matthew

D2.1 Report on Multiple Secure Elements

Project number:	ICT-610436
Project acronym:	**MATTHEW**
Project title:	Multi-entity-security using Active Transmission Technology for Improved Handling of Exportable Security Credentials without Privacy Restrictions
Project Start Date:	1 November, 2013
Duration:	36 months
Programme:	FP7-ICT-2013-10

Deliverable Type:	Preliminary Report
Reference Number:	ICT-610436-D2.1
Workpackage:	WP 2
Due Date:	25 September, 2015
Actual Submission Date:	25 September, 2015

Responsible Organisation:	CRX
Editor:	Pascal Paillier
Dissemination Level:	PU
Revision:	1.0

Abstract:	This document explores the secure transferability of credentials in the specific context of the three use cases demonstrated within the project.
Keywords:	Multiple roots of trust, secure elements, credentials, privacy, PUF, pairings

This project has received funding from the European Unions Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 610436.

Editor

Pascal Paillier(CRX)

Contributors (ordered according to beneficiary numbers)

Erich Wenger (IAIK)
Holger Bock (IFAT)
Martin Deutschmann, Michael Höberl (TEC)
Cécile Delerablée (CRX)

Executive Summary

At a high-level perspective, MATTHEW intends to enable new applications and services on mobile platforms using *multiple roots of trust*. In that respect, the privacy of users is considered as being a critical asset and therefore MATTHEW considers this problem from the design phase on. Dealing with multiple roots of trust enables the potentiality of *transfers of credentials* from one secure element to another. Of course, the meaning of the term *credential* may differ depending on the considered application. Therefore operating multiple roots of trust opens the way to a notion of mobility for credentials, which may be stored and used within different secure elements as users interact with their devices in their daily life. This document explores the transferability of credentials in the specific context of the three use cases demonstrated within the project.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Relation of WP2/T2.1/D2.1 to MATTHEW goals

1.1.1 Key Goals of MATTHEW

At a high-level perspective, MATTHEW intends to enable new applications and services on mobile platforms using *multiple roots of trust*. In that respect, the privacy of users is considered as being a critical asset and therefore MATTHEW considers this problem from the design phase on. Beyond investigating privacy-enhancing technologies, the project aims at integrating them into the multiple roots of trust concept.

Most interestingly, dealing with multiple roots of trust enables the potentiality of *transfers of credentials* from one secure element to another. Of course, the meaning of the term *credential* may differ depending on the considered application. One may view a credential as some form of cryptographic object such as a key or a certificate provided by a third party, but irrespective of its form, the user experience attached to a credential always comes as the ability to perform some privileged action (payment, access, authentication, etc.). Therefore operating multiple roots of trust opens the way to a notion of mobility for credentials, which may be stored and used within different secure elements as users interact with their devices in their daily life.

1.1.2 Task 2.1 - Multiple Secure Elements (M01-M24)

Given that credentials may be subject to transfers between secure elements, it is rather obvious that such transfers cannot be done in a naive way without raising security and/or privacy concerns. If the transfer of a credential ends up achieving a mere copy, one potentially faces a cloning threat. Simultaneously, transfer and subsequent use of the transferred credential should not reveal indiscreet information about the user *e.g.* some of her identity attributes. Task 2.1 intends to explore the issue of achieving a notion of secure transferability of credentials between secure elements across multiple devices, most essentially in the context of the three use cases demonstrated by the project. This notion of secure transferability of a credential is referred to as *multiple entity security* in this document.

1.1.3 Purpose of this Deliverable

This deliverable explores the nature of the "multiple entity security" in the above sense, of the three applications considered within MATTHEW.

1.2 Policy-enforcing Solutions

There are many approaches to solve the challenges derived from above. It is possible to use sophisticated cryptographic algorithms as well as simple organisational solutions to tackle these challenges. While the project partners are going to use cryptography as the main tool, it is important to keep the non-cryptographic solutions in mind.

One of the key challenges, *transferability*, can be achieved using some of the following techniques:

- The most straightforward solution is to have a device that can be moved using physical means from one mobile platform to another. In a mobile setting, *e.g.* a ticket can be bound to an SD-card and the SD-card can be **physically moved** from one mobile platform to another. The disadvantage of such a solution is the value an SD-card might hold (*e.g.* private pictures or private data).
- Tickets can also be represented as data (serial numbers or cryptographic credentials). The assumption might be that the data is safely stored on a "secure" device and the owners of two devices want to transfer a ticket. Computers cannot move data. Data can only be copied or overwritten with other data, a.k.a "deleted". From a security standpoint, neither the **copy-and-delete** nor the **delete-and-copy** approach are sufficient to both guarantee a safe transfer of the data, and a safe disposal of the original data.
- Even if the security goals are built upon a **closed** system (security by obscurity), operators are not safe from attacks. Examples like the reverse engineering of the Crypto1 algorithm used on MIFARE cards [11, 24] highly discourage a closed system. In **open** systems, everybody has the ability to check the security of the used cryptographic protocols, or even build a secure implementation themselves. Having replica devices will actually not jeopardise the security goals.

1.3 Structure of this Report

This report intends to cover the "multiple entity security" aspects of the three use cases considered within MATTHEW:

Use case 1 (advanced mobile banking application): the transfer of payment credentials from one device to another is covered by simply transferring the payment-enabled microSD card;

Use case 2 (advanced access control application): this use case does not support any transferability of credentials between secure elements. However, a SIM card can be moved from one device to another;

Use case 3 (advanced ticketing application): tickets can be transferred from one secure element to another one, either via an online connection, or using a local device-to-device connection such as the NFC channel.

Each use case is considered individually within this report. We finally give an outlook on future works in Chapter 6.

Chapter 2

Implementation Aspects

When the goal is to build a secure infrastructure by designing a cryptographic protocol, it is necessary to both consider the security goals (of cryptographic nature and of implementation-security nature) and the practical performance demands. Within MATTHEW's WP2 there are some implementation aspects within tasks 2.2-4 that have a direct influence on the designed algorithm.

2.1 From Traditional Cryptography to Pairings

Bilinear pairings have been used to design ingenious protocols for such tasks as one-round three-party key agreement, identity-based encryption, aggregate signatures, and an ever-growing corpus of other cryptographic mechanisms. In comparison with pre-pairing cryptography e.g. the design techniques that prevailed until about the year 2000, pairings have made it possible to devise simple and elegant solutions to long-standing design problems that traditionally seemed out of reach or impractical. Their introduction has been considerably beneficial to the field of public-key cryptography by enabling a great simplification of many schemes and protocols ranging from encryption to signatures with specific properties, from zero-knowledge proofs to compact e-cash, to name a few.
In our quest for a privacy-preserving ticketing system as per Use Case 3, the use of pairings through a system *la* BBS [4] has allowed us to conceive *compact wallets of tickets* that can be presented iteratively while relying essentially on the same constant-size information. Such efficiency is not known to be achievable using non-pairing cryptography. Thus, pairings play an essential role in our ticketing application.
Implementing pairings both efficiently and securely (in terms of counteracting physical attacks such as side-channels and fault injection) is far from just being an engineering problem but requires innovative solutions either in hardware or software. Therefore our research efforts within MATTHEW include pairing-specific hardware design to a great extent, and more generally a search for the right balance between hardware and software partitioning, as shown in further sections.

2.2 The Role of PUFs

A Physically Unclonable Function (PUF) can be described as a physical system which, when measured or challenged, provides unique, repeatable and unpredictable responses. Creating a physical copy of the PUF with an identical challenge-response behaviour is hard, thus resulting in a structure which is unclonable even by the manufacturer [18].

2.2.1 Introduction to SRAM PUFs

Silicon PUFs are a specific category of Physically Unclonable Functions which exploit the uncontrollable manufacturing in silicon which are a result of the IC fabrication process. SRAM PUFs are the most important representative of silicon PUFs. They utilize the unpredictable power up behaviour of uninitialized SRAM cells. When SRAM cells are powered up, their state either switches to "1" or to "0", depending only on a tiny imbalance of the threshold voltage level of two of the six transistors. The voltage imbalance comes from random doping concentrations in the silicon, which are out of the control of the manufacturer. Even though each cell's power-up state is randomly distributed over the cells of one SRAM block and over the cells of different devices, the same cell has a very high probability of powering up in the same state repeatedly over time [17]. If we view each SRAM cell as a single bit, the resulting bit string will tend to have the (nearly) same value every time the device is powered up and this way can serve as the response of a PUF as described above.

2.2.2 Introduction to RO PUF

The former introduced SRAM-PUF is part of the memory-based PUF group as it uses the power-up value of SRAM cells. The delay-based PUFs are another group, which relay on timing variances on a die. These variances are once again caused by uncontrollable variations during the manufacturing process of the chip, therefore each chip will have different timing characteristics. Based on this, the ring oscillator and Arbiter PUF are very prominent members of the delay-based PUF group.
The Arbiter PUF uses two equally routed nets on a die and applies at the same time a signal to them. An Arbiter circuit is used at the end of the two wires to elect the winner. As this race would only generate one bit, additional switches are placed within the *race track* to create various tracks. Thus, the response is not only limited to one bit. However, as the implementation of an Arbiter PUFs requires full control, which is impossible or at least very difficult on an FPGA, this PUF type is not further considered. The RO PUF uses frequency variations of ring oscillators to form a response. Therefore, it is necessary to implement an odd number of inverters and a feedback loop as displayed in Figure 2.1.

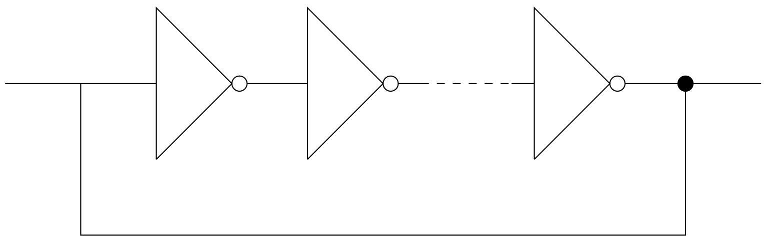

Figure 2.1: Ring Oscillator based on an odd Number of Inverters and a Feedback Loop.

