LPN transformation has an interesting feature not pointed out in [20]: it provides tremendous flexibility in the choice of n rows to combine into an invertible matrix. Exploiting the Ring-LPN structure means that our only

potential targets are $e_1, e_2, \ldots, e_q$, whereas the analogous LPN transformation is free to target any set of n positions. See Section 4 for an example.

Forgetting the Ring Structure. For each $j \neq i$, and for each $k \in \{0, \ldots, n-1\}$, define $v_{j,k} \in \mathbf{F}_2^{\{0,1,\ldots,n-1\}}$ as the vector whose ℓth bit is the coefficient of x^k in the ring element $(r_j/r_i)x^\ell$. Then the coefficient of x^k in the known quantity $(r_j/r_i)e_i + e_j$ is exactly $v_{j,k} \cdot e_i + e_{j,k}$, where $e_{j,k}$ means the coefficient of x^k in e_j. We now have $(q-1)n$ vectors $v_{j,k}$ and noisy dot products $v_{j,k} \cdot e_i + e_{j,k}$.

Clearing Bits. Build a table of 2^b vectors as follows, where b is another algorithm parameter. For each of the above $(q-1)n$ vectors $v_{j,k}$ in turn, use the first b bits of $v_{j,k}$ as a table index, and store $v_{j,k}$, together with its noisy dot product, at the corresponding table position. If there is already a vector at that position in the table, do not overwrite that vector; instead xor it into $v_{j,k}$, and xor its noisy dot product into $v_{j,k} \cdot e_i + e_{j,k}$.

Now discard all the vectors in the table. There are at least $(q-1)n - 2^b$ vectors *not* in the table, and each of them now has its first b bits all set to 0. Each noisy dot product has become more noisy: if two bits $e_{j,k}$ are each set independently with probability $(1-\delta)/2$ then their xor is set with probability $(1-\delta^2)/2$.

The table requires $2^b(n-b+2)$ bits of storage: each entry has n bits, minus b bits implicit from the table position, plus 1 bit for a noisy dot product, plus 1 bit to say whether the table entry is used. The list of vectors *not* in the table overwrites the original list of vectors without consuming any extra space.

We comment that, unless $(q-1)n$ is much larger than 2^b, one expects some table entries to be unused, so the $(q-1)n - 2^b$ bound is somewhat pessimistic. We also repeat a comment from Levieil and Fouque [22, Section 4]: starting from $t \approx (q-1)n/2^b$ different vectors sharing their first b bits, one can build $t(t-1)/2$ differences having their first b bits clear, rather than just $t-1$ differences with a single vector; this expands the pool of vectors beyond the original number of samples. For simplicity we avoid this option.

Clearing More Bits. Repeat the above table-building process a total of a times, where a is another algorithm parameter. This produces a sequence of at least $(q-1)n - 2^b a$ vectors v, each having its first ab bits clear and each accompanied by a noisy dot product $v \cdot e_i + \cdots$, where the noise is set with probability $(1-\delta^{2^a})/2$.

(In the extreme case $a = 0$, these vectors are the original $(q-1)n$ vectors with noise probability $(1-\delta)/2$. No storage is required for the table in this case, and the vectors can be compressed to the ring elements r_j/r_i.)

This bit-clearing procedure, without the initial transformation from s to e_i, was introduced by Blum, Kalai, and Wasserman in [3]. Blum, Kalai, and Wasserman cleared $n-1$ bits, obtaining noisy dot products $(0, \ldots, 0, 0) \cdot s + \cdots$ and $(0, \ldots, 0, 1) \cdot s + \cdots$, and then computed the last bit of s by a majority vote of the $(0, \ldots, 0, 1) \cdot s + \cdots$ dot products.

Guessing the Remaining Bits of e_i. The algorithm hopes at this point that e_i has Hamming weight $\leq W$ in its final $n - ab - \ell$ positions, where W and ℓ are two more algorithm parameters.

The first ab bits of e_i are not relevant to the noisy dot products $v \cdot e_i$ for the vectors v constructed above. We thus consider possible patterns for the remaining bits of e_i, i.e., patterns of n bits that have Hamming weight 0 in their first ab positions and Hamming weight $\leq W$ in their final $n - ab - \ell$ positions. There are exactly $2^\ell \sum_{w \leq W} \binom{n-ab-\ell}{w}$ such patterns.

For each pattern p, compute the number of vectors v such that the noisy dot product $v \cdot e_i + \cdots$ matches $v \cdot p$. If this number is large enough (see the next subsection) then p is overwhelmingly likely to match the final $n - ab$ positions of e_i. If this number is not large enough for any choice of p then the algorithm stops.

Carrying out this computation separately for each pattern would require counting a total of $V 2^\ell \sum_{w \leq W} \binom{n-ab-\ell}{w}$ dot products, where $V \geq (q-1)n - 2^b a$ is the number of vectors v. For $W \geq 2$ it is much better to share work between similar patterns. The idea is simple: if p, p' differ in only one bit, say $p_b \neq p'_b$, then the number of v with $v \cdot p = v \cdot e_i + \cdots$ is the sum of

- the number of v with $v \cdot p = v \cdot e_i + \cdots$ and $v_b = 1$ and
- the number of v with $v \cdot p = v \cdot e_i + \cdots$ and $v_b = 0$,

while the number of v with $v \cdot p' = v \cdot e_i + \cdots$ is the sum of

- the number of v with $v \cdot p \neq v \cdot e_i + \cdots$ and $v_b = 1$ (the complement of the first summand above) and
- the number of v with $v \cdot p = v \cdot e_i + \cdots$ and $v_b = 0$ (the same as the second summand above).

Repeating the same split ℓ times obtains 2^ℓ patterns as a "fast Walsh transform" of 2^ℓ summands, each involving $V/2^\ell$ vectors on average. The summands require counting only $V \sum_{w \leq W} \binom{n-ab-\ell}{w}$ dot products.

Levieil and Fouque in [22] used the extreme case $\ell = n - ab$ to search through all possibilities for $n - ab$ bits of s. Kirchner in [20] used smaller ℓ, and $W < n - ab - \ell$, but required e_i to have Hamming weight $\leq W$ in its final $n - ab$ positions; our variant has higher success probability.

Statistics: Filtering Out the Noise. If $e_i = p$ then $v \cdot e_i + \cdots$ matches $v \cdot p$ exactly when the noise $\cdots$ equals 0. Recall that this occurs with probability $(1 + \delta^{2^a})/2$. In a pool of V vectors one expects an average of $V(1 + \delta^{2^a})/2$ matches, with a standard deviation proportional to $\sqrt{V}$.

If the final $n - ab$ bits of e_i do not match p then presumably $v \cdot e_i + \cdots + v \cdot p = v \cdot (e_i + p) + \cdots$ is set with probability $1/2$. (We do not claim that this analysis is provable.) In a pool of V vectors one then expects an average of $V/2$ matches, again with a standard deviation proportional to $\sqrt{V}$.

If the difference of averages, namely $V\delta^{2^a}/2$, is an order of magnitude larger than $\sqrt{V}$ then these two distributions have negligible overlap. The simplest strategy here is to define "large enough" as, e.g., $V/2 + 10\lceil\sqrt{V}\rceil$ (and require $\delta^{2^a}/2 \geq 20\lceil\sqrt{V}\rceil$) so that there is negligible chance of ever encountering a false positive; asymptotically 10 should be replaced by something growing (sublogarithmically) with the number of tests. A more complicated strategy is to define

"large enough" as, e.g., $V/2 + 2\lceil\sqrt{V}\rceil$, and to put more work into analyzing the occasional false positives.

Finishing Up. At this point the final $n-ab$ positions of e_i are known. Eliminate those positions from the original vectors $v_{j,k}$, obtaining a new problem with n replaced by ab, and solve that problem recursively.

4 Analysis and Optimization

This section analyzes the attack of the previous section, and gives several examples of reasonable parameter choices.

Success Probability and Cost. The chance that r_i is invertible depends on the selected ring but is indistinguishable from 1 for all Lapin proposals. The chance that e_i has Hamming weight $\leq W$ in its final $n - ab - \ell$ positions is exactly $\sum_{w \leq W} \binom{n-ab-\ell}{w} \tau^w (1-\tau)^{n-ab-\ell-w}$: for any particular weight w there are $\binom{n-ab-\ell}{w}$ choices of positions for w bits, chance τ of each of those bits being set, and chance $1-\tau$ of each of the other bits being clear. If $\sqrt{V}$ is sufficiently large compared to $2/\delta^{2^a}$ then any such e_i will in fact be recognized with overwhelming probability.

Each multiplication of r_j by $1/r_i$ costs $\leq 4n^2$ bit operations by standard techniques (and asymptotically $n^{1+o(1)}$ bit operations); the exact cost depends on the selected ring. Similar comments apply to the multiplication of r_j/r_i by $r_i s + e_i$, and to the computation of $v_{j,k}$, a "dual" multiplication. The initial computation of $1/r_i$ is easily amortized into $3 + o(1)$ multiplications. Overall there are $3q$ multiplications involved in computing the vectors $v_{j,k}$ and $\leq 2q$ additions, for a total of $\leq 12q(n^2 + n)$ bit operations.

Clearing bits involves at most $a(q-1)n^2$ bit xors. One of the factors n here is unnecessarily pessimistic: the first clearing xors only $n - b + 1$ bits, the second clearing xors only $n - 2b + 1$ bits, etc.

The dot products count at most $V \sum_{w \leq W} \binom{n-ab-\ell}{w} w$ bits. The fast Walsh transforms involve $\ell 2^\ell \sum_{w \leq W} \binom{n-ab-\ell}{w}$ additions of integers bounded by V. Each Walsh transform is performed separately, using a table of size 2^ℓ.

If e_i is not successful then the attack can try again, as many as q times, without any additional queries. If e_i is successful then the final recursion, determining the remaining bits of e_i, has negligible cost for any reasonable parameters.

An Attack against $n = 512$ with $\tau = 1/8$. Take $n = 512$, $\tau = 1/8$, $q = 5 \cdot 2^{35} + 3 \cdot 2^{21} + 1$, $a = 5$, $b = 44$, $W = 4$, and $\ell = 24$. Here $n - ab - \ell = 268$, and the chance that e_i has Hamming weight ≤ 4 in its final 268 positions is $\sum_{w \leq 4} \binom{268}{w} (1/8)^w (7/8)^{268-w} \approx 2^{-35.055}$. (For comparison: Requiring e_i to have Hamming weight ≤ 4 in its final $n - ab = 292$ positions, as in [20], would have chance only about $2^{-39.194}$.)

The initial q oracle outputs $(r_i, r_i s + e_i)$ consume $1024q$ bits of memory, about $5 \cdot 2^{42}$ bytes. The $(q-1)n$ expanded vectors $v_{j,k}$, together with their noisy dot products, consume $513(q-1)n$ bits of memory, about $5 \cdot 2^{53}$ bytes. The table

adds fewer than 2^{50} bytes. The computation of these vectors costs approximately $2^{58.910}$ bit operations. (For comparison: If we did not exploit the Ring-LPN structure then we would have to perform approximately q multiplications of $n \times n$ matrices, costing approximately $2^{65.3}$ bit operations by schoolbook matrix multiplication or somewhat fewer bit operations by fast matrix multiplication.)

Clearing $ab = 220$ bits produces at least $(q-1)n - 2^b a = 3 \cdot 2^{30}$ vectors; we discard any extra vectors so that $V = 3 \cdot 2^{30}$. The bias decreases to $(3/4)^{32} \approx 2^{-13.28}$, but $3 \cdot 2^{30}$ vectors easily filter out this level of noise. This clearing involves at most $a(q-1)n^2 \approx 2^{57.644}$ bit operations.

The dot products count approximately $2^{61.248}$ bits, the main bottleneck in the computation. The Walsh transforms use $2^{56.254}$ additions.

This computation is repeated $2^{35.055}$ times on average, for a total of about $2^{97.5}$ bit operations. We comment that all of these bit operations are easily vectorized. Once the final 292 bits of e_i are known, the Ring-LPN outputs for e_i are converted into Ring-LPN outputs for the first 220 bits of e_i, which are found recursively at much less cost. Once all of e_i is known, computing s is trivial.

To summarize, this attack finds the Ring-LPN-512 secret using $<2^{56}$ bytes of memory, $<2^{38}$ queries, and $<2^{98}$ bit operations, as announced in Section 1.

We chose these parameters to emphasize memory at some cost in bit operations. Instead taking $q = 2^{62} + 2^{61} + 2^{51}$, $a = 6$, $b = 69$, $W = 5$, and $\ell = 40$ would use $<2^{78}$ bytes of memory, $<2^{63}$ queries, and $<2^{88}$ bit operations, beating the algorithm of [22] both in space and in time.

An Attack against $n = 1024$ with $\tau = 1/20$. The following example illustrates the impact of τ. Take $n = 1024$, $\tau = 1/20$, $q = 10$, $a = 2$, $b = 12$, $W = 2$, and $\ell = 2$. Here $n - ab - \ell = 998$, and the chance that e_i has Hamming weight ≤ 2 in its final 998 positions is $\sum_{w \leq 2} \binom{998}{w} (1/20)^w (19/20)^{998-w} \approx 2^{-63.369}$.

