Privacy Preserving Payments on Computational RFID Devices with Application in Intelligent Transportation Systems

Gesine Hinterwälder[1], Christof Paar[1,2], 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

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.

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 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

$\mathcal{U}$		$\mathcal{B}$
		$w \in_{\mathcal{R}} \mathbb{Z}_q$
		$a \leftarrow g^w$
$s \in_{\mathcal{R}} \mathbb{Z}_q^*$	$\xleftarrow{a,b}$	$b \leftarrow (Ig_2)^w$
$A \leftarrow (Ig_2)^s$		
$z' \leftarrow z^s$		
$x_1, x_2, u, v \in_{\mathcal{R}} \mathbb{Z}_q$		
$B \leftarrow g_1^{x_1} g_2^{x_2}$		
$a' \leftarrow a^u g^v$		
$b' \leftarrow b^{su} A^v$		
$c' \leftarrow \mathcal{H}(A, B, z', a', b')$		
$c \leftarrow c'/u \mod q$	$\xrightarrow{c}$	
	$\xleftarrow{r}$	$r \leftarrow cx + w \mod q$
$g^r \overset{?}{=} h^c a$		
$(Ig_2)^r \overset{?}{=} z^c b$		
$r' \leftarrow ru + v \mod q$		

Table 2. Brands' spending protocol

$\mathcal{U}$		$\mathcal{S}$
	$\xrightarrow{A,B,sign(A,B)}$	$A \overset{?}{\neq} 1$
$r_1 \leftarrow d(u_1 s) + x_1 \mod q$	$\xleftarrow{d}$	$d \leftarrow \mathcal{H}_0(A, B, I_{\mathcal{S}}, \text{date/time})$
$r_2 \leftarrow ds + x_2 \mod q$	$\xrightarrow{(r_1,r_2)}$	
		Verify sign(A, B)
		$g_1^{r_1} g_2^{r_2} \overset{?}{=} A^d B$

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**M**+2**S**+7**A** instead of 12**M**+4**S**+7**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]:

$$\begin{aligned}\mathcal{H} :& G_q \times G_q \times G_q \times G_q \times G_q \rightarrow \mathbb{Z}_q^*\\ \mathcal{H}_0 :& G_q \times G_q \times \text{SHOP-ID} \times \text{DATE/TIME} \rightarrow \mathbb{Z}_q\end{aligned}$$

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 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.

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

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.

6 Acknowledgements

We thank Andy Rupp and Andrés Molina-Markham for their input to this work.

References

1. L. Batina, J.-H. Hoepman, B. Jacobs, W. Mostowski, and P. Vullers. Developing Efficient Blinded Attribute Certificates on Smart Cards via Pairings. In *Smart Card Research and Advanced Application*, volume 6035, pages 209–222, 2010.
2. P. Bichsel, J. Camenisch, T. Groß, and V. Shoup. Anonymous credentials on a standard java card. In *ACM conference on Computer and communications security*, pages 600–610, 2009.
3. J. W. Bos, M. E. Kaihara, T. Kleinjung, A. K. Lenstra, and P. L. Montgomery. On the Security of 1024-bit RSA and 160-bit Elliptic Curve Cryptography. *IACR Cryptology ePrint Archive*, 2009:389, 2009.
4. S. Brands. Untraceable Off-line Cash in Wallets with Observers (Extended Abstract). In *Proceedings of the 13th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '93, pages 302–318, 1994.
5. E. Clemente-Cuervo, F. Rodríguez-Henríquez, D. O. Arroyo, and L. Ertaul. A PDA Implementation of an Off-line e-Cash Protocol. In *Security and Management*, pages 452–458, 2007.

6. B. Cohen. AES-hash. 2001. `http://csrc.nist.gov/groups/ST/toolkit/BCM/documents/proposedmodes/aes-hash/aeshash.pdf`.
7. J.-S. Coron. Resistance against Differential Power Analysis for Elliptic Curve Cryptosystems. In *Workshop on Cryptographic Hardware and Embedded Systems*, CHES '99, pages 292–302, 1999.
8. J. Daemen and V. Rijmen. The Rijndael Block Cipher. 1999. `http://ftp.csci.csusb.edu/ykarant/courses/w2005/csci531/papers/Rijndael.pdf`.
9. N. Gura, A. Pate, A. Wander, H. Eberle, and S. C. Shantz. Comparing Elliptic Curve Cryptography and RSA on 8-bit CPUs. In *Cryptographic Hardware and Embedded Systems - CHES 2004*, volume 3146/2004, pages 119–132, 2004.
10. D. Hankerson, A. J. Menezes, and S. Vanstone. *Guide to Elliptic Curve Cryptography*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2003.
11. T. S. Heydt-Benjamin, H.-J. Chae, B. Defend, and K. Fu. Privacy for Public Transportation. In *Privacy Enhancing Technologies*, pages 1–19, 2006.
12. T. I. Incorporated. MSP430x2xx Family User's Guide (Rev. H). 2011.
13. T. I. Incorporated. MSP43F261x Mixed Signal Microcontroller. 2011.
14. N. Meloni. New Point Addition Formulae for ECC Applications. In *Arithmetic of Finite Fields*, volume 4547, pages 189–201. 2007.
15. A. Menezes, P. C. van Oorschot, and S. A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.
16. P. L. Montgomery. Speeding the Pollard and Elliptic Curve Methods of Factorization. In *Mathematics of Computation*, pages 243–264, 1987.
17. C. Pendl, M. Pelnar, and M. Hutter. Elliptic Curve Cryptography on the WISP UHF RFID Tag. In *Workshop on RFID Security - RFIDsec*, volume 7055, pages 32 – 47. Springer, 2011.
18. C. 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. M. Rivain. Fast and Regular Algorithms for Scalar Multiplication over Elliptic Curves. *IACR Cryptology ePrint Archive*, 2011:338, 2011.
20. A.-R. Sadeghi, I. Visconti, and C. Wachsmann. User Privacy in Transport Systems Based on RFID E-Tickets. In *PiLBA*, 2008.
21. H. Zhang, J. Gummeson, B. Ransford, and K. Fu. Moo: A Batteryless Computational RFID and Sensing Platform. 2011. `https://web.cs.umass.edu/publication/docs/2011/UM-CS-2011-020.pdf`.