This oscillator has to be evaluated for fixed time period. This is usually done by counting the high states of the oscillators output. The resulting counter value is afterwards compared against the counter value of another oscillators to form a binary result. Based on this, it is necessary to implement at least $2n$ ring oscillators to form a response with n independent bits.

2.2.3 PUF based Key Generation

A promising usage of PUFs is secure key generation in silicon, which eliminates the need for storing keys in NVM technologies. PUF based key generation provides advantages like physical unclonability and tamper evidence, compared to storing the keys in NVM.

Since PUF responses are inherently noisy and may not be uniformly random, a post-processing function is needed to condition the raw PUF responses into high-quality cryptographic keys. This post-processing function is known as the fuzzy extractor or helper data algorithm (HDA) in the literature. Several fuzzy extractor schemes suitable for different implementations have been proposed in the literature. Bösch et al. for example proposed the usage of concatenated codes, where the inner stage is a conventional error correcting code such as BCH, Reed-Muller, or Golay code, and the outer stage is a repetition code. This fuzzy extractor offers simple encoding/decoding and efficient hardware implementation [7]. We differentiate between three cases, how PUF can be deployed as key generators.

1. *Key derived from silicon*

   This is the basic application, that derives the key directly from the binary PUF response. No additional building blocks (e.g. key generation routine) are needed, the solution provides an out of the box cryptographic key. A typical usage is to use the derived key as a key for an symmetric encryption scheme as AES. The encrypted content can be only decrypted if the reconstruction of the key is successful.
   The essential question is the design of the HDA which influences the reliability of the key reconstruction and the length of the needed PUF response. Further it has to be made sure that the derived key has enough entropy. A common solution is the usage of hash functions for the entropy extraction. Figure 2.2 depicts the basic framework of the key generation.

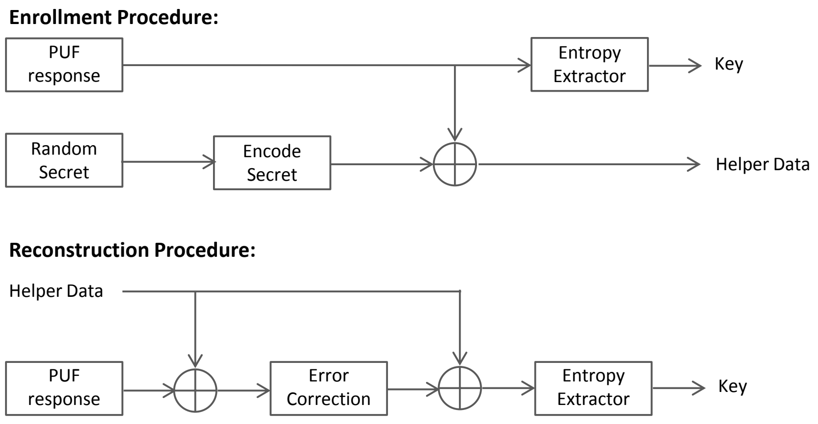

Figure 2.2: Derivation of a symmetric key

2. *Protection of externally generated key*

   The difference of this setup to the framework described above is that the key is not derived from the PUF response, but provided by external means. This could for example be realized with the help of a RNG and a key generation routine, which generates the key with the desired properties during an enrolment phase. The PUF in this setup plays the role of a “key guard”, as the sensitive key material is simply masked with the binary PUF response. Only if the reconstruction of the PUF response is successful, the de-masking works properly and the key can be used. In this setup entropy extraction by means of a hash function does not need to be put in place additionally, however an external key generator (e.g. RNG) is needed. Figure 2.3 depicts this scenario.

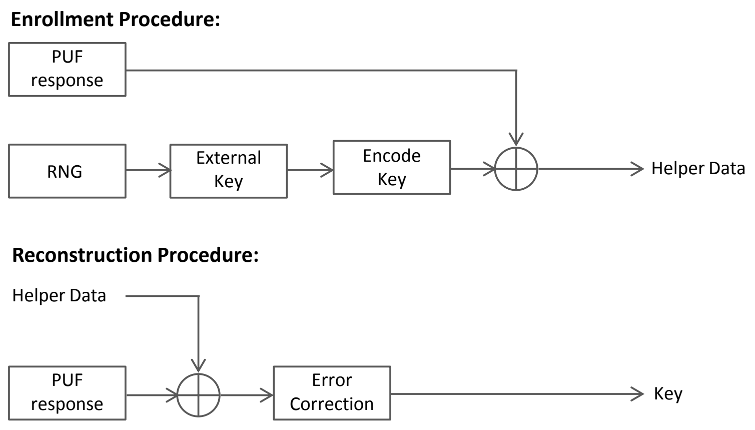

Figure 2.3: Protection of a symmetric key

3. *Development of a public key system*

   In some cases the PUF might be included in a public key infrastructure. In other words the key derived by the PUF represents, for example, the private key in an asymmetric encryption scheme (see Section 5.5.3). In this example the private key y is derived from the PUF and the public key is derived by computing g^y.
   In some cases it might, however, be not possible to directly include the PUF into a public key infrastructure, i.e., it is not possible to derive the private key from the response. In such a case the PUF is used to derive a key that is used to encrypt the used private key of the application.

2.2.4 Attacking a PUF Based Key Generation/Storage Framework

There are various attacks known on several different PUF types. One example is an attack on RO-PUFs as described in [14]. These attacks are mainly based on the fact that helper data is public and can therefore be manipulated. The simplest countermeasure would be to make the helper data not public, this would, however, undermine the PUF principle. Therefore, a solution for preventing helper data manipulation is to generate a keyed Keyed-Hash Message Authentication Code (HMAC) of the helper data [13]. During the enrolment this HMAC is calculated by computing $Hash(p, k)$, where p corresponds to the helper data and k is the PUF based key, and store it with the helper data. As soon as a reconstruction of the key takes place, the reconstructed key is used with the helper data to verify the HMAC. If $HMAC \neq HMAC'$, the reconstruction aborts.
Based on this example, one can see that it is necessary to define different attack scenarios. This attack scenarios will help to identify vulnerable points of the system and how to close them. Within Table 2.1 three scenarios are summarized. The first scenario considers only adversaries that have knowledge of the system architecture and are able to manipulate the helper data. The second scenarios is the more sophisticated one as an attacker has a lot of resources, i.e. an attacker is able to perform side channel analysis or has the money and time for long-term evaluation of the system. The last scenario considers compared to the second scenario also

invasive attacks. This means, an attacker opens the package of the chip to be able to place *e.g.* probes directly on the IC.

	Attacker 1	Attacker 2	Attacker 3
R/W Access to HD	✓	✓	✓
Knowledge of System Architecture	✓	✓	✓
Resources		✓	✓
Invasive Attacks			✓

Table 2.1: Defining the abilities of an Attacker.

These three scenarios are used as a starting point for further investigations which will be carried out during the remaining time of the MATTHEW project.

2.3 Real-World Performance and Security Considerations

One challenge is to develop a cryptographic protocol that fulfils the security and functional requirements. However, in the end the performance characteristics of a secure implementations distinguish a practically usable protocol from other protocols.

The ongoing research within WP2 partners resulted in a paper [25] presented at CHES 2014. Unterluggauer and Wenger implemented and optimised Elliptic Curve Cryptography and Cryptographic Pairings on an embedded processor. Their findings were already shared in MATTHEW meetings before publication. Thus, the protocol designers got early feedback on the potential performance characteristics of the implementation of their protocol.

Although [25] is discussed in detail within D2.2, the most essential achievements are summarised in the following.

The research on pairing-based cryptography brought forth a wide range of protocols interesting for future embedded applications. One significant obstacle for the widespread deployment of pairing-based cryptography are its tremendous hardware and software requirements. Unterluggauer and Wenger [25] present three side-channel protected hardware/software designs for pairing-based cryptography yet small and practically fast: the plain ARM Cortex-M0+-based design computes a pairing in less than one second. The utilization of a multiply-accumulate (MAC) instruction-set extension or a light-weight drop-in hardware accelerator that is placed between CPU and data memory improves runtime up to six times.

With a 10.1 kGE large drop-in module and a 49 kGE large platform, the design is one of the smallest pairing designs available. Its very practical runtime of 162 ms for one pairing on a 254-bit BN curve and its reusability for other elliptic-curve based cryptosystems offer a great solution for every microprocessor-based embedded application.
To verify the achievement of the area and performance goals initially set, the three microprocessor-based platforms (Cortex-M0+, MAC, Drop-in) were evaluated with respect to hard- and software. Regarding the overall hardware platforms, runtime, area, power, and energy consumption are distinctive. Regarding the software part, the evaluation focuses on the runtimes of the underlying finite-field arithmetic and the most expensive operations used within protocols (see Table 2.2): the point multiplications in $\mathbb{G}_1$ and $\mathbb{G}_2$, the exponentiation in $\mathbb{G}_T$, and the pairing operation.

Table 2.2: Performance of various operations on different architectures.