The initial q oracle outputs consume 2560 bytes of memory, the $(q-1)n$ expanded vectors consume 1180800 bytes of memory, and the table consumes 519168 bytes of memory. The computation of these vectors costs approximately $2^{26.909}$ bit operations. Clearing $ab = 24$ bits produces at least $(q-1)n - 2^b a = 1024$ vectors, easily filtering out a bias of $0.9^4 = 0.6561$; this clearing involves at most $a(1-q)n^2 \approx 2^{24.170}$ bit operations. The dot products count approximately $2^{28.927}$ bits. The Walsh transforms use $2^{21.927}$ additions.

Overall one iteration of the attack uses $2^{29.373}$ bit operations. Running $2^{63.369}$ iterations of the attack uses $<2^{21}$ bytes of memory, $<2^{64}$ queries (with q iterations for each batch of q queries), and $<2^{93}$ bit operations.

For comparison, [22] says that $n = 1024$ and $\tau = 1/20$ take 2^{118} bytes of memory; the number of bit operations is even larger. We emphasize that the drastic reduction in memory consumption, and most of the reduction in bit operations, should be credited to Kirchner's paper [20].

A variant of the same attack is somewhat slower but reduces the total number of queries to just 10 and still fits into $<2^{21}$ bytes of memory. Instead of targeting 1 of 10 noisy vectors (producing 9 noisy vectors) and then forgetting the ring structure (producing 4608 noisy bits), first forget the ring structure (producing 5120 noisy bits) and then target 512 noisy bits (also producing 4608 noisy

bits). The remaining steps are exactly as before. This variant requires matrix multiplications instead of ring multiplications, but has the advantage of allowing $\binom{5120}{512} \approx 2^{2395}$ targets from the same 10 queries, far more than the $2^{63.369}$ targets needed on average. The same variant also works for LPN, using just 5120 queries.

References

[1] — (no editor): Second international workshop on security, privacy and trust in pervasive and ubiquitous computing (SecPerU 2006), 29 June 2006, Lyon, France. Institute of Electrical and Electronics Engineers (2006). See [6]

[2] Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. In: STOC 2000 [**28**], pp. 435–440 (2000); see also newer version [**3**]. Citations in this document: §2

[3] Blum, A., Kalai, A., Wasserman, H.: Noise-tolerant learning, the parity problem, and the statistical query model. Journal of the ACM 50, 506–519 (2003); see also older version [**2**]. Citations in this document: §1, §3

[4] Boyd, C. (ed.): Advances in cryptology — ASIACRYPT 2001: proceedings of the 7th international conference on the theory and application of cryptology and information security held on the Gold Coast, December 9–13, 2010, proceedings. LNCS, vol. 2248. Springer (2001). ISBN 3-540-42987-5. See [16]

[5] Bringer, J., Chabanne, H.: Trusted-HB: a low-cost version of HB^+ secure against man-in-the-middle attacks. IEEE Transactions on Information Theory, 4339–4342 (2008), http://eprint.iacr.org/2008/042. Citations in this document: §1

[6] Bringer, J., Chabanne, H., Dottax, E.: HB^{++}: a lightweight authentication protocol secure against some attacks. In: [**1**], pp. 28–33 (2006). Citations in this document: §1

[7] Canteaut, A. (ed.): Fast software encryption, 2012. 19th International Workshop, FSE 2012, Washington, DC, USA, March 19-21, 2012. Revised Selected Papers. LNCS, vol. 7549. Springer (2012). ISBN 978-3-642-34046-8. See [15]

[8] Duc, D.N., Kim, K.: Securing HB^+ against GRS man-in-the-middle attack. In: Proceedings of SCIS 2007 (2007). Citations in this document: §1

[9] Edwards, C., Edwards, T., Leech, T.: Hoodwinked! (2005), http://www.imdb.com/title/tt0443536/. Citations in this document: §⋆

[10] Eisenbarth, T., Gong, Z., Güneysu, T., Heyse, S., Indesteege, S., Kerckhof, S., Koeune, F., Nad, T., Plos, T., Regazzoni, F., Standaert, F.-X., van Oldeneel tot Oldenzeel, L.: Compact implementation and performance evaluation of block ciphers in ATtiny devices. In: Proceedings of the ECRYPT Workshop on Lightweight Cryptography, Louvain-la-Neuve, Belgium, November 2011; Africacrypt 2012, to appear (2011), http://perso.uclouvain.be/fstandae/lightweight_ciphers/. Citations in this document: §1

[11] Feldhofer, M., Dominikus, S., Wolkerstorfer, J.: Strong authentication for RFID systems using the AES algorithm. In: CHES 2004 [**17**], pp. 357–370 (2004). Citations in this document: §1

[12] Frumkin, D., Shamir, A.: Un-Trusted-HB: security vulnerabilities of Trusted-HB (2009), http://eprint.iacr.org/2009/044. Citations in this document: §1

[13] Gilbert, H., Robshaw, M.J.B., Seurin, Y.: $HB^\sharp$: increasing the security and efficiency of HB^+. In: EUROCRYPT 2008 [**27**], pp. 361–378 (2008). Citations in this document: §1

[14] Håstad, J.: Some optimal inapproximability results. In: STOC 1997 [**21**], pp. 1–10 (1997). Citations in this document: §2

[15] Heyse, S., Kiltz, E., Lyubashesvky, V., Paar, C., Pietrzak, K.: Lapin: An efficient authentication protocol based on Ring-LPN. In: FSE 2012 [**7**], pp. 346–365 (2012). Citations in this document: §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §1, §2, §2, §2

[16] Hopper, N.J., Blum, M.: Secure human identification protocols. In: ASIACRYPT 2001 [**4**], pp. 52–66 (2001). Citations in this document: §1, §2

[17] Joye, M., Quisquater, J.-J. (eds.): Cryptographic hardware and embedded systems — CHES 2004: 6th international workshop, Cambridge, MA, USA, August 11–13, 2004, proceedings. LNCS, vol. 3156. Springer (2004). ISBN 3-540-22666-4. See [11]

[18] Juels, A., Weis, S.A.: Authenticating pervasive devices with human protocols. In: CRYPTO 2005 [**26**], pp. 293–308 (2005). Citations in this document: §1

[19] Kiltz, E., Pietrzak, K., Cash, D., Jain, A., Venturi, D.: Efficient authentication from hard learning problems. In: EUROCRYPT 2011 [**24**], pp. 7–26 (2011). Citations in this document: §1, §2

[20] Kirchner, P.: Improved generalized birthday attack (2011), http://eprint.iacr.org/2011/377. Citations in this document: §1, §1, §3, §3, §3, §4, §4

[21] Leighton, F.T., Shor, P.W. (eds.): Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing, El Paso, Texas, USA, May 4-6, 1997. ACM (1997). See [14]

[22] Levieil, É., Fouque, P.-A.: An improved LPN algorithm. In: SCN 2006 [**25**], pp. 348–359 (2006). Citations in this document: §1, §1, §1, §1, §1, §1, §1, §1, §3, §3, §4, §4

[23] Munilla, J., Peinado, A.: HB-MP: a further step in the HB-family of lightweight authentication protocols. Computer Networks 51, 2262–2267 (2007). Citations in this document: §1

[24] Paterson, K.G. (ed.): Advances in cryptology — EUROCRYPT 2011, 30th annual international conference on the theory and applications of cryptographic techniques, Tallinn, Estonia, May 15–19, 2011, proceedings. LNCS, vol. 6632. Springer (2011). ISBN 978-3-642-20464-7. See [19]

[25] De Prisco, R., Yung, M. (eds.): Security and cryptography for networks, 5th international conference, SCN 2006, Maiori, Italy, September 6–8, 2006, proceedings. LNCS, vol. 4116. Springer (2006). ISBN 3-540-38080-9. See [22]

[26] Shoup, V. (ed.): Advances in cryptology — CRYPTO 2005: 25th annual international cryptology conference, Santa Barbara, California, USA, August 14–18, 2005, proceedings. LNCS, vol. 3621. Springer (2005). ISBN 3-540-28114-2. See [18]

[27] Smart, N.P. (ed.): Advances in cryptology — EUROCRYPT 2008, 27th annual international conference on the theory and applications of cryptographic techniques, Istanbul, Turkey, April 13–17, 2008, proceedings. LNCS, vol. 4965. Springer (2008). ISBN 978-3-540-78966-6. See [13]

[28] Yao, F.F., Luks, E.M. (eds.): Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing, May 21-23, 2000, Portland, OR, USA. ACM (2000). ISBN 1-58113-184-4. See [2]

On Using Instruction-Set Extensions for Minimizing the Hardware-Implementation Costs of Symmetric-Key Algorithms on a Low-Resource Microcontroller

Hannes Groß and Thomas Plos

Institute for Applied Information Processing and Communications (IAIK),
Graz University of Technology, Inffeldgasse 16a, 8010 Graz, Austria
Hannes.Gross@student.tugraz.at, Thomas.Plos@iaik.tugraz.at

Abstract. Due to the continuously increasing design complexity of passive radio-frequency identification (RFID) tags, relying on microcontroller-based architectures will become vital in the future. Re-using the microcontroller for multiple tasks, e.g., protocol handling and computing cryptographic algorithms is advantageous from a system point-of-view. In this work we present instruction-set extensions (ISEs) for minimizing the hardware-implementation costs of symmetric-key algorithms on a synthesizable 8-bit microcontroller. We have analyzed the block ciphers: Present-80, $SEA_{96,8}$, and XTEA. Integrating ISEs has reduced the hardware-implementation costs by 4 to 48 %. When considering the reuse of the microcontroller for protocol handling, overhead costs for implementing encryption and decryption functionality of the block ciphers are between 519 and 1 021 GEs for a 130 nm CMOS technology. Implementation costs for encryption-only versions are between 333 and 520 GEs. Our results emphasize that integrating ISEs for lowering the hardware-implementation costs of symmetric-key algorithms on low-resource microcontrollers is beneficial.

Keywords: Low-resource 8-bit microcontroller, passive RFID tags, symmetric-key cryptography, instruction-set extensions, low-area hardware implementation.

1 Introduction

Radio-frequency identification (RFID) technology is the enabler for a variety of new applications, a prominent example among others is the so-called Internet of Things. Many of these new applications will require RFID tags to support additional functionality, which increases their design complexity. Especially security functionality will play an important role [1]. In order to cope with this increased complexity of the tags, new design concepts such as programmable approaches are necessary. When relying on programmable approaches, applying optimization techniques from microprocessor domain like instruction-set extensions (ISEs) seems reasonable.

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 149–164, 2013.

Most RFID tags are passively powered (i.e., extract the power supply from the reader field) as they are much cheaper in price than, for example, active tags that require an extra battery. Another factor that influences the manufacturing costs of tags is chip size. Tags are designed to consume as little chip size as possible. This is typically achieved by using finite-state machine (FSM) approaches that are implemented as dedicated hardware circuits. However, when the functionality that tags have to provide increases and RFID communication protocols become more complex, using a fixed FSM approach is no longer practical and lacks agility. The result is a raise in production costs, longer time to market, and delayed re-design cycles.

Several publications have already demonstrated that using a programmable approach (e.g., a low-resource microcontroller) for implementing low-cost RFID tags is beneficial [23,32,34]. Programmable approaches provide much more flexibility that eases development and testing of a design by still fulfilling the stringent requirements of passive RFID tags. As shown in [24], a low-resource microcontroller that is used for handling the control tasks on a tag can also be reused for implementing cryptographic algorithms. This reuse has the advantage that expensive resources like memory can be utilized much more efficiently. Resulting overhead costs of the cryptographic algorithms in terms of chip size are lower as those of dedicated hardware modules.

The implementation of cryptographic algorithms is not limited to pure software or pure hardware solutions. Rather, there is also the concept of instruction-set extensions that combine the advantages from both software domain (higher flexibility) and hardware domain (increased performance). In 2000, Burke et al. [3] came up with suggestions for using ISEs to improve the performance of symmetric-key cryptography on microprocessors. Numerous publications that deal with ISEs and symmetric-key cryptography have followed over the years. For example, ISEs for the Advanced Encryption Standard (AES) [8,9,13,20,22,29,30], DES and 3-DES [22], and several SHA-3 candidates [5]. Grabher et al. [12] have also presented ISEs for so-called bit-sliced implementations of DES, Serpent, AES, Present, and SHA-1. All of these implementations aimed at improving the performance of the algorithms (i.e., reduce the execution time). Moreover, most of the publications address powerful 32-bit processors, exceptions are the works of Tillich et al. [30] and Constantin et al. [5] that focus on 8-bit and 16-bit microcontrollers, respectively. Although code-size reductions have been reported in most cases, overall implementation costs typically increase because of the hardware requirements of the integrated ISEs. The reason for this is that non of the published papers has used ISEs with the primary goal to lower the overall implementation costs of an algorithm, but aim at reducing the execution time instead.

In this work we present ISEs for symmetric-key algorithms on a low-resource 8-bit microcontroller that is suitable for future passive RFID tags. In contrast to related work we do not focus on maximizing the performance but on minimizing the overall implementation costs of the algorithms concerning chip size. As implementation costs we treat the program code in the ROM synthesized as

lookup table and the chip-size requirements of the ISEs. We analyze the block-cipher algorithms Present-80, $SEA_{96,8}$, and XTEA. The results for a 130 nm CMOS process technology illustrate that the ISEs allow to reduce the chip-size requirements by up to 48 % compared to a pure software implementation. When considering the presence of program code for protocol handling in the ROM, implementation costs of the block ciphers are even lower. Besides the reduction of the implementation costs, speedups between 1.20 and 3.97 are achieved. We conclude that ISEs are highly suitable to reduce the implementation costs of symmetric-key algorithms, which is of special interest for resource-constrained devices like passive RFID tags.