Design	$\mathbb{F}_\mathbf{p}$ **Add**	**Mul**	**Inv**	$\mathbb{G}_\mathbf{1}$ **Mul**	$\mathbb{G}_\mathbf{2}$ **Mul**	$\mathbb{G}_\mathbf{T}$ **Exp**	$\mathbb{G}_\mathbf{1} \times \mathbb{G}_\mathbf{2}$ **Pairing**	**RAM**	**ROM**
	[Cycles]	[Cycles]	[kCycles]	[kCycles]	[kCycles]	[kCycles]	[kCycles]	[Byte]	[Byte]
				BN158					
Cortex-M0+	112	1,800	331	4,828	11,775	22,871	17,389	1,856	13,980
MAC	112	361	72	1,129	4,042	10,736	7,828	1,796	11,232
Drop-in	56	161	29	493	1,577	4,322	3,182	1,876	10,364
				BN254					
Cortex-M0+	166	3,782	1,122	16,071	38,277	72,459	47,643	2,828	18,116
MAC	166	934	285	4,323	11,449	27,460	17,960	2,836	12,572
Drop-in	75	335	97	1,566	4,858	12,076	7,763	2,880	10,764

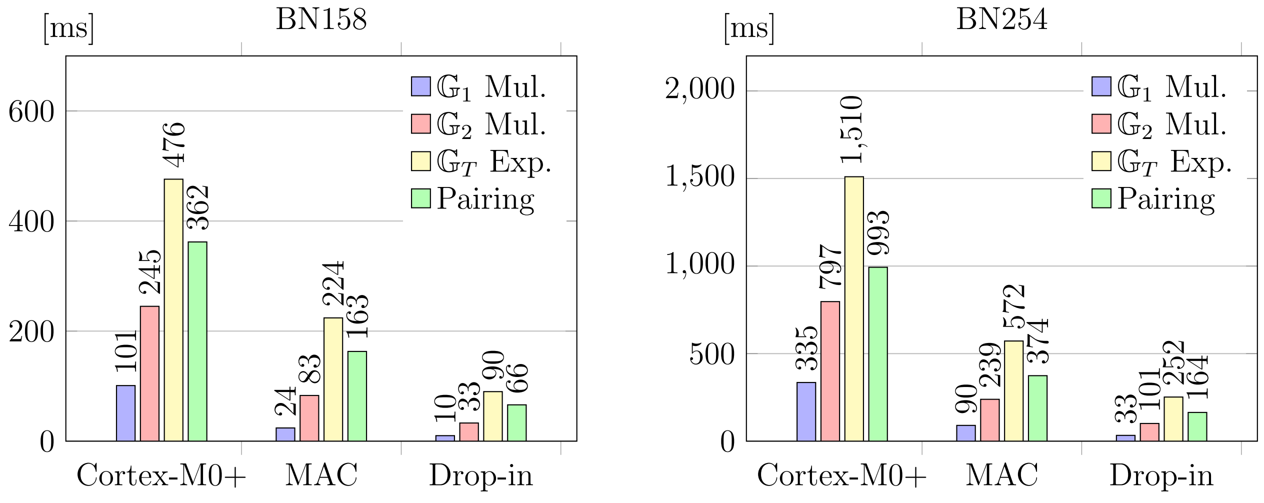

Figure 2.4: Group operations at 48 MHz

Given a typical clock frequency of 48 Mhz, the performance results of the point multiplications in $\mathbb{G}_1$, $\mathbb{G}_2$, the exponentiation in $\mathbb{G}_T$, and the pairing operation are illustrated in Figure 2.4. The respective runtimes support our choice of a 32-bit architecture: providing 128-bit security, the drop-in based platform does pairing computations in highly practical 164 ms. The pure *embedded* software implementation performs the same computation in 993 ms.

Based on these results it is rather straightforward for protocol designers to approximate the potential runtime of their protocol.

Chapter 3

Use Case 1: Advanced Mobile Banking Application

Use case 1 represents a payment application that allows to enable non-NFC smartphones for contactless payment procedures.

In this use case the focus point is clearly set on the technology enabling communication in critical environments. The only easily accessible slot on modern smartphones and tablets is the microSD slot the usage of the microSD form factor is a business enabler for improved payment procedures by means of mobile devices. Alternatively these devices have a Bluetooth Low Energy (BLE) connectivity for the connection to the different peripherals such as wearable devices the user may have and that could integrate the NFC functionality.

3.1 Privacy Aspects

EMVCo compliant payment procedures are not based on protocols aiming for anonymity. Nevertheless the protocols foreseen in use case 1 are taking care of unattended communication via NFC by switching off the NFC connection on the contactless microSD card or in the BLE connected SMD package. Thus users who are not aware of the risk being traced by malicious readers do not need to set any action themselves to prevent tracking of their payment-enabled microSD card or SMD package.

3.2 Transferability of Credentials

Transferability of credentials may herein be seen as the "trivial case" of transferability, interaction with other Secure Elements is not needed in this case. Simply removing the microSD card from one mobile device (*e.g.* a smartphone) and plugging it to the microSD card slot of another one (*e.g.* a tablet) would allow for transfer of the credentials to another platform component, and similarly for the SMD package in a wearable device with the disconnection and reconnection via BLE. Since payment devices are not meant to be used by more than one user PIN-sharing is not foreseen in the EMVCo payment environment and usually prevented by organisational means in conjunction with the contract between the user and the payment service provider, usually a bank. Thus such transfer is not meant as a transfer from one user to another – as it is depicted in use case 3 – but transfer from one mobile device to another mobile device of the same user is feasible.

3.3 Considered Options in the Future

Especially single tap payment protocols that do not foresee any user authentication or Cardholder Verification Method (CMV) may be performed even when the card is in the hands of illegitimate users. Remaining potential security vulnerabilities and social engineering techniques might raise the wish of end users to "lock" the microSD card or SMD package to a dedicated platform instance ("My payment card only works in my mobile phone"). This variant could help ensure that transactions may be performed only on authorized platform devices. In such a variant the protocol for the payment procedure would have to be augmented by an authentication step – *e.g.* an asymmetric challenge-response protocol allowing the payment app on the card to verify that it is plugged to the very device of the intended user. Convenience for the user would not be reduced by such a step, but a stolen card could not be used in a different mobile device.

The appropriate private key for such an asymmetric authentication of the mobile device could be stored securely in the device's embedded secure element (eSE) either in encrypted form in non-volatile memory or as a PUF-key derived from intrinsic production variations of the respective silicon manufacturing process. Only the corresponding public key would have to be transferred to the microSD card (*i.e.* SMD package) to allow for response-verification in the above mentioned protocol variant.

Chapter 4

Use Case 2: Advanced Access Control Application

Use case 2 deals with an advanced access control application that is incorporating a four-eye-principle for access to high-security areas. The protocol foresees the presentation of two mobile devices with the actual credentials being protected by means of secure elements in a SIM-type form factor. When presenting the first mobile device to the reader of the access control system, it is determined who the second user must be to fulfil the four-eyes principle reflected in the access policy for the high-security area.

Thus in the access control system it is mandatory to identify the individuals involved in such case, interaction on protocol level between both SIMs in the two mobile devices and the SAM in the access control system.

4.1 Privacy Aspects

During the actual protocol execution in the access control use case there are no privacy measures in place with respect to the verifier role. This means, that the two single persons being involved are known to the verifier instance, which is mandatory for the access control system in high security areas. To avoid tracking of such persons in environments other than the dedicated protected high-security area the application may switch off the NFC connection after execution of the protocol.

4.2 Transferability of Credentials

In this Use Case 2, credentials may be transferred in a physical way by removal and transfer of the SIM card with access control application from one mobile platform to the other. User authentication towards this application may be achieved by password or biometric features. Logical transfer from one secure element to the other is not foreseen in this type of use case, since access control policies for very high security areas are bound to individual persons, and roles as such must not be transferred from one user to another.

Chapter 5

Use Case 3: Advanced Ticketing Application

5.1 Motivation

Within MATTHEW, the third use-case is meant to show that the use of a secure element, when combined with proper cryptography, can provide a high level of privacy to users in a widespread application such as (public) transport. Although transportation systems usually rely on plain and simple identification of users with maximal disclosure (and therefore with full traceability of their actions), there is no technical requirement to do so and equally secure systems can be cryptographically engineered to preserve the privacy of users while supporting the same basic functionality: access control. In that sense, privacy-preserving transport systems achieve a notion of privacy-by-design where traditional access control mechanisms fail to realize any level of user anonymity.

In this section, we describe an advanced ticketing application that can be implemented in typical transportation networks (trains, metro, bus, etc.). The application achieves a high level of anonymity for users, while making it possible to revoke anonymity for misbehaving users. To exemplify the use of multiple secure elements, we specifically designed the application in such a way that tickets – our notion of ticket is a digital counterpart of regular tickets often printed on mag-striped paper – can be securely transferred from one secure element to another. Therefore tickets can be safely given by users to others across multiple devices, while maintaining privacy.

5.2 Problem Statement

Transportation is a typical context where tracking users is always possible by default. Entering the network requires the presentation of a valid paper ticket or contactless smart card that always embeds some form of a unique identifier. Along with that identifier, access rights and some form of cryptographic evidence of authenticity such as a digital signature are also provided to the access point. Passing through a turnstile with a contactless card (this being a most commonly found scenario with modern transport systems across the globe these days) therefore means that the card is both identified and authenticated at access time. However, cryptography not only provides technical means to authenticate an entity (*i.e.* the card here), it also provides techniques to do so *without* requiring that entity to be identified. Group signatures are such an example of a cryptographic mechanism that achieves that concept. We discuss that in more detail below.

Of course, it is often thought to be desirable to track users when statistics about the network traffic have to be collected. It is usual that suburban trains, metro or bus networks and so forth, maintain some form of database to keep track of how many passengers are found taking a specific line over time, so that the frequency of trains/buses can be adjusted during the day. Ultimately, resource management is given as a justification for the full identification and tracking of passengers. Technically, however, this argument is largely incorrect, because the same traffic information can often be mathematically inferred from just the evolution of the number of accesses at each access point over time, given the geometry of the network. This information, obviously, does not require a full identification of users.

Interestingly, our privacy-preserving ticketing application can also be used to achieve secure access control in a large number of contexts beyond the transportation use-case, such as remote resource management or private cloud storage. Applying our protocol in these contexts would be somewhat straightforward; note however that the protocol could be slightly simplified in these cases as near-field transfers of tickets would not be required.

Privacy-preserving ticketing is also close to anonymous eCash, since one ticket is close to one coin, but the constraints in eCash are more important (mainly because the amount of a spending has to be very flexible), thus technical solutions are not applicable in practice for the moment. For us privacy-preserving ticketing is a better candidate to have an industrial outcome.

5.3 Design Requirements

Several cryptographic techniques support anonymous entity authentication. Among them, attribute-based credentials (or ABCs) are probably the most versatile mechanism, although also the most conceptually complex one. Direct anonymous attestation (DAA) schemes as well as ring signatures provide other technical examples of anonymous authentication. Historically though, group signatures are the most well-known and mature set of cryptographic techniques that support authentication while providing a strong form of anonymity.

Until now in MATTHEW, we have chosen to focus on group signatures rather than on ABCs. Our motivation for that was twofold:

1. group signatures are easier to understand and implement. They also can be adapted to achieve additional properties;

2. tickets do not seem to require a notion of hidden attribute. Possibly, tickets might be attached to geographical areas so that the zones covered by the ticket might be seen as a potential attribute. However, the number of zones is usually small (*e.g.* Paris has exactly 5 zones) and it may seem an overkill for a ticket to hide which zones are covered.

Building upon techniques that are reminiscent to pairing-based group signatures, our first anonymous ticketing protocol supports *single-use* tickets. Indeed we distinguish between two forms of tickets:

- **single-use tickets**: a single-use ticket can only provide access once to the transport system.

- **long-term tickets**: these are credentials that last for a prescribed time duration *e.g.* one month, irrespective of the number of accesses.

These two notions of tickets are reminiscent to the ones usually found in real-life transport networks. At a cryptographic level, however, their features are somewhat different, and capturing these features into a cryptographic scheme calls for different techniques. The first protocol described later on supports single-use tickets, and the second one supports long-term tickets.

An interesting characteristic of our protocols is that they may easily be integrated with a PUF. Tickets are therefore bound to a particular device and can only be presented by that device. This feature enforces the secure element that embeds the enrolled PUF as the only entity that can actually perform related ticket issuance and presentation – unless the PUF key is compromised and the tickets that have been issued but not yet presented are successfully extracted from the device's memory.

5.4 Towards a solution

Since their introduction by Chaum and van Heyst in [10], a lot of group signature constructions have been proposed, for example in [4, 6, 2, 9, 8, 12, 1, 3, 21, 20, 19, 23, 22]. Among those constructions, BBS [4] is known to be a particularly simple and efficient scheme, that can be adapted *mutatis mutandis* to our use case. Some modifications are needed to fulfill the requirements that we have identified for Use Case 3, essentially because a standard group signature such as BBS does not support prevention against double-spending and transfer of credentials. Therefore our protocol, which we describe in detail in the next section, has to support the following enhancements from BBS:

- we need serial numbers (and associated proofs of knowledge) in each ticket presentation (signature), to prevent from double spending;
- the above serial number should depend on group credentials, but should not reveal any information on signer's identity, even to the Issuer (Authority). The group credentials used in BBS are thus not sufficient (since the Issuer knows all the elements);
- we need a partial traceability to detect double-spending of tickets (based on serial numbers)
- we need user revocation
- we need anonymous transfer of credentials (tickets), and anonymous presentation using transferred credentials
- for efficiency reasons, Type-3 pairings are preferred, thus the encryption scheme used in BBS (relying on the DLIN Assumption) is replaced with ElGamal Encryption, and the proofs of knowledge are specifically adapted.

We also augmented the group credentials of BBS with one element not known by the Issuer, as in [12] for example, and we used this element to compute the (unique) serial number associated to a ticket. To perform this computation, the verifiable random function (VRF) proposed by Dodis and Yampolskiy [15] is a natural candidate, already used in similar contexts. User revocation depends on the form of group credentials and can easily be adapted from [12] to the case of Type-3 pairings (some elements in $\mathbb{G}_1$ cannot be computed from elements of $\mathbb{G}_2$, thus they have to be published by the authority). To perform anonymous transfer and enable the use of transferred tickets, we modified the signing (ticket presentation) algorithm.
Overall, our ticketing system builds on BBS but adapts it with specific enhancements that are demanded by the ticketing context.

5.5 Specifying the Cryptographic Protocol for single-use tickets

At a high-level perspective, our protocol presents the following features:

1. Tickets are used only once
2. Users withdraw books of tickets (of flexible size)
3. Centralized database
4. The Issuer can create as many tickets as wanted
5. Strong anonymity (computational) with inspection

5.5.1 Preliminaries

Algebraic setting. All group-related operations needed in the following will apply in three isomorphic cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order p. We furthermore assume that there exists an admissible bilinear map (pairing) $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ which can be evaluated efficiently. This bilinear group system is assumed to be of Type 3 (no efficiently computable isomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$).

5.5.2 Entities

We distinguish between 4 different roles in the system:

1. Issuer $\mathcal{I}$ (*e.g.* a server) with private key $\gamma \in \mathbb{Z}_p$ and public key $w = g_2^{\gamma}$, that manages a public database DB for storing spent tickets serial numbers. The Issuer is the entity that can create new tickets;
2. User ($\mathcal{U}$), typically the secure element;
3. Verifier ($\mathcal{V}$), typically the access point (turnstile equipped with an NFC reader);
4. Inspector/Opener ($\mathcal{O}$) (*e.g.* a server) with a key pair $(\zeta, h = u^{\zeta})$, that can open a presented ticket and reveal from which user it came from.

5.5.3 Procedures

Setup. Before the system can be launched, all system parameters are generated as follows:

1. generate a Type 3 bilinear group system $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e)$;
2. randomly select two generators $g_1 \xleftarrow{\$} \mathbb{G}_1$ and $g_2 \xleftarrow{\$} \mathbb{G}_2$;
3. randomly select generators $u, g \xleftarrow{\$} \mathbb{G}_1$;
4. publish all parameters $(p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, u, g, g_1, g_2)$.

The Issuer and the Opener also take part in the setup process:

1. the Issuer
   (a) randomly selects $\gamma \overset{\$}{\leftarrow} \mathbb{Z}_p$;
   (b) computes $w = g_2^{\gamma}$;
   (c) publishes w and keeps γ private.
2. the Opener
   (a) randomly selects $\zeta \overset{\$}{\leftarrow} \mathbb{Z}_p$;
   (b) computes $h = u^{\zeta}$;
   (c) publishes h and keeps ζ private.

User Registration. A new user $\mathcal{U}$ (*i.e.* a new device) joins the system by successively

1. generating a key pair $(\mathsf{pk}_{\mathcal{U}}, \mathsf{sk}_{\mathcal{U}})$,
2. opening an account by giving $\mathsf{pk}_{\mathcal{U}}$, the the issuer.

$\mathcal{I}$ creates an account indexed by $\mathsf{pk}_{\mathcal{U}}$.

Issuance Protocol. Issuance is a two-party protocol played between $\mathcal{U}$ and $\mathcal{I}$ to jointly generate a book of n tickets. Here n is an arbitrary number that can be fixed in the system, or dynamically chosen by the User at issuance time. Ticket issuance is as follows:

1. $\mathcal{U}$ and $\mathcal{I}$ establish an authenticated encrypted channel
2. $\mathcal{U}$ randomly takes the PUF key y seen as an integer in $\mathbb{Z}_p$, computes $C = h^y$, and a NIZK signature of knowledge π of y under public key $\mathsf{pk}_{\mathcal{U}}$.
3. $\mathcal{U}$ sends C and π to $\mathcal{I}$
4. $\mathcal{I}$ verifies π, picks $x \overset{\$}{\leftarrow} \mathbb{Z}_p$ (different from all previously picked values), computes
$$A = (g_1 \cdot C)^{\frac{1}{\gamma+x}} = (g_1 \cdot h^y)^{\frac{1}{\gamma+x}} \in \mathbb{G}_1$$
and sends (A, x) to $\mathcal{U}$.
5. $\mathcal{U}$ signs $\mathsf{transcript} = (C, \pi, A)$ using its secret key $\mathsf{sk}_{\mathcal{U}}$ and sends to $\mathcal{I}$ the corresponding signature σ.
6. $\mathcal{I}$ checks σ and sends x to $\mathcal{U}$
7. The account is modelled as a chained list
$$\{\mathsf{pk}_{\mathcal{U}}, (\mathsf{transcript}_1, x_1, \sigma_1), \ldots, (\mathsf{transcript}_t, x_t, \sigma_t)\}$$
$\mathcal{I}$ updates the account by appending $(\mathsf{transcript}, x, \sigma)$ to the account related to $\mathsf{pk}_{\mathcal{U}}$.
8. Given (A, x, y), $\mathcal{U}$ checks that $A = (g_1 \cdot h^y)^{\frac{1}{\gamma+x}}$ by verifying that
$$e\left(A, w \cdot g_2^x\right) = e\left(h^y \cdot g_1, g_2\right)$$
$\mathcal{U}$ forms the book of tickets $\mathsf{book} = (A, x, y, J)$, with $J \leq n$ the number of unspent tickets. Note that if n is not fixed, it must be included in book, which would be computed with a generator $g_1^{(n)}$ corresponding to this n. Also note that, although y is a part of the wallet, storing it is unnecessary since it is easily recovered anytime by calling the PUF.

Remark. For simplicity reasons we considered that the PUF key is an element from $\mathbb{Z}_p$, but it is generally not the case. To be more precise, the PUF key can be hashed together with a counter corresponding to a book of tickets, to obtain a value y associated to this book of tickets, and to the PUF key. This also prevents from collisions of serial numbers computed from a given PUF key for various books of tickets.