The remainder of this work is organized as follows. Section 2 provides a brief description of the deployed low-resource 8-bit microcontroller. In Section 3 the analyzed block ciphers are introduced in short. The algorithm-specific ISEs and the resulting implementation costs of the block ciphers are presented in Section 4. The influence on the implementation costs in the presence of additional program code for protocol handling is shown in Section 5, followed by conclusions in Section 6.

2 Description of the Low-Resource 8-Bit Microcontroller

The 8-bit microcontroller used in this work has been designed for simple controlling tasks with focus on low chip area and low power consumption. The microcontroller bases on a reduced instruction-set computer (RISC) architecture with separate program and data memory (i.e., a Harvard architecture). Both memories are freely scalable which allows adjusting their size during design phase exactly for the requirements of the desired application. The program memory (program ROM) is implemented as lookup table in hardware and is divided into several pages. Each page can hold up to 256 16-bit instruction words. A maximum of 256 pages is supported. Data memory is realized as register file with up to 64 8-bit registers. The register file consists of three special-purpose (SP) registers, input-output (I/O) registers, and general-purpose (GP) registers. SP registers are used for indicating status information (e.g., zero flag) of the arithmetic-logic unit (ALU), the paging mechanism of the program ROM, and a dedicated accumulator (ACC) register. I/O registers allow interfacing of external modules. GP registers are used for computing and temporarily storing data.

The instruction set of the microcontroller involves 36 instructions, which are divided into logical operations (AND, XOR), arithmetic operations (addition, subtraction), and control-flow operations (CALL, branching). Most of the operations are executed within a single clock cycle. For program development, a self-written instruction-set simulator and an assembler are used. The simulator allows easy and fast testing/debugging of programs, which is important for verifying the correct operation of complex protocols. Implementing such complex protocols directly in hardware via FSMs is typically not only more prone to errors but requires also more time when using hardware simulations. When a program is working properly, the assembler is used to generate a ROM file that can directly be integrated into the HDL model of the microcontroller.

Synthesizing the microcontroller for a 130 nm CMOS process technology leads to a chip area of about 5.3 kGEs[1] (including 64 registers but without program ROM). Power simulations with Cadence First Encounter resulted in a power consumption around 1 µA at a supply voltage of 1.2 V and a clock frequency of 100 kHz. More information about the microcontroller can be found in [24].

3 Overview of the Selected Block Ciphers

In the following section the selected block ciphers are introduced briefly, which are: Present, SEA, and XTEA. The block ciphers have been chosen because of their low implementation costs on the 8-bit microcontroller and their different design intentions, respectively. Present is designed for efficient implementation in hardware, whereas SEA and XTEA primarily focus on efficient software implementation. Selecting algorithms with different design intentions allows a better evaluation of the suitability of ISEs to minimize the implementation costs regarding chip size.

3.1 PRESENT

Present was designed by Bogdanov et al. in 2007 [2] and has been integrated in the ISO/IEC 29192-2 standard for lightweight cryptography in 2012 [15]. It supports key sizes of 80 and 128 bits and is applied on plain-text blocks with a length of 64 bits. In this work we only focus on the 80-bit version (Present-80). An encryption/decryption operation consists of 31 substitution-permutation network (SPN) rounds. The substitution layer of the SPN can be implemented with a single 4-bit S-box that is sequentially applied on the 16×4-bit block of the state. In the permutation layer, the state bits are mixed bitwise in a regular way. Present is a hardware-oriented algorithm intended for resource-constrained applications like passive RFID tags where chip area and power consumption are limited. The best attack on Present-80 applies on 26 rounds which has been published by Cho in 2010 [4].

3.2 SEA

Standaert et al. presented the Scalable Encryption Algorithm (SEA) which focuses on small embedded applications in 2006 [28]. The intention for designing this algorithm was its suitability for processors with limited instruction set. SEA is not designed for processors with a specific data width, as key and block size are variable (must be a multiple of 6 $\times$ processor data width). The recommended number of rounds also depends on key and block size. For our work we have selected $\text{SEA}_{96,8}$ (i.e., key and block size are 96 bits) which aims at 8-bit microcontroller platforms and uses 93 rounds. SEA is software oriented and uses simple operations like AND, OR, XOR, and modular addition. Another intention

[1] One gate equivalent (GE) is the chip area consumed by a 2-input NAND gate.

of the designers has been to provide provable security. The designers claim that the only weakness of SEA could be its simple structure, making it potentially vulnerable to algebraic attacks.

3.3 XTEA

In 1994, Wheeler et al. introduced the Tiny Encryption Algorithm (TEA) [31], which is a software oriented 64-bit block cipher with a 128-bit key and 32 encryption/decryption rounds (corresponding to 64 Feistel rounds). The algorithm has been designed especially for 32-bit microcontrollers with a limited instruction set. The reference implementation needs less than 20 lines of C code (for encryption and decryption) and consists of shift, AND, XOR, and addition operations. As TEA was broken in 1997, Wheeler et al. introduced extensions for TEA leading to its successor the extended TEA (XTEA) [21]. The extensions mainly address the weak key scheduling of TEA. The best attack on XTEA is a related-key rectangle attack on 36 Feistel rounds of the block cipher, published by Lu in 2008 [17].

4 Instruction-Set Extensions

This section describes the implemented ISEs and presents the achieved implementation results for a 130 nm standard-cell CMOS technology from Faraday [10]. Synthesis is done with Cadence RTL compiler. For every block cipher three software implementations for our synthesizable low-resource microcontroller with the optimization targets *speed* (shortest execution time), *balanced* (trade-off between code size and execution time), and *size* (smallest code size) are considered. The implementations presented in [24] served as starting point for our work.

For each implementation, we compare the basic version (without using any ISE) with versions that have been obtained by implementing only a single ISE as well as a version that combines all proposed ISEs for a concerning algorithm. Two tables are presented in this section that give an overview of the implementation results. Table 1 shows the impact of the ISEs on the program size and the execution time for encrypting/decrypting data. Table 2 compares the impact of the ISEs on the hardware-implementation costs of the algorithms. As implementation costs we consider the area requirement of the ISEs and the part of the program ROM that relates to the code for the algorithm implementation. The ROM is realized as lookup table, which gets mapped by the synthesizer to an unstructured mass of standard cells. Registers that are used for the algorithm implementation are not counted as additional costs, as we assume that they are already used by the tag for protocol handling and thus can be re-used without causing further costs. Present-80 requires for example 29 registers, $\text{SEA}_{96,8}$ between 29 (optimization target speed) and 30 registers (optimization targets balanced and size), and XTEA 37 registers. The implemented ISEs do not affect the number of registers required by an algorithm.

Table 1. Impact of the implemented ISEs on program size and execution time of the block ciphers

Algorithm	Target	Implementation	Program Size	Program Rel. change	Encryption Exec. time	Encryption Speed-up	Decryption Exec. time	Decryption Speed-up
			[Bytes]	[%]	[Cycles]	[-]	[Cycles]	[-]
Present-80	Speed	No extensions	2 170	0	8 985	1.00	11 591	1.00
		Sbox	**1 032**	**-52**	**7 315**	**1.23**	**9 639**	**1.20**
		Perm	**1 674**	**-23**	**5 114**	**1.76**	**7 748**	**1.50**
		Sbox+Perm	**532**	**-75**	**3 410**	**2.63**	**5 734**	**2.02**
	Balanced	No extensions	1 168	0	15 042	1.00	17 677	1.00
		Sbox	**882**	**-24**	**7 819**	**1.92**	**10 175**	**1.74**
		Perm	**686**	**-41**	**11 074**	**1.36**	**13 709**	**1.29**
		Sbox+Perm	**398**	**-66**	**3 790**	**3.97**	**6 146**	**2.88**
	Size	No extensions	944	0	28 062	1.00	60 426	1.00
		Sbox	**810**	**-14**	**15 662**	**1.79**	**46 694**	**1.29**
		Perm	**462**	**-51**	**24 094**	**1.16**	**56 459**	**1.07**
		Sbox+Perm	**324**	**-66**	**11 478**	**2.44**	**42 479**	**1.42**
	-	AVR [25]	2 398	-	9 595	-	9 820	-
	-	AVR [7]	936	-	10 723	-	11 239	-
	-	AVR [6]	1 000	-	11 342	-	13 599	-
$SEA_{96,8}$	Speed	No extensions	806	0	8 053	1.00	8 053	1.00
		Swap	**696**	**-14**	**8 145**	**0.99**	**8 145**	**0.99**
		SboxRot	**614**	**-24**	**5 093**	**1.58**	**5 093**	**1.58**
		Swap+SboxRot	**504**	**-37**	**5 185**	**1.55**	**5 185**	**1.55**
	Balanced	No extensions	504	0	8 597	1.00	8 597	1.00
		Swap	**332**	**-34**	**8 689**	**0.99**	**8 689**	**0.99**
		SboxRot	**376**	**-25**	**5 637**	**1.52**	**5 637**	**1.52**
		Swap+SboxRot	**268**	**-47**	**5 729**	**1.50**	**5 729**	**1.50**
	Size	No extensions	350	0	14 723	1.00	14 723	1.00
		Swap	**270**	**-23**	**10 767**	**1.37**	**10 767**	**1.37**
		SboxRot	**332**	**-5**	**9 823**	**1.50**	**9 823**	**1.50**
		Swap+SboxRot	**252**	**-28**	**5 867**	**2.51**	**5 867**	**2.51**
	-	AVR [26]	2 132	-	9 654	-	9 654	-
	-	AVR [28]	386	-	17 745	-	17 745	-
	-	AVR [6]	426	-	41 604	-	40 860	-
XTEA	Speed	No extensions	1 148	0	7 263	1.00	7 776	1.00
		AddC	**830**	**-28**	**5 183**	**1.40**	**5 664**	**1.37**
		SubC	**1 120**	**-2**	**7 263**	**1.00**	**7 582**	**1.03**
		Swap	**726**	**-37**	**7 903**	**0.92**	**8 353**	**0.93**
		AddC+SubC+Swap	**532**	**-54**	**5 823**	**1.25**	**6 079**	**1.28**
	Balanced	No extensions	700	0	8 702	1.00	9 242	1.00
		AddC	**540**	**-23**	**6 558**	**1.33**	**7 162**	**1.29**
		SubC	**686**	**-2**	**8 702**	**1.00**	**9 048**	**1.02**
		Swap	**612**	**-13**	**7 998**	**1.09**	**8 516**	**1.09**
		AddC+SubC+Swap	**438**	**-37**	**5 854**	**1.49**	**6 242**	**1.48**
	Size	No extensions	464	0	13 024	1.00	13 485	1.00
		AddC	**398**	**-14**	**11 008**	**1.18**	**11 469**	**1.18**
		SubC	**450**	**-3**	**13 024**	**1.00**	**13 291**	**1.01**
		Swap	**458**	**-1**	**12 832**	**1.01**	**13 271**	**1.02**
		AddC+SubC+Swap	**378**	**-19**	**10 816**	**1.20**	**11 077**	**1.22**
	-	AVR [26]	1 160	-	6 718	-	6 718	-
	-	PIC [19]	962	-	7 408	-	7 408	-

In the following, details about the ISEs are given that have been implemented for reducing the overhead costs of the block ciphers Present-80, $SEA_{96,8}$, and XTEA. The implementations support both encryption and decryption operation of the algorithms. For comparison, also results for encryption-only versions are provided, which could be interesting for RFID applications where decryption functionality is not required on the tag.

Table 2. Impact of the ISEs on the hardware-implementation costs of the block ciphers

Algorithm	Target	Implementation	Hardware costs			Area efficiency	Area change	
			ISEs	ROM area	Total			
			[GEs]	[GEs]	[GEs]	[Bits/GE]	[GEs]	[%]
Present-80	Speed	No extensions	0	2 278	2 278	7.62	0	0
		Sbox	**117**	**1 623**	**1 740**	**5.09**	**-538**	**-24**
		Perm	**375**	**1 872**	**2 247**	**7.16**	**-31**	**-1**
		Sbox+Perm	**409**	**1 135**	**1 544**	**3.75**	**-734**	**-32**
	Balanced	No extensions	0	1 882	1 882	4.96	0	0
		Sbox	**117**	**1 423**	**1 540**	**4.96**	**-342**	**-18**
		Perm	**373**	**1 608**	**1 981**	**3.41**	**98**	**5**
		Sbox+Perm	**408**	**1 014**	**1 422**	**3.14**	**-460**	**-24**
	Size	No extensions	0	1 524	1 524	4.96	0	0
		Sbox	**120**	**1 171**	**1 291**	**5.54**	**-233**	**-15**
		Perm	**374**	**1 172**	**1 546**	**3.15**	**22**	**1**
		Sbox+Perm	**411**	**833**	**1 244**	**3.11**	**-280**	**-18**
$SEA_{96,8}$	Speed	No extensions	0	1 786	1 786	3.61	0	0
		Swap	**23**	**1 583**	**1 606**	**3.52**	**-180**	**-10**
		SboxRot	**170**	**1 474**	**1 644**	**3.33**	**-142**	**-8**
		Swap+SboxRot	**169**	**1 232**	**1 401**	**3.27**	**-385**	**-22**
	Balanced	No extensions	0	1 202	1 202	3.36	0	0
		Swap	**23**	**861**	**884**	**3.08**	**-317**	**-26**
		SboxRot	**170**	**995**	**1 165**	**3.02**	**-37**	**-3**
		Swap+SboxRot	**169**	**771**	**940**	**2.78**	**-262**	**-22**
	Size	No extensions	0	910	910	3.08	0	0
		Swap	**23**	**779**	**802**	**2.77**	**-108**	**-12**
		SboxRot	**169**	**907**	**1 076**	**2.93**	**166**	**18**
		Swap+SboxRot	**169**	**708**	**877**	**2.85**	**-33**	**-4**
	-	Mace [18]	-	-	3 758	-	-	-
XTEA	Speed	No extensions	0	2 283	2 283	4.02	0	0
		AddC	**1**	**1 675**	**1 676**	**3.96**	**-607**	**-27**
		SubC	**3**	**2 185**	**2 188**	**4.10**	**-95**	**-4**
		Swap	**16**	**1 704**	**1 720**	**3.40**	**-563**	**-25**
		AddC+SubC+Swap	**26**	**1 154**	**1 180**	**3.69**	**-1103**	**-48**
	Balanced	No extensions	0	1 687	1 687	3.32	0	0
		AddC	**1**	**1 243**	**1 244**	**3.47**	**-445**	**-26**
		SubC	**6**	**1 562**	**1 568**	**3.51**	**-119**	**-7**
		Swap	**16**	**1 431**	**1 447**	**3.42**	**-240**	**-14**
		AddC+SubC+Swap	**25**	**975**	**1 000**	**3.59**	**-687**	**-41**
	Size	No extensions	0	1 105	1 105	3.36	0	0
		AddC	**1**	**956**	**957**	**3.33**	**-148**	**-13**
		SubC	**8**	**1 113**	**1 121**	**3.23**	**16**	**1**
		Swap	**18**	**1 080**	**1 098**	**3.39**	**-8**	**-1**
		AddC+SubC+Swap	**31**	**914**	**945**	**3.31**	**-161**	**-14**
	-	Feldhofer [11]	-	-	2 636	-	-	-