Ticket Presentation Protocol. The ticket presentation (or ticket spending) protocol is played between $\mathcal{U}$ and $\mathcal{V}$ with respect to a book of tickets $\mathsf{book} = (A, x, y, n, J)$. The transaction parameters given by $\mathcal{V}$ are grouped as $m = \mathsf{time}||\mathsf{infos}||\mathsf{pk}_{\mathcal{V}}$.

1. $\mathcal{U}$ randomly selects $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned} T_1 &= u^{\alpha} \\ T_2 &= A \cdot h^{\alpha} \\ S &= g^{\frac{1}{y+J}} \end{aligned}$$

2. $\mathcal{U}$ randomly picks $r_1, r_2, r_3, r_4 \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned} R_1 &= u^{r_4} \\ R_2 &= e\left(T_2, g_2\right)^{r_1} \cdot e\left(h, w\right)^{-r_4} \cdot e\left(h, g_2\right)^{-r_3} \\ R_3 &= S^{r_2} \end{aligned}$$

3. $\mathcal{U}$ then computes

$$\begin{aligned} c &= \mathcal{H}(T_1, T_2, S, R_1, R_2, R_3, m) \\ s_1 &= r_1 + c \cdot x \bmod p \\ s_2 &= r_2 + c \cdot y \bmod p \\ s_3 &= r_3 + c \cdot (x \cdot \alpha + y) \bmod p \\ s_4 &= r_4 + c \cdot \alpha \bmod p \end{aligned}$$

The (signed) ticket is

$$\mathsf{ticket} = (T_1, T_2, S, c, s_1, s_2, s_3, s_4, m, J)$$

The serial number corresponding to ticket is S. $\mathcal{U}$ sends ticket to $\mathcal{V}$ and decrements J by 1. $\mathcal{V}$ then

1. computes

$$\begin{aligned} \tilde{R_1} &= u^{s_4} \cdot {T_1}^{-c} \\ \tilde{R_2} &= e\left(T_2, g_2\right)^{s_1} \cdot e\left(h, w\right)^{-s_4} \cdot e\left(h, g_2\right)^{-s_3} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \\ \tilde{R_3} &= S^{s_2 + c \cdot J} \cdot g^{-c} \end{aligned}$$

2. checks that

$$\begin{aligned} J &\leq n \\ c &= \mathcal{H}(T_1, T_2, S, \tilde{R_1}, \tilde{R_2}, \tilde{R_3}, m) \end{aligned}$$

3. grants access if it is valid. $\mathcal{V}$ keeps $\mathsf{receipt} = (m, \mathsf{ticket})$ in its backlog.

Note that the computations can be optimized, as described in section 5.6.

Background Validation Protocol. Protocol played between $\mathcal{V}$ and $\mathcal{O}$ to report spent tickets to the Issuer.

1. $\mathcal{V}$ sends $\mathsf{receipt} = (m, \mathsf{ticket})$ to $\mathcal{I}$, no need for an authenticated nor encrypted channel;
2. $\mathcal{I}$ verifies ticket and checks in its database DB that S is absent. If so, S is added to the database, otherwise the ticket attached to the serial number S has been used at least twice, and $\mathcal{I}$ takes a security measure such as requiring opening the ticket and revoking the device.

Ticket Opening. Given $\mathsf{receipt} = (m, \mathsf{ticket})$, $\mathcal{O}$:

- parses ticket as $(T_1, T_2, S, c, s_1, s_2, s_3, s_4, m, J)$
- computes $A = T_2 \cdot T_1^{-\zeta}$, and retrieves the corresponding user (with an account containing a $\mathsf{transcript}$ including A)

5.5.4 Offline Transfer

Protocol that involves two users $\mathcal{U}_1$ and $\mathcal{U}_2$. The two of them are assumed to be offline at transfer time, but a communication channel is required between $\mathcal{U}_1$ and $\mathcal{U}_2$ (NFC communication between the two secure elements). $\mathcal{U}_1$ is assumed to own a non-empty book of tickets, and $\mathcal{U}_2$ is only assumed to be a registered user, with at least one book of tickets (possibly empty, but used in the transfer in order to have traceable credentials). At a high-level perspective, the transfer consists of the following stages:

1. Spending of ticket t_1 by $\mathcal{U}_1$, where $\mathcal{U}_2$ plays the Verifier. The signed message must include an information indicating that the transaction is a transfer, together with an information related to the receiver of the transferred ticket.
2. $\mathcal{U}_2$ can then spend this ticket by sending it to a verifier, together with a (privacy-preserving) proof that he is the actual receiver of the transfered ticket. This can be achieved by including a masked element corresponding to $\mathcal{U}_2$ in the signed information during the spending.

More formally, the transfer protocol is as follows:

Offline Transfer Protocol. The offline transfer protocol is played between two users $\mathcal{U}_1$ (owner of a book of tickets $\mathsf{book}_1 = (A_1, x_1, y_1, n, J_1))$ and $\mathcal{U}_2$ (owner of a possibly empty book of tickets $\mathsf{book}_2 = (A_2, x_2, y_2, n, J_2))$.

1. $\mathcal{U}_2$ randomly selects $r \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes

$$\begin{aligned} E_1 &= u^r \\ E_2 &= A_2 \cdot h^r \end{aligned}$$

The transaction parameters given by $\mathcal{U}_2$ are grouped as $m = \mathsf{time}||E_1||E_2$.

2. $\mathcal{U}_1$ randomly selects $\alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned} T_1 &= u^{\alpha} \\ T_2 &= A_1 \cdot h^{\alpha} \\ S &= g^{\frac{1}{y_1 + J_1}} \end{aligned}$$

3. $\mathcal{U}_1$ randomly picks $r_1, r_2, r_3, r_4 \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned} R_1 &= u^{r_4} \\ R_2 &= e\left(T_2, g_2\right)^{r_1} \cdot e\left(h, w\right)^{-r_4} \cdot e\left(h, g_2\right)^{-r_3} \\ R_3 &= S^{r_2} \end{aligned}$$

4. $\mathcal{U}_1$ then computes

$$\begin{aligned} c &= \mathcal{H}(T_1, T_2, S, R_1, R_2, R_3, m) \\ s_1 &= r_1 + c \cdot x_1 \bmod p \\ s_2 &= r_2 + c \cdot y_1 \bmod p \\ s_3 &= r_3 + c \cdot (x_1 \cdot \alpha + y_1) \bmod p \\ s_4 &= r_4 + c \cdot \alpha \bmod p \end{aligned}$$

The (signed) ticket is

$$\mathsf{ticket} = (T_1, T_2, S, c, s_1, s_2, s_3, s_4, m, J_1)$$

The serial number corresponding to ticket is S. $\mathcal{U}_1$ sends ticket to $\mathcal{U}_2$ and decrements J_1 by 1. $\mathcal{U}_2$ then (optionally)

1. computes

$$\begin{aligned} \tilde{R}_1 &= u^{s_4} \cdot {T_1}^{-c} \\ \tilde{R}_2 &= e\left(T_2, g_2\right)^{s_1} \cdot e\left(h, w\right)^{-s_4} \cdot e\left(h, g_2\right)^{-s_3} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \\ \tilde{R}_3 &= S^{s_2 + c \cdot J} \cdot g^{-c} \end{aligned}$$

2. checks that

$$\begin{aligned} J_1 &\leq n \\ c &= \mathcal{H}(T_1, T_2, S, \tilde{R}_1, \tilde{R}_2, \tilde{R}_3, m) \end{aligned}$$

3. keeps $\mathsf{receipt}_{1,2} = (m, \mathsf{ticket})$ in book_2.

Note that as in the Ticket Presentation Protocol, the computations can be optimized, as described in section 5.6.
Note also that we do not need U_2 to prove that E_1 and E_2 are correctly computed, because if it is not the case, the following presentation protocol will not succeed.

Transferred Ticket Presentation Protocol. The transferred ticket presentation protocol is played between $\mathcal{U}_2$ and $\mathcal{V}$ with respect to a book of tickets $\mathsf{book}_2 = (A_2, x_2, y_2, n, J_2, \mathsf{receipt}_{1,2}, r)$. The transaction parameters given by $\mathcal{V}$ are grouped as $m' = \mathsf{time}||\mathsf{infos}||\mathsf{pk}_{\mathcal{V}}$.

1. $\mathcal{U}_2$ randomly picks $r_5, r_6, r_7 \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes (with E_2 in m)

$$\begin{aligned} R'_1 &= u^{r_7} \\ R'_2 &= e(E_2, g_2)^{r_5} \cdot e(h, w)^{-r_7} \cdot e(h, g_2)^{-r_6} \end{aligned}$$

2. $\mathcal{U}_2$ then computes

$$\begin{aligned} c &= \mathcal{H}(\mathsf{receipt}, R'_1, R'_2, m') \\ s_5 &= r_5 + c \cdot x_2 \bmod p \\ s_6 &= r_6 + c \cdot (x_2 \cdot r + y_2) \bmod p \\ s_7 &= r_7 + c \cdot r \bmod p \end{aligned}$$

The (signed) ticket is

$$\mathsf{ticket} = (\mathsf{receipt}_{1,2}, c, s_5, s_6, s_7, m')$$

The serial number corresponding to ticket is S (included in $\mathsf{receipt}_{1,2}$). $\mathcal{U}_2$ sends ticket to $\mathcal{V}$ and delete ($\mathsf{receipt}_{1,2}$, r) from book_2.
$\mathcal{V}$ then

1. verifies that $\mathsf{receipt}_{1,2}$ is correct (Ticket Presentation Protocol),
2. computes

$$\begin{aligned} \tilde{R}'_1 &= u^{s_7} \cdot {R'_1}^{-c} \\ \tilde{R}'_2 &= e(E, g_2)^{s_5} \cdot e(h, w)^{-s_7} \cdot e(h, g_2)^{-s_6} \cdot e(g_1, g_2)^{-c} \cdot e(E, w)^{c} \end{aligned}$$

3. checks that

$$c = \mathcal{H}(\mathsf{receipt}, \tilde{R}'_1, \tilde{R}'_2, m')$$

4. grants access if it is valid. $\mathcal{V}$ keeps $\mathsf{receipt} = (m', \mathsf{ticket})$ in its backlog.