4.1 PRESENT-80

As Present is mainly hardware oriented it offers several promising improvement candidates for ISEs. The first one is the implementation of the 4-bit S-box and its inverse in hardware. This is done by implementing a dedicated *Sbox* instruction that operates on a whole 8-bit register value. The microcontroller's ALU is extended with an extra 3-to-1 multiplexer and two 4-bit S-boxes each for both substitution and inverse substitution as illustrated in Figure 1. Since the three original software implementations of Present use different approaches for realizing the S-box lookup (e.g., single 4-bit or combined 8-bit table), also the impact

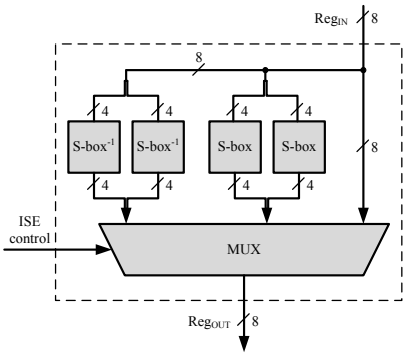


Fig. 1. Hardware extensions of the *Sbox* instruction for Present-80 within the ALU

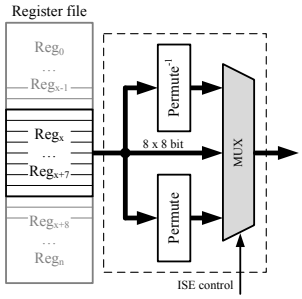


Fig. 2. Hardware extensions of the *Perm* instruction for Present-80 within the register file

of the *Sbox* instruction on the implementation costs is different. As shown in Table 2, the area gain in GEs of the speed-optimized version is more than two times the area gain of the size-optimized version. The reverse effect can be seen for the speedup which is slightly higher for the size-optimized version (cf. Table 1).

Bit-wise permutation is also very expensive in software, because for every permuted bit a branch and a bit-set instruction are needed. In hardware this is just re-wiring. For the 64-bit state of Present, the permutation and inverse permutation operation are integrated into eight dedicated registers (Reg_x to Reg_{x+7}) as depicted in Figure 2. The *Perm* instruction changes the output of these registers either to permuted, inverse permuted or to the original value. The area savings that can be obtained with the *Perm* instruction are much lower than those of the *Sbox* instruction. However, execution time of the algorithm implementations can be reduced by approximately 4000 clock cycles.

When combining the *Sbox* and *Perm* instructions, area savings between 18 (size-optimized version) and 32 % (speed-optimized version) can be achieved. Speedups between 2.44 and 3.97 for encryption and 1.42 and 2.88 for decryption can be realized with the ISEs compared to the pure software implementations. An interesting observation concerning the *Perm* instruction is that using it alone leads only to marginal area savings, but when applying it in combination with the *Sbox* instruction, larger area savings are possible. In that way, encryption and decryption operation of Present can be implemented with overhead costs between 1 244 and 1 544 GEs.

4.2 $SEA_{96,8}$

SEA is designed for efficient implementation on devices with limited instruction set like our low-resource microcontroller. Hence, code size of the pure software implementations of SEA on our microcontroller is already quite compact, compared for example to the implementations of Present. Each Feistel round of SEA

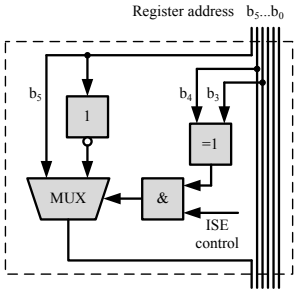


Fig. 3. Hardware extensions of the *Swap* instruction used for $SEA_{96,8}$

Register address	Register content
000xxx	Static registers
001xxx	Left half of state
010xxx	Left half of key
011xxx	Static registers
100xxx	Static registers
101xxx	Right half of state
110xxx	Right half of key
111xxx	Static registers

Swappable

Fig. 4. Register organization of state and key when using the *Swap* instruction for $SEA_{96,8}$

works only on one half of the state and on one half of the key, which are interchanged afterwards. Thus, SEA can be realized by implementing two Feistel rounds in software (one for each half) that are executed in an alternating manner, or more efficiently by implementing only one Feistel round in software and by swapping the two halves of state and key before starting the next round. Swapping the two halves in software requires 18 move instructions on our microcontroller platform.

The first ISE that we have implemented makes swapping of the two halves of state and key easier. Therefore, the 6-bit address space of the microcontroller's register file is partitioned into a static and a swappable part. We have defined 32 registers as swappable (with addresses `x01xxx` and `x10xxx`), as storing state and key of SEA requires 24 registers (32 is the next-larger power of two). When register swapping is enabled through the *Swap* instruction, the most-significant bit of the register address is inverted if it is a swappable register (i.e., register address `001xxx` becomes `101xxx` and vice versa), as illustrated in Figure 3. By storing state and key in the register file as shown in Figure 4, swapping between the two state/key halves is done in a single clock cycle by simply changing the register addressing. The *Swap* instruction can also be used for other block ciphers that have a Feistel structure. The area gain of this ISE is between 108 GEs (size-optimized version) and 317 GEs (balanced version) as illustrated in Table 2. Area savings are larger for the speed-optimized and the balanced versions as they implement two rounds of SEA in the pure software version (one for each half of state and key), which is no longer necessary when using the *Swap* instruction. The situation is different for the achievable speedup (cf. Table 1), where only the size-optimized version can benefit (speedup of 1.37), as it uses explicit register swapping in software for the basic version without ISEs.

The second ISE combines two operations of SEA which are consecutively used for the state calculation as well as for the key schedule. The *SboxRot* instruction operates on three bytes at once. First, eight 3-bit S-box lookups are performed on the three bytes, followed by a bit rotation. For this ISE three dedicated registers have been used. When the *SboxRot* instruction is active, the substituted

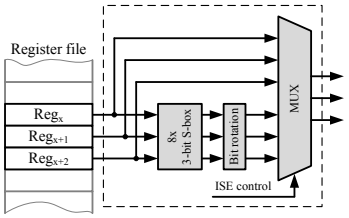


Fig. 5. Hardware extensions of the *SboxRot* instruction for SEA$_{96,8}$ within the register file

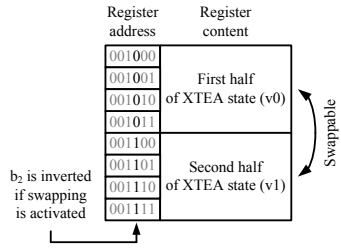


Fig. 6. Register organization of the XTEA state for implicit swapping of the state halves v0 and v1 via the *Swap* instruction

and rotated result of the register output appears instead of the original register content as depicted in Figure 5. The efficiency of this ISE also depends on the number of Feistel rounds implemented in the original software version. The instruction allows to achieve area savings between 37 and 142 GEs for the speed-optimized and balanced version, as they implement two Feistel rounds in the original version. For the size-optimized version, which originally has only one round implemented, the extension has even a negative effect on the overall area but still gives a speedup of 1.50.

As the basic software implementations of SEA (i.e., no ISEs are used) are already quite compact, achievable savings are lower as in case of Present. By using the *Swap* instruction the balanced and the size-optimized version of SEA can be realized with 884 and 802 GEs overhead costs, respectively. Speedup is between 0.99 and 1.37. For the speed-optimized version, minimal overhead costs are achieved by combining the *Swap* and the *SboxRot* instructions, leading to 1 401 GEs (385 GEs savings) and by bringing a speedup of 1.55. In that way, our implementations of SEA are 2.68 to 4.68 times smaller than a pure hardware implementation (the so far smallest stand-alone hardware implementation of SEA consumes 3 758 GEs [18]).

4.3 XTEA

XTEA is optimized for a 32-bit microcontroller architecture and uses 32-bit addition for encryption and its inverse (i.e., 32-bit subtraction) for decryption. The 8-bit microcontroller used in this work only supports 8-bit addition without considering an input carry. In the ALU of the microcontroller the ability for incrementing an operand is already implemented, as it is needed for certain instructions of the basic instruction set. Hence, realizing the *AddC* instruction, which considers a carry-status bit when adding the content of two 8-bit registers is rather cheap, but has a very positive effect on the program size and the execution time for all algorithm variants. Area savings between 148 and 607 GEs are achieved. Speedup is in the range of 1.18 to 1.40.

The second ISE for XTEA addresses optimization of subtraction which is needed for the decryption operation. In the original configuration of the microcontroller, there is only constant-to-accumulator subtraction implemented. Hence, the best way to subtract the value of one register from another one is calculating the two's complement of one register and adding it to the second one. Calculating the two's complement is done by inverting the content of a register first and incrementing it by one afterwards. Those two steps are done by the *SubC* instruction in one clock cycle. The instruction has a positive effect on the implementation costs of the speed optimized and balanced versions, leading to savings of 95 and 119 GEs, respectively. Speedup is only marginal. Both the *AddC* and the *SubC* instructions will also have a positive effect on the implementation costs of other algorithms that use addition or subtraction of larger bit widths.

XTEA bases on a Feistel structure similar to SEA. One round of XTEA consists of two Feistel rounds which are almost identical but the left and right halves of the state are interchanged. In software, interchanging the 64-bit state requires 24 move instructions. By using a *Swap* instruction, interchanging of the state halves is done implicitly by changing the register-addressing scheme (very similar to the *Swap* instruction used for SEA). One half of the state (v0) is stored for example in registers 8 to 11 and the other half (v1) in registers 12 to 15, as illustrated in Figure 6. With the *Swap* instruction, bit b_2 of the register address is inverted, leading to the effect that v0 and v1 appear interchanged when being accessed. This ISE has most impact on the speed optimized and the balanced version of XTEA as they have implemented two Feistel rounds in the pure software variant, saving 240 to 563 GEs of area. However, the large area savings of the speed-optimized version come at cost of slightly longer execution times.

When combining all three ISEs (*AddC*, *SubC*, and *Swap*), area savings between 14 % (161 GEs) and 48 % (1 103 GEs) can be achieved as shown in Table 2. Total implementation costs of XTEA are in the range of 945 GEs for the size-optimized version to 1 180 GEs for the speed-optimized version. Compared to the smallest stand-alone hardware module of XTEA, overhead costs of our implementations are 2.23 to 2.79 times lower. Achievable speedup is between 1.20 and 1.49 compared to a pure software implementation. The additional hardware costs of the ISEs are rather cheap compared to the ISEs of the other two algorithms. Integrating more-advanced ISEs such as XORing or adding on 32-bit basis would lead to a further speedup, but also significantly increase the implementation costs. XORing two 32-bit values, for example, would require at least 32 extra XOR gates and 32 extra multiplexers. This corresponds to overhead costs of more than 160 GEs for this ISE, which are far above the savings that result from the code-size reduction. ISE costs for realizing 32-bit addition are even higher.

4.4 Encryption-Only Implementations

For applications where only encryption functionality is required, implementation costs of the block ciphers can be further reduced by omitting decryption function-

Table 3. Program size, execution time, and hardware costs of the encryption-only implementations of the block ciphers

Algorithm	Target	Implementation	Program size	Exec. time	Hardware costs			Area efficieny
					ISEs	ROM area	Total	
			[Bytes]	[Cycles]	[GEs]	[GEs]	[GEs]	[Bits/GE]
Present-80	Optimal	**Sbox+Perm**[1]	**224**	**3 406**	**213**	**575**	**788**	**3.12**
	-	Rolfes [27]	-	547	-	-	1 075	-
	-	Yap [33]	-	516	-	-	1 030	-
$SEA_{96,8}$	Speed	**Swap+SboxRot**	**270**	**5 173**	**161**	**715**	**876**	**3.02**
	Size		**192**	**5 843**	**161**	**556**	**717**	**2.76**
XTEA	Speed	**AddC+Swap**	**264**	**4 739**	**13**	**572**	**585**	**3.69**
	Size		**254**	**9 732**	**13**	**685**	**698**	**2.96**

ality. We have implemented encryption-only versions of all three block ciphers by using the ISEs described above. For Present, a single version (*optimal* version) has been implemented that provides both shortest execution time and smallest code size. For SEA and XTEA we have implemented speed and sized-optimized versions. Implementations with optimization target balanced have been omitted as they led to very similar results as the size-optimized versions. An overview of the hardware costs of the encryption-only implementations is given in Table 3.