Note that the computations can also be optimized, as described in section 5.6.

Background Validation Protocol. Identical to the one for regular (not transferred) tickets.

Ticket Opening. The Opener can retreive A_1 thanks to the previous Ticket Opening (applied to $\mathsf{receipt}_{1,2}$), or A_2 by computing $A_2 = E_2 \cdot E_1^{-\zeta}$.

5.5.5 Online Transfer

Protocol that involves two users $\mathcal{U}_1$ and $\mathcal{U}_2$ and the Issuer. The two of them are assumed to be online at transfer time.

1. Spending of ticket t_1 by $\mathcal{U}_1$ with $\mathcal{I}$ as the verifier
2. Issuing of a new equivalent ticket t_2 to $\mathcal{U}_2$.

Remark. In case of high efficiency constraint, the new issuing could be replaced by erasing a serial number in DB (corresponding to a ticket spent by $\mathcal{U}_2$, which then could be used once again), but in this case the two transactions could be linked, since they would contain the same serial number.

5.6 Efficiency Optimization

The ticket presentation protocol (essentially the computation of R_2) is the most critical part of the system, since it has to be executed in constrained devices, in a very limited period of time. For this reason, the choice of the representation of the computations can be important. More precisely, the computation of the element R_2 can be rewritten as follows :

$$\begin{aligned} R_2 &= e(T_2, g_2)^{r_1} \cdot e(h, w)^{-r_4} \cdot e(h, g_2)^{-r_3} \\ &= e\left(T_2^{r_1} \cdot h^{-r_3}, g_2\right) \cdot e(h, w)^{-r_4} \end{aligned}$$

Thus, by pre-computing $V = e(h, w)$ during the Setup, the computation of R_2 can be reduced to one multi-exponentiation in $\mathbb{G}_1$, one exponentiation in $\mathbb{G}_T$, and one pairing.
Depending on the implementation, one could prefer the following representation, with pre-computation of $W = e(h, w)$ during the Setup and $B = e(A, g_2)$ during the Issuance:

$$\begin{aligned} R_2 &= e(T_2, g_2)^{r_1} \cdot e(h, w)^{-r_4} \cdot e(h, g_2)^{-r_3} \\ &= e(A, g_2)^{r_1} \cdot e(h, g_2)^{r_1 \cdot \alpha} \cdot e(h, w)^{-r_4} \cdot e(h, g_2)^{-r_3} \\ &= B^{r_1} \cdot V^{r_1 \cdot \alpha - r_3} \cdot W^{-r_4} \end{aligned}$$

In this case, the computation of R_2 can be reduced to one multi-exponentiation in $\mathbb{G}_T$.
The verification algorithm can also be optimized in a similar way:

$$\begin{aligned} \tilde{R_2} &= e(T_2, g_2)^{s_1} \cdot e(h, w)^{-s_4} \cdot e(h, g_2)^{-s_3} \cdot e(g_1, g_2)^{-c} \cdot e(T_2, w)^{c} \\ &= e\left(T_2^{s_1} \cdot h^{-s_3} \cdot g_1, g_2\right) \cdot e(h, w)^{-s_4} \cdot e(h, g_2)^{-s_3} \cdot e(g_1, g_2)^{-c} \cdot e(T_2, w)^{c} \end{aligned}$$

Another possible optimization. With state-of-the-art pairing implementations, pairing evaluations can be cheaper than exponentions in $\mathbb{G}_T$. In this case, it R_2 can be rewritten as follows

$$\begin{aligned} R_2 &= e(T_2, g_2)^{r_1} \cdot e(h, w)^{-r_4} \cdot e(h, g_2)^{-r_3} \\ &= e\left(T_2^{r_1} \cdot h^{-r_3}, g_2\right) \cdot e\left(h^{-r_4}, w\right) \end{aligned}$$

In this case, the computation of R_2 can be reduced to one multi-exponentiation and one exponentiation in $\mathbb{G}_1$, and two pairing computations.

5.7 Security Properties

The following properties shall be verified by the protocol.

Correctness. Honestly-generated ticket presentations verify and open correctly.

Traceability. A collusion of malicious Users and the Issuer cannot present a ticket that does not trace (after opening) to a member of the coalition.

Anonymity. The issuance and presentation of a ticket are unlinkable even if the Issuer and Verifier collude.

Detection of double spending. A malicious User cannot double-spend a valid ticket without being detected by the Issuer.

Proof of Correctness. Let $\mathsf{ticket} = (T_1, T_2, S, c, s_1, s_2, s_3, s_4, m, J)$ be an honestly-generated ticket, grom the book of tickets $\mathsf{book} = (A, x, y, J)$. Then

$$
\begin{aligned}
\tilde{R_1} &= u^{s_4} \cdot {T_1}^{-c} = u^{r_4 + c\alpha} \cdot u^{-c\alpha} = u^{r_4} = R_1 \\
\tilde{R_2} &= e\left(T_2, g_2\right)^{s_1} \cdot e\left(h, w\right)^{-s_4} \cdot e\left(h, g_2\right)^{-s_3} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \\
&= e\left(T_2, g_2\right)^{r_1 + c\cdot x} \cdot e\left(h, w\right)^{-r_4 - c\alpha} \cdot e\left(h, g_2\right)^{-r_3 - c(x\alpha + y)} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \\
&= R_2 \cdot e\left(T_2, g_2\right)^{c\cdot x} \cdot e\left(h, w\right)^{-c\alpha} \cdot e\left(h, g_2\right)^{-c\cdot(x\alpha + y)} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \\
&= R_2 \cdot e\left(A, w \cdot g_2^x\right)^{c} \cdot e\left(h^\alpha, w \cdot g_2^x\right)^{c} \cdot e\left(h, w\right)^{-c\alpha} \cdot e\left(h, g_2\right)^{-c\cdot(x\alpha + y)} \cdot e\left(g_1, g_2\right)^{-c} \\
&= R_2 \cdot e\left(h^y \cdot g_1, g_2\right)^{c} \cdot e\left(h^\alpha, g_2^x\right)^{c} \cdot e\left(h, g_2\right)^{-c\cdot(x\alpha + y)} \cdot e\left(g_1, g_2\right)^{-c} \\
&= R_2 \cdot e\left(h^y, g_2\right)^{c} \cdot e\left(h^\alpha, g_2^x\right)^{c} \cdot e\left(h, g_2\right)^{-c\cdot(x\alpha + y)} \\
&= R_2 \cdot e\left(h, g_2\right)^{c\cdot y} \cdot e\left(h, g_2\right)^{c\cdot x\cdot\alpha} \cdot e\left(h, g_2\right)^{-c\cdot(x\alpha + y)} \\
&= R_2 \\
\tilde{R_3} &= S^{s_2 + c\cdot J} \cdot g^{-c} = S^{r_2 + c\cdot y + c\cdot J} \cdot S^{-c(y+J)} \\
&= S^{r_2} = R_3
\end{aligned}
$$

Thus $\mathcal{H}(T_1, T_2, S, \tilde{R_1}, \tilde{R_2}, \tilde{R_3}, m) = \mathcal{H}(T_1, T_2, S, R_1, R_2, R_3, m) = c$.

Proof of Anonymity. The Anonymity relies on the DDH assumption. The proof is similar to the proof of anonymity in BBS, adapted to the case of El Gamal encryption instead of Linear encryption (thus DDH assumption instead of DLin assumption). The serial number as well as the related proof of knowledge can be perfectly simulated, as the rest of the signature.

Proof of Traceability and Detection of double spending. This property mainly results from the traceability of the group signature of [12], and the security of the VRF of [15]. We mainly have to modify the underlying assumption (see below), to adapt to the case of Type-3 pairing (because of the absence of an isomorphism from $\mathbb{G}_2$ to $\mathbb{G}_1$). The detection of double spending relies on the security of the VRF (relying on the DBDHI assumption), which ensures that it is hard to compute a valid serial number without knowning the secret key, and that each serial number is unique, for one value of the index.

5.7.1 Complexity Assumptions

Let G_1, G_2 be cyclic groups of prime order p. Let g_1 be a generator of G_1 and g_2 a generator of G_2.

q-Strong Diffie-Hellman Problem. The q-SDH problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is defined as follows: given a $(q+2)$-tuple $(g_1, g_2, g_2^{\gamma}, \ldots, g_2^{\gamma^q})$ as input, output a pair $(g_1^{\frac{1}{\gamma+x}}, x)$ where $x \in \mathbb{Z}_p^\star$. This problem was used in [5], and then in a lot of papers, like [4] for example.

Modified Strong Diffie-Hellman Problem. The problem in $(\mathbb{G}_1, \mathbb{G}_2)$ is the same as the original q-SDH problem, except that the tuple has to be given also in $\mathbb{G}_1$. More precisely, the tuple $(g_1, g_1^{\gamma}, \ldots, g_1^{\gamma^q}, g_2, g_2^{\gamma}, \ldots, g_2^{\gamma^q})$ is given as input, and the output is unchanged, and is a pair $(g_1^{\frac{1}{\gamma+x}}, x)$ where $x \in \mathbb{Z}_p^\star$.

One can note that if there exists an isomorphism from G_2 to G_1, this problem is equivalent to the original q-SDH problem, since the additional elements of the input in G_1 could be computed from the tuple in G_2. In our context (Type-3 pairing), there is no such isomorphism, but the problem is stil difficult (the same generic arguments as for the first problem hold).