When implementing only encryption functionality of Present, inverse substitution and inverse permutation can be removed from the *Sbox* and *Perm* instructions, saving around 200 GEs. In total, area requirements of Present can be reduced by more than 450 GEs, resulting in overall overhead costs of 788 GEs. Our implementation is significantly smaller than the dedicated hardware modules presented in [27,33]. Removing decryption functionality from SEA saves between 160 and 525 GEs, leading to area requirements of 717 and 876 GEs, respectively. Savings between 247 and 595 GEs have been achieved for XTEA, where we have not used the *SubC* ISE as it only affects decryption operation. In case of XTEA, the speed-optimized version led to even lower hardware costs than the size-optimized version, although it has a code size that is 10 bytes larger (due to better optimization by the synthesizer). XTEA is the algorithm with the smallest area, consuming less than 700 GEs.

5 Overhead Costs of the Block Ciphers in Presence of an RFID Communication Protocol

Using a programmable tag architecture provides not only much more flexibility and agility during the design phase of a tag than dedicated hardware concepts, but allows also a more efficient reuse of components. When using for example the microcontroller on the tag for both handling the RFID protocol and computing the cryptographic algorithm, area efficiency of the program ROM improves

[1] Inverse substitution and inverse permutation functionality has been removed from the *Sbox* and *Perm* instructions.

Table 4. Hardware costs of the block ciphers when considering a combined implementation of protocol handling and cryptographic algorithm in the program ROM

Algorithm	Enc/dec	Target	Implementation	Hardware costs			Area efficieny	Rel. area change
				ISEs	ROM area	Total		
				[GEs]	[GEs]	[GEs]	[Bits/GE]	[%]
Present-80	Enc/dec	Speed	**Sbox+Perm**	**413**	**865**	**1 278**	**4.92**	**-17**
		Balanced		**413**	**617**	**1 030**	**5.13**	**-28**
		Size		**414**	**607**	**1 021**	**4.27**	**-18**
	Enc	Optimal	**Sbox+Perm**[1]	**222**	**298**	**520**	**6.02**	**-34**
$SEA_{96,8}$	Enc/dec	Speed	**Swap+SboxRot**	**170**	**985**	**1 155**	**4.09**	**-18**
		Balanced		**169**	**410**	**579**	**5.24**	**-38**
		Size		**169**	**350**	**519**	**5.76**	**-41**
	Enc	Speed	**Swap+SboxRot**	**169**	**371**	**540**	**5.82**	**-38**
		Size		**168**	**279**	**447**	**5.51**	**-38**
XTEA	Enc/dec	Speed	**AddC+SubC+Swap**	**30**	**954**	**984**	**4.46**	**-42**
		Balanced		**30**	**618**	**648**	**5.67**	**-35**
		Size		**30**	**578**	**608**	**5.20**	**-36**
	Enc	Speed	**AddC+Swap**	**19**	**314**	**333**	**6.73**	**-43**
		Size		**20**	**340**	**360**	**5.97**	**-48**

and overhead costs of the cryptographic algorithm decrease. The reason for this behavior is that the additional code used by the microcontroller for protocol handling leads to a larger program ROM (has to store more instructions in the lookup table), but the synthesizer can better optimize larger lookup tables by removing redundancies within them. Hence, the number of bits that can be represented by a single GE increases when integrating additional functionality (i.e., area efficiency increases).

In order to show the impact of protocol handling on the overhead costs of our algorithm implementations, we have integrated code for processing the RFID protocols ISO/IEC 14443-4 [16] and ISO/IEC 7816-4 [14] to the program ROM of our microcontroller. Protocol handling requires 2 142 bytes of code, resulting in a chip area of 5 424 GEs after synthesis. When adding code for processing a cryptographic algorithm and synthesizing the resulting program ROM, overhead costs introduced by the cryptographic algorithm can be determined. In that way, we have calculated the total implementation costs of the cryptographic algorithms (including costs for ISEs) in presence of protocol handling. The results for encryption/decryption and encryption-only variants are summarized in Table 4. Compared to the stand-alone implementations presented in Section 4, hardware costs of the algorithms are reduced by 17 to 48 % (relative area change with respect to the values given in Table 2 and Table 3). For realizing both encryption and decryption functionality, SEA has the lowest area requirements, consuming only 519 to 1 155 GEs. The smallest encryption-only implementation has been achieved with XTEA, requiring only 333 GEs (the speed-optimized version has been again slightly smaller). Our results clearly illustrate that using a programmable approach that is supported by ISEs for computing cryptographic

[1] Inverse substitution and inverse permutation functionality has been removed from the *Sbox* and *Perm* instructions.

algorithms is highly advantageous for resource-constrained devices like passive RFID tags.

6 Conclusion

In this work we showed that ISEs can be used to significantly reduce the implementation costs of symmetric-key algorithms on a constrained 8-bit microcontroller platform. We analyzed the three block ciphers: Present-80, $SEA_{96,8}$, and XTEA. Integrating ISEs into the microcontroller resulted in program code that is up to 75 % smaller, leading to 4 to 48 % lower hardware-implementation costs when synthesizing the program ROM as lookup table and also considering the additional hardware costs of the ISEs. When assuming that the program ROM also contains code for handling an RFID communication protocol, which is the natural use case for such a constrained microcontroller, implementation costs of the block ciphers further decrease by 17 to 48 %. For a 130 nm CMOS technology, encryption and decryption operation of Present-80 can be implemented with overhead costs of about 1000 GEs, $SEA_{96,8}$ and XTEA with around 600 GEs. Encryption-only implementations that omit decryption functionality can be realized with 333 to 520 GEs. Speedups are between 1.20 and 3.97 compared to a pure software implementation on the microcontroller. We conclude that integrating ISEs into resource-constrained microcontrollers is advantageous for lowering the overall implementation costs of cryptographic algorithms, making it a promising design approach for future low-cost RFID tags.

Acknowledgements. This work has been supported by the European Commission through the ICT programme under contract ICT-2007-216676 (ECRYPT II).

References

1. Aigner, M., Burbridge, T., Ilic, A., Lyon, D., Soppera, A., Lehtonen, M.: RFID Tag Security. Building Radio Frequency IDentification for the Global Environment (BRIDGE), European Commission Contract No. IST-2005-03346, White Paper (2008)
2. Bogdanov, A.A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: An Ultra-Lightweight Block Cipher. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007)
3. Burke, J., McDonald, J., Austin, T.: Architectural Support for Fast Symmetric-Key Cryptography. In: ASPLOS-IX Proceedings of the 9th International Conference on Architectural Support for Programming Languages and Operating Systems, Cambridge, MA, USA, November 12-15, pp. 178–189. ACM Press, New York (2000)
4. Cho, J.Y.: Linear Cryptanalysis of Reduced-Round PRESENT. In: Pieprzyk, J. (ed.) CT-RSA 2010. LNCS, vol. 5985, pp. 302–317. Springer, Heidelberg (2010)

5. Constantin, J., Burg, A., Gürkaynak, F.K.: Investigating the Potential of Custom Instruction Set Extensions for SHA-3 Candidates on a 16-bit Microcontroller Architecture. Cryptology ePrint Archive, Report 2012/050 (2012), http://eprint.iacr.org/
6. Eisenbarth, T., Gong, Z., Güneysu, T., Heyse, S., Indesteege, S., Kerckhof, S., Koeune, F., Nad, T., Plos, T., Regazzoni, F., Standaert, F.-X., van Oldeneel tot Oldenzeel, L.: Compact Implementation and Performance Evaluation of Block Ciphers in ATtiny Devices. In: Mitrokotsa, A., Vaudenay, S. (eds.) AFRICACRYPT 2012. LNCS, vol. 7374, pp. 172–187. Springer, Heidelberg (2012)
7. Eisenbarth, T., Kumar, S., Paar, C., Poschmann, A., Uhsadel, L.: A Survey of Lightweight-Cryptography Implementations. IEEE Design & Test of Computers - Design and Test of ICs for Secure Embedded Computing 24(6), 522–533 (2007) ISSN 0740-7475
8. Elbirt, A.J.: Fast and Efficient Implementation of AES via Instruction Set Extensions. In: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW 2007), vol. 1, pp. 396–403. IEEE Computer Society (May 2007)
9. EnSilica. A study of AES and its efficient implementation on eSi-RISC. WhitePaper (January 2010)
10. Faraday Technology Corporation. Faraday FSA0A_C 0.13 μm ASIC Standard Cell Library (2004), Details available online, http://www.faraday-tech.com
11. Feldhofer, M., Wolkerstorfer, J.: Hardware Implementation of Symmetric Algorithms for RFID Security. In: RFID Security: Techniques, Protocols and System-On-Chip Design, pp. 373–415. Springer (2008)
12. Grabher, P., Großschädl, J., Page, D.: Light-Weight Instruction Set Extensions for Bit-Sliced Cryptography. In: Oswald, E., Rohatgi, P. (eds.) CHES 2008. LNCS, vol. 5154, pp. 331–345. Springer, Heidelberg (2008)
13. Hodjat, A., Verbauwhede, I.: Interfacing a High Speed Crypto Accelerator to an Embedded CPU. In: Conference Record of the Thirty-Eighth Asilomar Conference on Signals, Systems, and Computers, vol. 1, pp. 488–492. IEEE (November 2004)
14. International Organisation for Standardization (ISO). ISO/IEC 7816-4: Information technology - Identification cards - Integrated circuit(s) cards with contacts - Part 4: Interindustry commands for interchange (1995), http://www.iso.org
15. International Organisation for Standardization (ISO). ISO/IEC 29192-2: Information technology – Security techniques – Lightweight cryptography – Part 2: Block ciphers (2012), http://www.iso.org
16. International Organization for Standardization (ISO). ISO/IEC 14443-4: Identification Cards - Contactless Integrated Circuit(s) Cards - Proximity Cards - Part4: Transmission Protocol (2008), http://www.iso.org
17. Lu, J.: Related-key rectangle attack on 36 rounds of the XTEA block cipher. International Journal of Information Security 8, 1–11 (2009)
18. Mace, F., Standaert, F.-X., Quisquater, J.-J.: ASIC Implementations of the Block Cipher SEA for Constrained Applications. In: Munilla, J., Peinado, A., Rijmen, V. (eds.) Workshop on RFID Security 2007 (RFIDSec 2007), Malaga, Spain, July 11-13, pp. 103–114 (2007)
19. Microchip Technology Inc. AN953: Data Encryption Routines for PIC18 Microcontrollers (January 2005), http://ww1.microchip.com/downloads/en/AppNotes/00953a.pdf

20. Nadehara, K., Ikekawa, M., Kuroda, I.: Extended Instructions for the AES Cryptography and their Efficient Implementation. In: IEEE Workshop on Signal Processing Systems (SIPS 2004), Austin, Texas, USA, pp. 152–157. IEEE Press (October 2004)
21. Needham, R.M., Wheeler, D.J.: Tea extensions. Technical report, Computer Laboratory, University of Cambridge (October 1997)
22. O'Melia, S., Elbirt, A.: Instruction Set Extensions for Enhancing the Performance of Symmetric-Key Cryptography. In: Annual Computer Security Applications Conference, ACSAC 2008, pp. 465–474 (December 2008)
23. Plos, T., Feldhofer, M.: Hardware Implementation of a Flexible Tag Platform for Passive RFID Devices. In: Proceedings of the 14th Euromicro Conference on Digital System Design Architectures, Methods and Tools (DSD 2011), Oulu, Finland, pp. 293–300. IEEE Computer Society (August 2011) ISBN 978-1-4577-1048-3
24. Plos, T., Groß, H., Feldhofer, M.: Implementation of Symmetric Algorithms on a Synthesizable 8-Bit Microcontroller Targeting Passive RFID Tags. In: Biryukov, A., Gong, G., Stinson, D.R. (eds.) SAC 2010. LNCS, vol. 6544, pp. 114–129. Springer, Heidelberg (2011)
25. Poschmann, A.Y.: Lightweight Cryptography - Cryptographic Engineering for a Pervasive World. PhD thesis, Faculty of Electrical Engineering and Information Technology, Ruhr-University Bochum, Germany (February 2009)
26. Rinne, S., Eisenbarth, T., Paar, C.: Performance Analysis of Contemporary Light-Weight Block Ciphers on 8-bit Microcontrollers (June 2007), http://www.crypto.ruhr-uni-bochum.de/imperia/md/content/texte/publications/conferences/lw_speed2007.pdf
27. Rolfes, C., Poschmann, A., Leander, G., Paar, C.: Ultra-Lightweight Implementations for Smart Devices – Security for 1000 Gate Equivalents. In: Grimaud, G., Standaert, F.-X. (eds.) CARDIS 2008. LNCS, vol. 5189, pp. 89–103. Springer, Heidelberg (2008)
28. Standaert, F.-X., Piret, G., Gershenfeld, N., Quisquater, J.-J.: SEA: A Scalable Encryption Algorithm for Small Embedded Applications. In: Domingo-Ferrer, J., Posegga, J., Schreckling, D. (eds.) CARDIS 2006. LNCS, vol. 3928, pp. 222–236. Springer, Heidelberg (2006)
29. Tillich, S., Großschädl, J.: Instruction Set Extensions for Efficient AES Implementation on 32-bit Processors. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 270–284. Springer, Heidelberg (2006)
30. Tillich, S., Herbst, C.: Boosting AES Performance on a Tiny Processor Core. In: Malkin, T. (ed.) CT-RSA 2008. LNCS, vol. 4964, pp. 170–186. Springer, Heidelberg (2008)
31. Wheeler, D.J., Needham, R.M.: TEA, a Tiny Encryption Algorithm.. In: Preneel, B. (ed.) FSE 1994. LNCS, vol. 1008, pp. 363–366. Springer, Heidelberg (1995)
32. Yan, H., Jianyun, H., Qiang, L., Hao, M.: Design of low-power baseband-processor for RFID tag. In: Proceedings International Symposium on Applications and the Internet Workshops (SAINT 2006), Phoenix, Arizona, USA, January 23-27, pp. 4–7. IEEE Computer Society (2006)
33. Yap, H., Khoo, K., Poschmann, A., Henricksen, M.: EPCBC - A Block Cipher Suitable for Electronic Product Code Encryption. In: Lin, D., Tsudik, G., Wang, X. (eds.) CANS 2011. LNCS, vol. 7092, pp. 76–97. Springer, Heidelberg (2011)
34. Yu, Y., Yang, Y., Yan, N., Min, H.: A Novel Design of Secure RFID Tag Baseband. In: RFID Convocation, Brussels, Belgium, March 14 (2007)