5.8 Performance Analysis and Profiling

We shall focus on the following performance metrics:

- **Size of books of tickets:** a book of tickets is of the form $\mathsf{book} = (A, x, y, J)$ with $J \leq n$. The PUF key y needs not be stored in the wallet but can be recovered by calling the PUF. Since elements of $\mathbb{G}_1$ are quite compact, a wallet could therefore be made as small as $\simeq 512 + 2\log_2 n$ bits for 128 bits of security.
- **Time of a presentation:** this remains to be investigated through implementation. The required time to carry out a ticket presentation is critical in practice since the access point has a limited time window of a few hundreds of milliseconds to allow or deny access. Full presentation must include transmissions over the NFC channel.
- **Time of an issuance:** this remains to be investigated through implementation.
- **Time of an online/offline transfer:** this remains to be investigated through implementation.

5.9 Implementation Security

Implementing the ticketing application securely requires an appropriate level of resistance against physical attacks such as side channels and fault injection. Here, the works on pairing implementations discussed in Section 2.3 shall be useful, as well as to provide efficient implementations for group operations.

5.10 Protocol 2 : long-term tickets

5.10.1 Features.

1. One ticket per device (monthly subscription for example)
2. Tickets are used until expiration or revocation

5.10.2 Entities.

1. Issuer $\mathcal{I}$ with private key $\gamma \in \mathbb{Z}_p$ and public key $w = g_2^{\gamma}$, a public database DB for storing spent tickets serial numbers.
2. User ($\mathcal{U}$)
3. Verifier ($\mathcal{V}$)
4. Inspector/Opener ($\mathcal{O}$) with a key pair $(\zeta, h = u^{\zeta})$.

5.10.3 Procedures

Group System. All group-related operations needed in the following will apply in three isomorphic cyclic groups $\mathbb{G}_1$, $\mathbb{G}_2$ and $\mathbb{G}_T$ of prime order p. We furthermore assume that there exists an admissible bilinear map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ which can be evaluated efficiently.

Setup. All system parameters are generated:

1. the Issuer generates groups $\mathbb{G}_1$, $\mathbb{G}_2$, $\mathbb{G}_T$, randomly selects generators $g_1 \overset{\$}{\leftarrow} \mathbb{G}_1$, $g_2 \overset{\$}{\leftarrow} \mathbb{G}_2$, $(u, g) \overset{\$}{\leftarrow} \mathbb{G}_1^2$.
2. $\mathcal{I}$ also generates $\gamma \overset{\$}{\leftarrow} \mathbb{Z}_p$, and computes $w = g_2^{\gamma}$,
3. the opener randomly selects $\zeta \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes $h = u^{\zeta}$.

Registration A new user $\mathcal{U}$ (*i.e.* device) joins the system by successively

1. generating a key pair $(\mathsf{pk}_{\mathcal{U}}, \mathsf{sk}_{\mathcal{U}})$,
2. opening an account by giving $\mathsf{pk}_{\mathcal{U}}$, the the issuer.

$\mathcal{I}$ creates an account indexed by $\mathsf{pk}_{\mathcal{U}}$.

Tickets Issuing Protocol. Protocol played between $\mathcal{U}$ and $\mathcal{I}$ to generate a ticket (which corresponds to a period of time for example).

1. $\mathcal{U}$ and $\mathcal{I}$ establish an authenticated encrypted channel
2. $\mathcal{U}$ randomly selects $y \overset{\$}{\leftarrow} \mathbb{Z}_p$, computes $C = h^y$, and a NIZK signature of knowledge π of y under public key $\mathsf{pk}_{\mathcal{U}}$.
3. $\mathcal{U}$ sends C and π to $\mathcal{I}$
4. $\mathcal{I}$ verifies π, picks $x \overset{\$}{\leftarrow} \mathbb{Z}_p$ (different from all previously picked values), computes
$$A = (g_1 \cdot C)^{\frac{1}{\gamma+x}} = (g_1 \cdot h^y)^{\frac{1}{\gamma+x}} \in \mathbb{G}_1$$
and sends (A, x) to $\mathcal{U}$.
5. $\mathcal{U}$ signs $\mathsf{transcript} = (C, \pi, A)$ using its secret key $\mathsf{sk}_{\mathcal{U}}$ and sends to $\mathcal{I}$ the corresponding signature σ.

6. $\mathcal{I}$ checks σ and sends x to $\mathcal{U}$
7. The account is modelled as a chained list

$$\{\mathsf{pk}_{\mathcal{U}}, (\mathsf{transcript}_1, x_1, \sigma_1), \ldots, (\mathsf{transcript}_t, x_t, \sigma_t)\}$$

   $\mathcal{I}$ updates the account by appending $(\mathsf{transcript}, x, \sigma)$ to the account related to $\mathsf{pk}_{\mathcal{U}})$.

8. Given (A, x, y), $\mathcal{U}$ checks that $A = (g_1 \cdot h^y)^{\frac{1}{\gamma+x}}$ by verifying that

$$e\left(A, w \cdot g_2^x\right) = e\left(h^y \cdot g_1, g_2\right)$$

   $\mathcal{U}$ forms the ticket $\mathsf{ticket} = (A, x, y)$.

Ticket Presentation Protocol. Protocol played between $\mathcal{U}$ and $\mathcal{V}$ with respect to ticket $\mathsf{ticket} = (A, x, y)$. The transaction parameters given by $\mathcal{V}$ are grouped as $m = \mathsf{time}||\mathsf{infos}||\mathsf{pk}_{\mathcal{V}}$.

1. $\mathcal{U}$ randomly selects $\alpha, \beta \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes

$$\begin{aligned} T_1 &= u^\alpha \\ T_2 &= A \cdot h^\alpha \end{aligned}$$

2. $\mathcal{U}$ randomly picks $r_1, r_2, r_3 \overset{\$}{\leftarrow} \mathbb{Z}_p$ and computes

$$\begin{aligned} R_1 &= u^{r_2} \\ R_2 &= e\left(T_2, g_2\right)^{r_1} \cdot e\left(h, w\right)^{-r_2} \cdot e\left(h, g_2\right)^{-r_3} \end{aligned}$$

3. $\mathcal{U}$ then computes

$$\begin{aligned} c &= \mathcal{H}(T_1, T_2, R_1, R_2, m) \\ s_1 &= r_1 + c \cdot x \bmod p \\ s_2 &= r_2 + c \cdot y \bmod p \\ s_3 &= r_3 + c \cdot (x \cdot \alpha + y) \bmod p \end{aligned}$$

The signature is

$$\sigma = (T_1, T_2, c, s_1, s_2, s_3, m)$$

$\mathcal{U}$ sends σ to $\mathcal{V}$. $\mathcal{V}$ then

1. computes

$$\begin{aligned} \tilde{R}_1 &= u^{s_2} \cdot {T_1}^{-c} \\ \tilde{R}_2 &= e\left(T_2, g_2\right)^{s_1} \cdot e\left(h, w\right)^{-s_2} \cdot e\left(h, g_2\right)^{-s_3} \cdot e\left(g_1, g_2\right)^{-c} \cdot e\left(T_2, w\right)^{c} \end{aligned}$$

2. checks that

$$c = \mathcal{H}(T_1, T_2, \tilde{R}_1, \tilde{R}_2, m)$$

3. grants access if it is valid. $\mathcal{V}$ keeps $\mathsf{receipt} = (m, \sigma)$ in its backlog.

Open. Given $\mathsf{receipt} = (m, \sigma)$, $\mathcal{O}$:

- parses σ as $(T_1, T_2, c, s_1, s_2, s_3, m)$
- computes $A = T_2 \cdot T_1^{-\zeta}$, and retrieves the corresponding user (with an account containing a $\mathsf{transcript}$ including A)

Revocation. In case of abuse (if a user did not pay his monthly subscription for example), the issuer has to be able to revoke a ticket. To this aim, the following methods can be considered:

- Use of a revocation list, which contains one element per revoked ticket (added by the issuer). In this case the verifier has to verify that a given signature has not been computed using revoked ticket. The drawback of this method is the size of the revocation list, which is linear in the number of revoked tickets. The verification is also linear in this number.
- Update of the parameters of the system, depending on the list of revoked tickets. In this case, the issuer can publish some elements, computed from the revoked tickets, so that only non-revoked tickets can be update. If this method is used, the non-revoked tickets have to be updated by their owners, which suppose that they are online before using their tickets after a system update.

We chose the second method, but the first one can be adapted to our protocol to manage short term revocations. More precisely, the long term revocation of BBS+ or [12] can be used in our context, since the personnal keys contained in a book of tickets are similar: when a book of tickets $\mathsf{book} = (A, x, y, J)$ has to be revoked, the Issuer can update and publish the parameters $\tilde{g_1}, \tilde{g_2}, \tilde{h}$ and $\tilde{u}$ such that

$$\tilde{g_1} = g_1^{\frac{1}{\gamma+x}} \ , \tilde{g_2} = g_2^{\frac{1}{\gamma+x}} \ , \tilde{h} = h^{\frac{1}{\gamma+x}} \ , \tilde{u} = u^{\frac{1}{\gamma+x}} \ , \tilde{w} = w^{\frac{1}{\gamma+x}} \ .$$

Note that $\tilde{w}$ could also be publicly computed as : $\tilde{w} = g_2 \cdot \tilde{g_2}^x$.
Then a user with an unrevoked book of tickets $\mathsf{book}_i = (A_i, x_i, y_i, J_i)$ can update A_i by computing:

$$\tilde{A_i} = (\tilde{g_1} \cdot \tilde{h}^y \cdot A_i)^{\frac{1}{x-x_i}} = \left(\tilde{g_1} \cdot \tilde{h}^y\right)^{\frac{1}{\gamma+x_i}}$$

The issuer can also keep track of this modification for each registered book of tickets.