DRV-Fingerprinting: Using Data Retention Voltage of SRAM Cells for Chip Identification

Daniel E. Holcomb[1], Amir Rahmati[2], Mastooreh Salajegheh[2], Wayne P. Burleson[2], and Kevin Fu[2]

[1] UC Berkeley
[2] UMass Amherst

Abstract. Physical unclonable functions (PUFs) produce outputs that are a function of minute random physical variations. Promoted for low-cost authentication and resistance to counterfeiting, many varieties of PUFs have been used to enhance the security and privacy of RFID tags. To different extents, applications for both identification and authentication require a PUF to produce a consistent output over time. As the sensing of minute variations is a fundamentally noisy process, much effort is spent on error correction of PUF outputs. We propose a new variant of PUF that uses well-understood properties of common memory cells as a fingerprint. Our method of fingerprinting SRAM cells by their data retention voltage improves the success rate of identification by 28% over fingerprints based on power-up state.

1 Introduction

RFID circuits can be identified or authenticated using static identifiers stored in non-volatile memory or through the use of identifying physical characteristics. Physical characteristics have several security advantages over static identifiers, including immutability and resistance to cloning and tampering. The physical characteristics can be viewed as an identifying fingerprint of a given device. More formally, physical fingerprints are a component of a particular type of physical unclonable function (PUF) that is originally described as a physically obfuscated key [4], and more recently as a weak PUF [6].

If used for identification or constructing secret keys, fingerprint observations must be consistent over time. Sensing the microscopic variations that make each device unique while also minimizing the impact of noise is a fundamental concern in PUFs. Much effort is spent on error correction of somewhat-unreliable fingerprints or PUF outputs. Error correcting codes are expensive in terms of the number of raw bits required to create a reliable key, and more so if the number of correctable errors must be large. Toward this goal, we present a new fingerprinting method that is more reliable across trials than comparable previous approaches.

In this work we propose a new method for chip fingerprinting that uses Data Retention Voltage (DRV) in SRAM as the identifier. The DRV of an SRAM is the minimum voltage at which its cells can retain state. DRV fingerprints are

J.-H. Hoepman and I. Verbauwhede (Eds.): RFIDSec 2012, LNCS 7739, pp. 165–179, 2013.

found to be more informative than other approaches for fingerprinting SRAM that have been proposed in research [6,8] and commercially.[1] The physical characteristics responsible for DRV are imparted randomly during manufacturing and therefore serve as a natural barrier against counterfeiting. The proposed technique has the potential for wide application, as SRAM cells are among the most common building blocks of nearly all digital systems including smart cards and programmable RFID tags.

The contributions of this work are as follows:

- Demonstrating that the DRVs of SRAM cells are consistent fingerprints capable of identifying devices among a population.
- Demonstrating that DRV fingerprints make use of physical variations in a way that is similar to SRAM power-up fingerprints, but that DRV fingerprints have the potential for more accurate identification.

The remainder of this paper is structured as follows: Section 2 introduces data retention voltage. Section 3 explains how the DRVs of SRAM cells are characterized. Section 4 evaluates DRV fingerprinting using experimental data. Sections 5 and 6 review related work and present directions for future work.

2 Data Retention Voltage

A data retention failure is said to occur when an SRAM cell spuriously flips state due to insufficient supply voltage. The data retention voltage (DRV) of an SRAM array signifies the minimum supply voltage at which all SRAM cells can store arbitrary state. DRV is studied in the literature as a limit to supply voltage scaling. Various simulation models [2,12,25] and silicon measurements [15] show modern SRAM DRVs to be under 300mV. Most previous literature focuses on cases where the supply voltage of the circuit remains safely above DRV. While remaining above DRV, the supply voltage can be adjusted to reduce leakage power [3], compensate for manufacturing variability [12], or compensate for environmental variations [25].

Each SRAM cell uses the positive feedback of cross-coupled inverters to hold state on two complementary storage nodes. Retention failures occur at low supply voltages because the low voltage weakens the positive feedback of the cross-coupled inverters. Due to asymmetric process variation, at some low supply voltages a transition from a written state to the opposite state becomes inevitable; observations about the direction of such transitions and the voltages at which they occur are the basis for DRV fingerprints. Any collection of SRAM cells has a distinctive DRV fingerprint because of its unique random process variation.

3 Characterizing the DRV of an SRAM Cell

The DRVs of SRAM cells are characterized by repeatedly lowering the SRAM supply voltage and observing the highest voltage at which each cell fails. If the

[1] http://www.intrinsic-id.com/

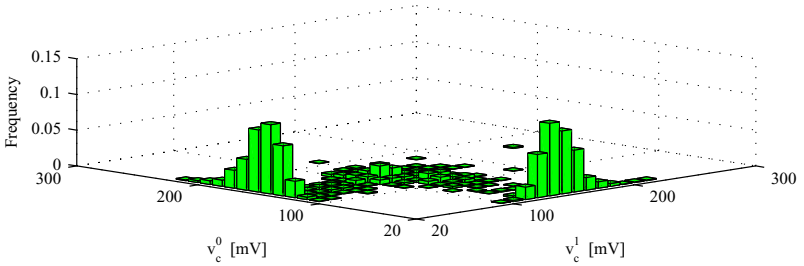


Fig. 1. The joint probability distribution function over all cells of the two variables (v_c^0 and v_c^1) comprising a DRV characterization. The distribution is determined experimentally using Algorithm 1, and shows that a large fraction of cells have the minimum possible value of 20mV for either v^0 or v^1, but none have the minimum value (or near-minimum values) for both. A cell with a minimum value for v^0 or v^1 is a cell that retains one written state across all test voltages.

SRAM supply node also supplies the processing core, then the low voltages used for the characterization will cause the core to reset and lose its state. Our experiments avoid this difficulty by using non-volatile memory to maintain persistency across the low voltages. However, a custom integrated circuit designed for DRV fingerprinting can also avoid this difficulty by using an SRAM supply node that is decoupled from the nominal supply node of the processor. This is often done, for example, in power-gated circuits where unused on-chip functional blocks are turned off entirely while the chip as a whole remains powered.

We characterize the DRV of an SRAM cell c with a pair $\langle v_c^0, v_c^1 \rangle$. Each v_c^w in the pair represents the highest voltage at which cell c will have a retention failure after state w is written to it. In principle, v_c^0 and v_c^1 are real-valued; but in practice, we approximate each using one of $N = (300mV - 20mV)/\Delta$ discrete values as shown in Algorithm 1. With Δ set at 10mV, the $N = 28$ possible values for v_c^0 and v_c^1 are $\{20mV, 30mV, \ldots, 290mV\}$. The frequency of observing different DRV pairs is shown in the joint probability distribution function of variables v_c^0 and v_c^1 in Fig. 1.

3.1 Experimental Setup

We examine the DRV of SRAM cells using Algorithm 1 implemented as follows: A microcontroller runs a program that sets all available memory bits to either 1 or 0. The supply voltage is then decreased to a value between 300mV and 20mV ($\Delta = 10mV$) for 5 seconds. When supply voltage is restored to 3V, the program stores the content of SRAM to the flash memory. Note that we conservatively use $t_{wait} = 5s$ to avoid missing marginal failures. Simulations by Nourivand et al. [12] using a procedure similar to Algorithm 1 show that waiting for $t_{wait} = 2ms$ at a reduced supply voltage is sufficient to observe retention failures. An Agilent U2541A-series data acquisition (DAQ) unit controls the supply voltage and the timing of when voltage is raised and lowered. Thermal tests are

Algorithm 1. Characterize the DRV fingerprint of a set of SRAM cells.

Prerequisite: C – a set of SRAM cells
Ensure: v_c^0, v_c^1 – the DRV characterizations of each SRAM cell $c \in C$.

Let V_{nom} be the nominal supply voltage (V_{dd}) for the chip
Let s_c refer to the logical state of SRAM cell $c \in C$.
Let s'_c refer to the logical state of NVM cell that corresponds to SRAM cell c.
for $w = 0, 1$ **do**
  for $c \in C$ **do**
    $s_c \leftarrow w$ {write w into SRAM cell}
    $s'_c \leftarrow w$ {write w into NVM cell}
    $v_c^w \leftarrow 0$ {value used if no retention failure observed}
  end for
  $v_{test} \leftarrow 300mV$ {initialize test voltage}
  while $v_{test} > 20mV$ **do**
    lower chip voltage from V_{nom} to v_{test}
    wait for t_{wait} seconds
    raise chip voltage from v_{test} to V_{nom}
    for $c \in C$ **do**
      if $(s_c = \neg w) \wedge (s'_c = w)$ **then**
        SRAM cell c had a retention failure from state w at voltage v_{test}, but previously had no failure at voltage $v_{test}+\Delta$. Therefore v_{test} approximates the largest voltage that induces a retention failure after writing w.
        $v_c^w \leftarrow v_{test}$
      end if
      $s'_c \leftarrow s_c$ {write SRAM to NVM}
    end for
    $v_{test} \leftarrow v_{test} - \Delta$ {try a lower voltage next}
  end while
end for

conducted inside of a Sun Electronics EC12 Environmental Chamber [22], and an OSXL450 infrared non-contact thermometer [13] with $\pm 2°C$ accuracy is used to verify the temperature. All experiments use instances of Texas Instruments MSP430 F2131 microcontrollers with 256 bytes of SRAM, of which 240 bytes are available for DRV fingerprinting. The DRV of each cell is characterized 20 times. The total runtime to characterize all 240 bytes of SRAM on a chip once using Algorithm 1 is given by t_{proc} in Eq. 1, and is 140 seconds for the conservative case of $\Delta = 10mV$ and $t_{wait} = 5s$.

$$t_{proc} = t_{wait} \times \frac{300mV - 20mV}{\Delta} \tag{1}$$

3.2 Information Content of SRAM Cell DRV

The DRV of each cell has N^2 possible outcomes representing all combinations of N outcomes for v_c^0 and the N outcomes for v_c^1 (in our case $N = 28$). The DRV of each cell is then a random variable X with N^2 outcomes denoted x_0 through

x_{N^2-1}. The total entropy $H(X)$ is the expected information value of the DRV of an unknown cell. Entropy depends (per Eq. 2) on the probabilities of each DRV outcome, denoted $p(x_i)$. In the ideal case where all N^2 outcomes are equally likely (e.g. $p(x_i) = 1/N^2$ for all x_i), each DRV would have almost 10 bits of entropy. Applying Eq. 2 to the decidedly non-uniform outcome probabilities of Fig. 1 shows the actual entropy of a DRV to be 5.12 bits. The most frequently observed DRV outcomes are given in Table. 1.

Eq. 1 shows that runtime is inversely proportional to Δ, so we consider the information loss from making Δ larger than 10mV. Fig. 2 shows the ideal and actual entropy of DRV characterizations when different values of Δ are used. In the extreme case where $\Delta = 140mV$, variables v_c^0 and v_c^1 are each restricted to the values $\{20mV, 160mV\}$, so the ideal entropy of the DRV is equivalent to 2 flips of a fair coin. The values of Δ used in Fig. 2 are chosen on account of being unambiguously recreatable from the $\Delta = 10mV$ data.

$$H(X) = -\sum_{i=1} p(x_i) \log p(x_i) \tag{2}$$

Table 1. The 4 most commonly observed weak and strong DRV characterizations, and the probability of observing each in a randomly selected trial

(a) Most common weak DRVs

Outcome $\langle v_c^0$, $v_c^1\rangle$	Freq.
$\langle 130mV$, $100mV\rangle$	0.0096
$\langle 120mV$, $100mV\rangle$	0.0076
$\langle 130mV$, $110mV\rangle$	0.0070
$\langle 120mV$, $110mV\rangle$	0.0070

(b) Most common strong DRVs

Outcome $\langle v_c^0$, $v_c^1\rangle$	Freq.
$\langle 20mV$, $130mV\rangle$	0.0893
$\langle 20mV$, $120mV\rangle$	0.0719
$\langle 130mV$, $20mV\rangle$	0.0685
$\langle 20mV$, $140mV\rangle$	0.0651

3.3 Observations about Strong and Weak Cells

We abstract the N^2 possible DRV characterizations (Fig. 1) into three classes[2] that are sufficient to demonstrate general observations about all DRVs:

- A *strongly 0* DRV characterization is a pair $\langle v_c^0, v_c^1\rangle$ such that $v_c^0 = 20mV$ and $v_c^1 > 20mV$. A strongly 0 DRV indicates that no retention failure occurs at any voltage v_{test} after state 0 is written.
- A *strongly 1* DRV characterization is a pair $\langle v_c^0, v_c^1\rangle$ such that $v_c^0 > 20mV$ and $v_c^1 = 20mV$. A strongly 1 DRV indicates that no retention failure occurs at any voltage v_{test} after state 1 is written.
- A *weak* DRV characterization is a pair $\langle v_c^0, v_c^1\rangle$ such that $v_c^0 > 20mV$ and $v_c^1 > 20mV$. A weak DRV indicates that a failure is observed at some voltage v_{test} after each state is written.