Chapter 6

Conclusive Remarks

6.1 Concluding on Multiple Entity Security

We explored in this document the transferability of credentials in the three specific contexts of the MATTHEW use cases. Although Use Case 1 (payment) and Use Case 2 (access control) are not specifically conceived to support the transfer of credentials other than through straightforward copy-deletion of cryptographic keys, Use Case 3 supports transferability by design. Transfers of tickets can be performed either in an online fashion using revocation and issuance, or offline (using the NFC channel typically) by a specific cryptographic mechanism.
Interestingly, in the context of Use Case 3, our cryptographic design enjoys all the sought properties identified during the requirement analysis performed in the early stages of the project. We may therefore claim that we successfully established the basis for an efficient cryptographic solution for privacy-respecting ticketing with by-design multiple entity security.

6.2 Future Advances

Cryptographic design enables a continuum of techniques, and our ticketing application may benefit from other adaptations. More explicitly, Attribute-Based Credentials or ABCs are a set of techniques that encompasses and goes beyond group signatures on which our ticketing system is based. Simple examples may show how ABCs could help improving on our current solution.
In a real-world transport application, it is usual to have a notion of geographical areas i.e. the network map is divided into several zones and there is a list of fares that depend on which zones are covered by the traveller across her journey (fares usually increase monotonically with area coverage). Addressing this aspect with our current solution can only be done by *superposing* multiple instantiations of our system; tickets attached to a certain zone (or more generally category) would have their own individual instance of the system, with their own parameters and so forth. Consequently, the real-world system actually becomes a superposition of different instances and involved entities (issuer, verifiers and users) possibly have to support multiple systems working in a concurrent fashion.
In ABCs, the zone(s) attached to a ticket (or any other notion of category) are captured as *an attribute* within a unique system. A ticket may then be seen as a container that collects a set of attributes e.g. covered geographical zone, fare (senior, student, unemployed, etc). Moreover, adopting an ABC system would allow to prove assertions on a ticket's attribute without revealing them: the verifier only ascertains that the predicate is true e.g. "covered zones lie within zone 3" but the actual value of the zone attribute remains hidden. This feature

implements the *minimal disclosure* paradigm and would therefore increase yet again the level of anonymity for users.
Given the attractiveness of minimal disclosure, which one cannot possibly improve upon in terms of privacy, we may envision in the project's roadmap to design specific extensions of our current ticketing system to support a few real-world attributes such as zones and special fares. These extensions would build on approximately the same low-level bricks that will have been implemented within MATTHEW, possibly including extra functionalities to support zero-knowledge proofs on hidden attributes.

Chapter 7

List of Abbreviations

ABC	Attribute-Based Credentials
BCH code	Bose-Chaudhuri-Hocquenghem code
BN curve	BarretoNaehrig curve
CMV	Card-Holder Verification Method
EC	European Commission
ECDSA	Elliptic Curve Digital Signature Algorithm
eSE	Embedded Secure Element
HDA	Helper Data Algorithm
IC	Integrated Circuit
NFC	Near-Field Communication
NIZK	Non-Interactive Zero-Knowledge
NVM	Non-Volatile Memory
PIN	Personal Identification Number
PKI	Private Key Infrastructure
PUF	Physically Unclonable Function
RNG	Random Number Generator
SD	Secure Digital
SIM	Subscriber Identity Module
SRAM	Static Random-Access Memory
TBD	to be determined
TPM	Trusted Platform Module

Bibliography

[1] Giuseppe Ateniese, Jan Camenisch, Susan Hohenberger, and Breno de Medeiros. Practical group signatures without random oracles. *IACR Cryptology ePrint Archive*, 2005:385, 2005.

[2] Giuseppe Ateniese, Jan Camenisch, Marc Joye, and Gene Tsudik. A practical and provably secure coalition-resistant group signature scheme. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*, pages 255–270. Springer, 2000.

[3] Giuseppe Ateniese and Breno de Medeiros. Efficient group signatures without trapdoors. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*, pages 246–268. Springer, 2003.

[4] Dan Boneh, Xavier Boyen, and Hovav Shacham. Short group signatures. In Franklin [16], pages 41–55.

[5] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In Colin Boyd, editor, *Advances in Cryptology - ASIACRYPT 2001, 7th International Conference on the Theory and Application of Cryptology and Information Security, Gold Coast, Australia, December 9-13, 2001, Proceedings*, volume 2248 of *Lecture Notes in Computer Science*, pages 514–532. Springer, 2001.

[6] Dan Boneh and Hovav Shacham. Group signatures with verifier-local revocation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, editors, *Proceedings of the 11th ACM Conference on Computer and Communications Security, CCS 2004, Washington, DC, USA, October 25-29, 2004*, pages 168–177. ACM, 2004.

[7] Christoph Bösch, Jorge Guajardo, Ahmad-Reza Sadeghi, Jamshid Shokrollahi, and Pim Tuyls. Efficient Helper Data Key Extractor on FPGAs. In *Cryptographic Hardware and Embedded Systems CHES 2008*, volume 5154 of *Lecture Notes in Computer Science*, pages 181–197. Springer Berlin Heidelberg, 2008.

[8] Xavier Boyen and Brent Waters. Compact group signatures without random oracles. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 427–444. Springer, 2006.

[9] Jan Camenisch and Anna Lysyanskaya. Signature schemes and anonymous credentials from bilinear maps. In Franklin [16], pages 56–72.

[10] David Chaum and Eugène van Heyst. Group signatures. In Donald W. Davies, editor, *Advances in Cryptology - EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques, Brighton, UK, April 8-11, 1991, Proceedings*, volume 547 of *Lecture Notes in Computer Science*, pages 257–265. Springer, 1991.

[11] Nicolas T. Courtois, Karsten Nohl, and Sean O'Neil. Algebraic Attacks on the Crypto-1 Stream Cipher in MiFare Classic and Oyster Cards. Cryptology ePrint Archive, Report 2008/166, April 2008.

[12] Cécile Delerablée and David Pointcheval. Dynamic fully anonymous short group signatures. In Phong Q. Nguyen, editor, *Progressin Cryptology - VIETCRYPT 2006, First International Conferenceon Cryptology in Vietnam, Hanoi, Vietnam, September 25-28, 2006, Revised Selected Papers*, volume 4341 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2006.

[13] Jeroen Delvaux, Dawu Gu, Dries Schellekens, and Ingrid Verbauwhede. Helper Data Algorithms for PUF-Based Key Generation: Overview and Analysis. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, PP(99):1–1, 2014.

[14] Jeroen Delvaux and Ingrid Verbauwhede. Key-recovery attacks on various ro puf constructions via helper data manipulation. In *Proceedings of the conference on Design, Automation & Test in Europe*, page 72. European Design and Automation Association, 2014.

[15] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In Serge Vaudenay, editor, *Public Key Cryptography - PKC 2005, 8th International Workshop on Theory and Practice in Public Key Cryptography, Les Diablerets, Switzerland, January 23-26, 2005, Proceedings*, volume 3386 of *Lecture Notes in Computer Science*, pages 416–431. Springer, 2005.

[16] Matthew K. Franklin, editor. *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*. Springer, 2004.

[17] Helena Handschuh. Hardware-anchored security based on sram pufs, part 1. *IEEE Security and Privacy*, 10(3):80–83, 2012.

[18] P. Koeberl, Jiangtao Li, A. Rajan, and Wei Wu. Entropy loss in puf-based key generation schemes: The repetition code pitfall. In *Hardware-Oriented Security and Trust (HOST), 2014 IEEE International Symposium on*, pages 44–49, May 2014.

[19] Fabien Laguillaumie, Adeline Langlois, Benoît Libert, and Damien Stehlé. Lattice-based group signatures with logarithmic signature size. In Kazue Sako and Palash Sarkar, editors, *Advances in Cryptology - ASIACRYPT 2013 - 19th International Conference on the Theory and Application of Cryptology and Information Security, Bengaluru, India, December 1-5, 2013, Proceedings, Part II*, volume 8270 of *Lecture Notes in Computer Science*, pages 41–61. Springer, 2013.

[20] Benoît Libert and Marc Joye. Group signatures with message-dependent opening in the standard model. In Josh Benaloh, editor, *Topics in Cryptology - CT-RSA 2014 - The*

Cryptographer's Track at the RSA Conference 2014, San Francisco, CA, USA, February 25-28, 2014. Proceedings, volume 8366 of *Lecture Notes in Computer Science*, pages 286–306. Springer, 2014.

[21] Benoît Libert, Thomas Peters, and Moti Yung. Short group signatures via structure-preserving signatures: Standard model security from simple assumptions. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part II*, volume 9216 of *Lecture Notes in Computer Science*, pages 296–316. Springer, 2015.

[22] Benoît Libert and Damien Vergnaud. Group signatures with verifier-local revocation and backward unlinkability in the standard model. In Juan A. Garay, Atsuko Miyaji, and Akira Otsuka, editors, *Cryptology and Network Security, 8th International Conference, CANS 2009, Kanazawa, Japan, December 12-14, 2009. Proceedings*, volume 5888 of *Lecture Notes in Computer Science*, pages 498–517. Springer, 2009.

[23] Benoît Libert and Moti Yung. Fully forward-secure group signatures. In David Naccache, editor, *Cryptography and Security: From Theory to Applications - Essays Dedicated to Jean-Jacques Quisquater on the Occasion of His 65th Birthday*, volume 6805 of *Lecture Notes in Computer Science*, pages 156–184. Springer, 2012.

[24] Karsten Nohl, David Evans, Starbug, and Henryk Plötz. Reverse-Engineering a Cryptographic RFID Tag. In *USENIX Security Symposium, San Jose, CA, USA, 31 July, 2008, Proceedings*, pages 1–9. USENIX, 2008.

[25] Thomas Unterluggauer and Erich Wenger. Efficient pairings and ECC for embedded systems. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems - CHES 2014 - 16th International Workshop, Busan, South Korea, September 23-26, 2014. Proceedings*, volume 8731 of *Lecture Notes in Computer Science*, pages 298–315. Springer, 2014.