[2] Note that no observation of $\langle v_c^0, v_v^1\rangle = \langle 20mV, 20mV\rangle$ is ever made, so we do not include this outcome in any of the three cases.

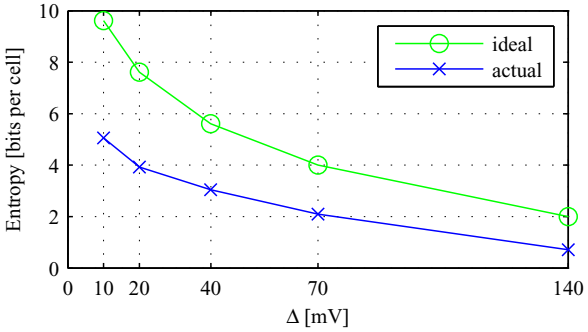


Fig. 2. Sweeping Δ from 10mV to 140mV shows that a loss of measurement precision reduces entropy of each cell's DRV characterization

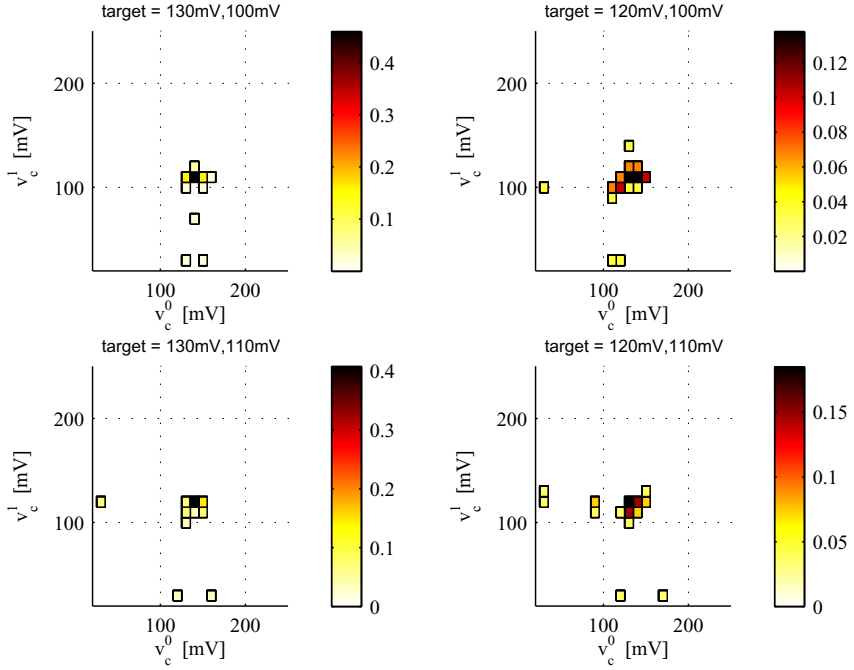


Fig. 3. For each of the 4 most frequently observed weak DRVs (see Table 1a), the DRV in a second trial from a cell that produced the frequently observed DRV in a first trial

The variation-dependent behavior of an SRAM cell occurs somewhere between 20mV and 300mV for each cell; above 300mV all cells can reliably hold either the 0 or the 1 state, and below 20mV no cells can do so. When a cell produces a strongly 0 or strongly 1 characterization, it means (per Algorithm 1) that for one written state the supply voltage is lowered all the way through the sensitive region down to 20mV and then raised back up without causing a failure.

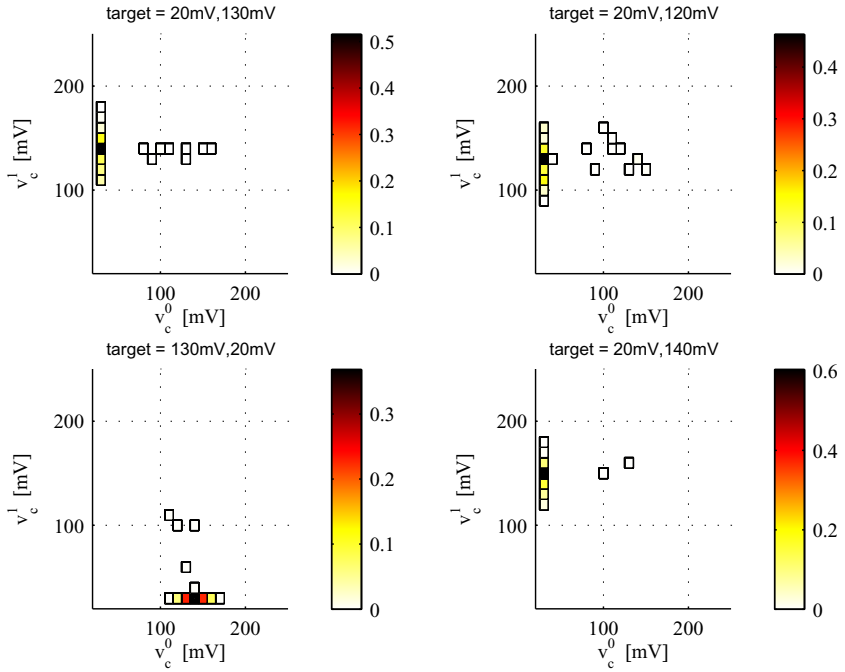


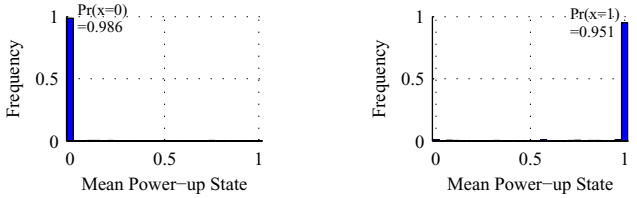
Fig. 4. For each of the 4 most frequently observed strong DRVs (see Table. 1b), the DRV in a second trial from a cell that produced the frequently observed DRV in a first trial

A strongly 0 or strongly 1 characterization therefore indicates a strong preference for one state over the other at all supply voltages. A weak characterization is when each written state flips at some voltage within the sensitive region, and neither state can be retained down to 20mV.

Both strong and weak DRV characterizations are largely repeatable across trials. Fig. 3 shows the distribution of DRVs produced by randomly selected cells for which the first DRV produced is one of the 4 most commonly observed weak DRVs from Table 1a; each plot shows the conditional probability distribution of a subsequent DRV characterization. Occasionally the same cells that produce a weak DRV produce a strong DRV in subsequent trials. Fig. 4 shows the same analysis for the 4 most commonly observed strong DRVs; none of the cells subsequently produces the opposite strong characterization.

3.4 Relation to Power-Up State

It is known that SRAM cells consistently power-up to the same state [6,8] in a majority of trials. Cells with highly reliable power-up states tend to be the same cells with strong DRV characterizations. Fig. 5 shows the mean power-up state over 28 trials for cells that produced a strongly 0 or strongly 1 DRV



(a) Strongly 0 DRV **(b)** Strongly 1 DRV

Fig. 5. The plot at left shows that 98.6% of SRAM cells that produce a strongly 0 DRV reliably power-up to state 0, as observed by a mean power-up state of 0. The plot at right shows that 95.1% of cells with strongly 1 DRVs reliably power-up to state 1. The DRV is from a single trial of the cell, and the mean power-up state is measured over 28 power-up trials.

Table 2. Probability of different pairwise outcomes when 2 DRV fingerprints are taken from a randomly chosen cell. Over the 5000 samples collected, no cell ever has a DRV that is strongly 1 in one trial and strongly 0 in another, but 5.6% of outcomes have one strong and one weak DRV.

	Strongly 0	**Weak**	**Strongly 1**
Strongly 0	35.80%	3.10%	0.00%
Weak	-	24.98%	2.48%
Strongly 1	-	-	33.64%

characterization. Among cells with strongly 0 DRV, 98.6% power-up to the 0 state in all 28 power-up trials (Fig. 5a). Similarly, 95.1% of cells characterized as strongly 1 consistently power-up to the 1 state (Fig. 5a). Although a strong DRV fingerprint is correlated to power-up tendency, the DRV provides a more informative identifier than does power-up by providing information about the maximum voltage at which the unfavored state cannot be reliably stored.

4 Fingerprint Matching

A DRV fingerprint is obtained from a single characterization of a set of adjacent cells within an SRAM. A k-bit fingerprint F_i comprises cell characterizations $\langle v_i^0, v_i^1\rangle, \langle v_{i+1}^0, v_{i+1}^1\rangle, \ldots, \langle v_{i+k-1}^0, v_{i+k-1}^1\rangle$. The difference between fingerprints is the sum of the differences between their corresponding single-cell characterizations. Recalling that each DRV is a point $\langle v_c^0, v_c^1\rangle$ in 2-dimensional space, we define the distance between two DRVs according to the square of their distance along each dimension (Eq. 3). For comparison, a second metric used is the Hamming distance between power-up trials; this is shown by Eq. 4, where p_i is the state of the i^{th} bit of SRAM after a power-up.

$$d1(F_i, F_j) = \sum_{n=0}^{k-1} \left(v_{i+n}^0 - v_{j+n}^0\right)^2 + \left(v_{i+n}^1 - v_{j+n}^1\right)^2 \tag{3}$$

$$hd(F_i, F_j) = \sum_{n=0}^{k-1} p_{i+n} \oplus p_{j+n} \tag{4}$$

4.1 Identification at Nominal Temperature

At the nominal operating temperature of $29°C$, three experiments compare DRV fingerprints with power-up fingerprints. These experiments are explained in the following subsections; the first shows the histograms of distances between fingerprints, and the second and third evaluate the accuracy of distance-based matching.

4.1.1 Histogram of Distances between Fingerprints

A first experiment shows that DRV fingerprints are repeatable and unique, as is necessary for successfully identifying chips within a population. Within-class pairings are of multiple fingerprints generated by the same set of cells on the same device. Between-class pairings are from different sets of cells on the same device, or from any sets of cells on different devices. The similarity of any two fingerprints is quantified by a distance, and this distance is the basis for determining the correct identity of a fingerprint. If within-class fingerprint pairings consistently have smaller distances than between-class pairings, then it is possible to determine identity by choosing an appropriate threshold that separates the two classes. The histograms of within-class and between-class distances for DRV and power-up fingerprints are shown in Fig. 6. These histograms represent all data collected from the MSP430F2131 microcontrollers at room temperature. The distances on the x-axes are not directly comparable across metrics; of importance is only whether the two classes are clearly separable within each plot.

4.1.2 Accuracy of Top Match

The next experiment performed at nominal temperature evaluates how reliably a single within-class DRV fingerprint can be identified among a population. This experiment matches a single 16-bit target fingerprint against a population containing another fingerprint from the same cells and one fingerprint from each of the 239 remaining locations across 2 chips. A positive result occurs if the closest match among the 240 possibilities is from the same SRAM cells as the target. The results of the top match experiment are shown in Table 3; the column labelled "co-top" shows the percentage of trials where there are multiple top matches and one of them correctly matches the target. Multiple top matches are relatively common in Hamming distance matching due to the small number of possible distances between fingerprints. Compared to power-up fingerprints, matching based on DRV fingerprints is 28% more likely to have the correct match be closer to the target (i.e. separated by a smaller distance) than all incorrect matches.

Table 3. Over 300 trials with a population of 240 16-bit fingerprints, DRV identification returns the fingerprint that correctly matches the target more reliably than power-up state identification. Matching based on power-up state more frequently returns a misidentified fingerprint, or returns multiple fingerprints among which one is the correct match (denoted "co-top").

	top	**co-top**	**misidentified**
DRV (d1)	99.7%	-	0.3%
Power-up	71.7%	24.7%	3.6%

4.1.3 Precision and Recall

The top match experiment is generalized to the case of identifying multiple correct matches among a larger population, and again shows DRV fingerprints to outperform power-up fingerprints. In this experiment, our goal is to find all correct matches in the population, without also finding too many incorrect matches. In doing so, the distance that is considered to be the threshold between a correct and incorrect match can be adjusted. If the threshold is too low then correct matches may not be identified, but if the threshold is too high then false positives will occur. *Recall* refers to the fraction of within-class pairings under the threshold, and *precision* refers to the fraction of pairings under the threshold that are within-class. Increasing the threshold will sacrifice precision for recall, and decreasing the threshold will sacrifice recall for precision. An ideal result is for both precision and recall to be 1; this result occurs if all correct matches are identified as within-class (perfect recall) with no incorrect ones identified as within-class (perfect precision).

The precision and recall plots of Fig. 7 are obtained by iterating the following procedure. One 16-bit segment of SRAM is chosen for identification. One fingerprint trial from this segment is chosen at random as the target, and it is matched against a population of 1019 fingerprints comprising 19 from the same SRAM segment (within-class pairings) and 1000 non-matching fingerprints (between-class pairings). The non-matching fingerprints are randomly selected among 20 trials from 239 other segments of SRAM[3]. The matching threshold is swept to find achievable precision-versus-recall tradeoffs, and each achievable tradeoff is a point in Fig. 7. The large number of tradeoff points in the plot is collected from multiple iterations of this procedure. The general trend is that DRV fingerprints produce better recall for a given precision, or better precision for a given recall compared to power-up fingerprints.

4.2 Impact of Temperature Variations

Given that DRV fingerprints would likely be used in real-world scenarios without precisely-controlled temperatures, a final experiment explores the impact of

[3] The 239 eligible 16-bit segments are the 119 remaining on the target's own chip, and all 120 such locations on the other device.

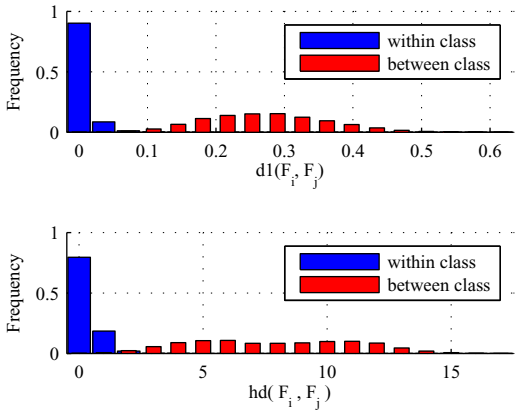


Fig. 6. Within-class and between-class distances of 16-bit fingerprints. The upper plot uses DRV fingerprints with distance metric $d1$ from Eq. 3. The lower plot uses power-up fingerprints with Hamming distance as a metric.

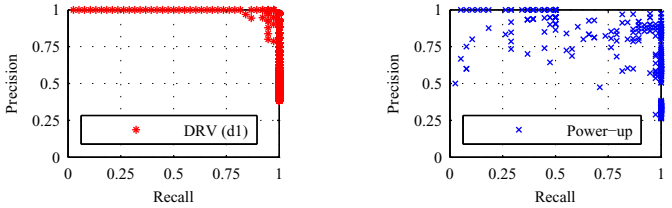


Fig. 7. Tradeoff points of precision and recall for trials of DRV fingerprints are generally closer to the ideal result of perfect precision and recall

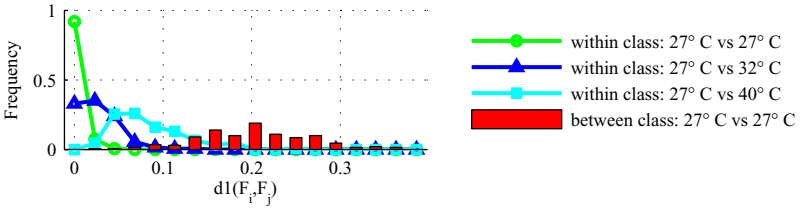


Fig. 8. The line plots show within-class distances when one fingerprint observation is made at $27°C$ and the second at $27°C$, $32°C$, or $40°C$; within-class distances increase with temperature, implying a diminished reliability. The bar plot shows between-class distances of 16-bit fingerprints taken at $27°C$. Because there does not exist a distance threshold that can separate the two classes when temperature is varied, it may be necessary to use larger fingerprints for reliable identification.

temperature on DRV fingerprints. This experiment is similar to the experiment of subsection 4.1.1, but the pairs of fingerprint observations used to generate the within-class distances are now made at different temperatures. The results are shown in Fig. 8. The increase of within-class distances across temperature implies a diminished reliability. To compensate for this, larger fingerprints (comprising more bits) may be needed for identification, and more robust error correcting codes may be needed in key-generation applications. If the increased within-class distances are due to a uniform shift in the DRVs of all cells, then a promising direction for future work would be to design a matching scheme that is insensitive to this type of uniform shift.

5 Related Works

A wide variety of PUFs and fingerprints based on custom or pre-existing integrated circuit components have been developed. The identifying features used by custom designs include MOSFET drain-current [10], timing race conditions [5], and the digital state taken by cross-coupled logic after a reset [20]. IC identification based on pre-existing circuitry is demonstrated using SRAM power-up state [6, 8], and physical variations of flash memory [14]. Lee et al. [9] derive a secret key unique to each IC using the statistical delay variations of wires and transistors across ICs. Bhargava et al. explore circuit-level techniques for increasing the reliability of SRAM PUFs [1]. An experimental evaluation of low-temperature data remanence on a variety of SRAMs is provided by Skorobogatov [19], and SRAM remanence in RFID has been studied by Saxena and Voris as a limitation to SRAM-based true random number generation [18].

Previous works [17, 23] have used error correction to construct secret keys from noisy PUF sources; however, this is expensive in terms of gates and other resources. To give an idea of the cost of error correction, BCH codes previously used with PUFs include one to correct 21 errors among 127 raw bits in creating a 64-bit key [21], and to correct 102 errors among 1023 raw bits in creating a 278-bit key [6]. The work of Guajardo et al. [6] uses a derivative of power-up SRAM state as a secret key; however, it requires an error correction code and imposes SRAM space overhead. Maes et al. [11] introduce an SRAM helper data algorithm to mask unreliable bits using low-overhead post-processing algorithms. Recently, Yu et al. [26] proposed a method of error correction for PUFs using a new syndrome coding scheme to minimize the information leaked by the error correction codes, and Hiller et al. extend this approach for SRAM PUFs [7]. Van Herrewege et al. [24] have designed a new lightweight authentication scheme using PUFs that does not require the reader to store a large number of PUF challenge and response pairs.

Given the low cost of the several bytes of SRAM that are used for DRV fingerprinting, a relatively significant practical cost may be associated with the generation of the test voltages for characterizing the DRVs. Emerging devices such as computational RFIDs [16] can use software routines to extract DRVs, but as contactless devices they must generate all test voltages on-chip. On-chip

dynamic control of SRAM supply voltage is assumed in the low-power literature at least since work on drowsy caches [3]. Supply voltage tuning has also been applied with canary cells to detect potential SRAM failures, and as a post-silicon technique to compensate for process variation and increase manufacturing yields [12].

6 Conclusions and Future Works

This work has demonstrated that SRAM DRV fingerprints are static identifiers of a device, and it has presented a simple characterization procedure and matching algorithms to use them as such. DRV fingerprints are similar to previously demonstrated power-up fingerprints, but they provide a more informative non-binary identifier of each cell. As a result of this, DRV fingerprints are identified up to 28% more reliably than are power-up fingerprints.

The practical limits of DRV fingerprint performance and reliability should be explored further. Within the constraints of acceptable precision, the runtime of the characterization procedure can be reduced by increasing the voltage step size Δ and reducing the time t_{wait} spent at each voltage (Eq. 1). An expanded evaluation could investigate the reliability of DRV fingerprints across a larger variety of devices and a range of environmental conditions. A high reliability could make DRV fingerprints suitable as a basis for key-generation with lightweight error correcting codes.

Acknowledgments. This research is supported by NSF grants CNS-0964641, CNS-0923313, CNS-0845874, and SRC task 1836.074. Additionally, this research was supported in part by the Gigascale Systems Research Center, one of six research centers funded under the Focus Center Research Program (FCRP), a Semiconductor Research Corporation entity. Any opinions, findings, and conclusions or recommendations expressed in these materials are those of the authors and do not necessarily reflect the views of the sponsors.

References

[1] Bhargava, M., Cakir, C., Mai, K.: Reliability enhancement of bi-stable PUFs in 65nm bulk CMOS. In: International Symposium on Hardware-Oriented Security and Trust (2012)
[2] Cabe, A.C., Qi, Z., Stan, M.R.: Stacking SRAM banks for ultra low power standby mode operation. In: Design Automation Conference (June 2010)
[3] Flautner, K., Kim, N., Martin, S.: Drowsy caches: simple techniques for reducing leakage power. In: International Symposium on Computer Architecture (2002)
[4] Gassend, B.: Physical Random Functions. Master's thesis. MIT, USA (2003)
[5] Gassend, B., Clarke, D., Van Dijk, M.: Silicon physical random functions. In: Proceedings of the IEEE Computer and Communications Society (2002)
[6] Guajardo, J., Kumar, S.S., Schrijen, G.-J., Tuyls, P.: FPGA Intrinsic PUFs and Their Use for IP Protection. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, vol. 4727, pp. 63–80. Springer, Heidelberg (2007)

[7] Hiller, M., Merli, D., Stumpf, F., Sigl, G.: Complementary IBS: Application specific error correction for PUFs. In: International Symposium on Hardware-Oriented Security and Trust (2012)
[8] Holcomb, D.E., Burleson, W.P., Fu, K.: Power-up SRAM state as an identifying fingerprint and source of true random numbers. IEEE Transactions on Computers (2009)
[9] Lee, J., Lim, D., Gassend, B., Suh, G., van Dijk, M., Devadas, S.: A technique to build a secret key in integrated circuits for identification and authentication applications. In: 2004 Symposium on VLSI Circuits. Digest of Technical Papers, pp. 176–179 (June 2004)
[10] Lofstrom, K., Daasch, W., Taylor, D.: IC identification circuit using device mismatch. In: IEEE International Solid-State Circuits Conference. Digest of Technical Papers, pp. 372–373 (2000)
[11] Maes, R., Tuyls, P., Verbauwhede, I.: Low-Overhead Implementation of a Soft Decision Helper Data Algorithm for SRAM PUFs. In: Clavier, C., Gaj, K. (eds.) CHES 2009. LNCS, vol. 5747, pp. 332–347. Springer, Heidelberg (2009)
[12] Nourivand, A., Al-Khalili, A.J., Savaria, Y.: Postsilicon Tuning of Standby Supply Voltage in SRAMs to Reduce Yield Losses Due to Parametric Data-Retention Failures. IEEE Transactions on Very Large Scale Integration (VLSI) Systems 1, 29–41 (2011)
[13] Omega Engineering, I. OSXL450 Infrared Non-Contact Thermometer Manual
[14] Prabhu, P., Akel, A., Grupp, L.M., Yu, W.-K.S., Edward Suh, G., Kan, E., Swanson, S.: Extracting Device Fingerprints from Flash Memory by Exploiting Physical Variations. In: McCune, J.M., Balacheff, B., Perrig, A., Sadeghi, A.-R., Sasse, A., Beres, Y. (eds.) Trust 2011. LNCS, vol. 6740, pp. 188–201. Springer, Heidelberg (2011)
[15] Qin, H., Cao, Y., Markovic, D., Vladimirescu, A., Rabaey, J.: SRAM leakage suppression by minimizing standby supply voltage. In: 5th International Symposium on Quality Electronic Design, pp. 55–60 (2004)
[16] Ransford, B., Clark, S., Salajegheh, M., Fu, K.: Getting things done on computational RFIDs with energy-aware checkpointing and voltage-aware scheduling. In: USENIX Workshop on Power Aware Computing and Systems (HotPower) (December 2008)
[17] Sadeghi, A.-R., Visconti, I., Wachsmann, C.: Enhancing RFID Security and Privacy by Physically Unclonable Functions. In: Information Security and Cryptography, pp. 281–307. Springer (September 2010)
[18] Saxena, N., Voris, J.: We can remember it for you wholesale: Implications of data remanence on the use of RAM for true random number generation on RFID tags. In: Proceedings of the Conference on RFID Security (2009)
[19] Skorobogatov, S.: Low temperature data remanence in static RAM. Tech. Rep. UCAM-CL-TR-536, University of Cambridge Computer Laboratory (2002)
[20] Su, Y., Holleman, J., Otis, B.: A digital 1.6 pj/bit chip identification circuit using process variations. IEEE Journal of Solid-State Circuits 43(1), 69–77 (2008)
[21] Suh, G., O'Donnell, C., Devadas, S.: AEGIS: a single-chip secure processor. IEEE Design & Test of Computers 24(6), 570–580 (2007)
[22] Sun Electronic Systems, I. Model EC1X Environmental Chamber User and Repair Manual (2011)
[23] Tuyls, P., Batina, L.: RFID-Tags for Anti-counterfeiting. In: Pointcheval, D. (ed.) CT-RSA 2006. LNCS, vol. 3860, pp. 115–131. Springer, Heidelberg (2006)

[24] Van Herrewege, A., Katzenbeisser, S., Maes, R., Peeters, R., Sadeghi, A.-R., Verbauwhede, I., Wachsmann, C.: Reverse Fuzzy Extractors: Enabling Lightweight Mutual Authentication for PUF-Enabled RFIDs. In: Keromytis, A.D. (ed.) FC 2012. LNCS, vol. 7397, pp. 374–389. Springer, Heidelberg (2012)
[25] Wang, J., Calhoun, B.H.: Techniques to Extend Canary-Based Standby VDD Scaling for SRAMs to 45 nm and Beyond. IEEE Journal of Solid-State Circuits 43(11), 2514–2523 (2008)
[26] Yu, M.-D., Devadas, S.: Secure and Robust Error Correction for Physical Unclonable Functions. IEEE Design & Test of Computers 27(1), 48–65 (2010)

Author Index