Pay As You Go

Foteini Baldimtsi[1], Gesine Hinterwälder[2], Andy Rupp[3], Anna Lysyanskaya[1], Christof Paar[2,4], Wayne P. Burleson[2]

[1] Computer Science Department, Brown University
{foteini,anna}@cs.brown.edu
[2] Department of Electrical and Computer Engineering, University of Massachusetts
{hinterwalder,burleson}@ecs.umass.edu
[3] AGT Group (R&D) GmbH, Germany
arupp@agtgermany.com
[4] Horst Görtz Institute for IT Security, Ruhr-University Bochum, Germany
christof.paar@rub.de

Abstract

Until recently, the desire for user privacy in intelligent transportation systems seemed to be at odds with the cost constraints of payment tokens. There is need for low-cost payment devices that can be produced in massive quantities while sophisticated cryptographic techniques seem to be too computationally intensive to be used in such devices. Our Pay-as-you-Go approach will demonstrate that it is possible to obtain privacy on very low-cost tokens, while retaining the benefits of useful data collection. We explore privacy-preserving payment protocols suitable for use in public transportation and show ways to efficiently implement those protocols on potential payment devices. Pay-as-you-Go is a multi-disciplinary research project, funded by NSF, that includes a diverse team of computer scientists, electrical and computer engineers and transportation engineers from Brown University and the University of Massachusetts.

Keywords: Electronic-cash, transportation payment systems, RFID security and privacy, mobile payments, refunds, anonymous-credentials

1 Introduction

Consider a large metropolitan area such as New York City. Its public transportation system is what makes the city work. But it does not work for free: the millions of passengers it serves every day must pay for the ride. Let us take a look at the underlying payment systems.

The simplest, and oldest, payment system is with actual cash, tokens, or tickets. One of its advantages is that the passengers do not leave behind an electronic trail of their comings and goings. However, it also has severe limitations: physical payments require cashiers or customized payment booths or turnstiles; it is hard to adapt the system to variable pricing or collect statistics that lead to better service, etc.

As a result, pre-paid or monthly cards (those that need to be swiped, or sometimes contactless cards) such as MetroCards in NYC and Charlie Cards in Boston have replaced the systems based on physical tokens. Contactless devices have also made paying tolls easier: systems such as E-ZPass give drivers a device that automatically pays for their highway tolls as they drive by the toll booth.

These convenient systems introduce concerns as to the privacy of their customers. Essentially, one's MetroCard or Charlie Card is a persistent identifier, and the MTA in New York, or MBTA in Boston, has the ability to locate an individual in a large metropolis, which prompts concerns among privacy advocates [31].

Additionally, these devices do not necessarily offer security for the transportation authorities either — for example, the Charlie Card was shown vulnerable to forgery by 3 MIT students doing a class project [29].

Privacy is an especially challenging problem in this context since it not only spans cryptographic theory and many engineering fields but extends into public policy areas such as environmental justice policy and sociology issues such as "fair access to all". However, in order to enable a large-scale deployment and broad acceptance of such a payment system, adequate security and privacy mechanisms are an essential requirement. Indeed, current users of FasTrak, the electronic toll collection system of California, rank "more secure technology to prevent security and privacy issues" in the top three recommendations of a recent study [26].

One may argue that giving up one's privacy is a small price to pay for such important benefits as ease and convenience, not to mention the fact that the information collected can facilitate advanced traveler information dissemination, traffic management, travel time estimation, emergency management, congestion pricing and carbon emissions control, and environmental justice assessments. However, perhaps one could get the best of both worlds: all the benefits of MetroCards without sacrificing privacy?

In theory, cryptographic techniques that make this possible exist. Electronic cash schemes (e-cash) have all the privacy benefits of actual physical cash. How can we implement them on constrained devices such as a MetroCard? How do we make them work with the same speed and convenience as non-privacy-preserving MetroCards? How do we still allow to collect the same useful information about traffic patterns, even while preserving the privacy of individuals?

This paper is about the Pay-As-You-Go (PAYG) project, which seeks to answer these questions. Our goal is to bridge the gap between theoretical constructions and practical implementation on RFID devices. Our starting point as far as crypto is concerned is state-of-the-art electronic cash schemes; we want to make them as efficient as possible, in part by customizing them to the scenario at hand. On the other end, our starting point is state-of-the-art RFID devices; we want to achieve highly efficient implementations of e-cash that would be appropriate for this scenario. Working from both ends of the problem, we will obtain a solution that offers speed and convenience on the one hand, and cryptographic guarantees of security and privacy on the other. By incorporating additional cryptographic techniques, we can derive additional benefits, such as variable pricing and privacy-preserving data collection.

Our results so far are promising. On the crypto end, we want (1) provable security and (2) high efficiency. The most efficient electronic cash scheme proposed in the literature is due to Brands [11]; unfortunately we have shown that known techniques do not allow to prove its security [4] making it inadequate for our purposes. Luckily, a scheme due to Abe [1] offers comparable efficiency and provable security at the same time. However, Abe's scheme doesn't allow the encoding of user's attributes (such as age, address, etc.) which is essential for the transportation setting. Encoding user's attributes in the coins allows us to implement additional features in our system such as variable pricing (e.g. reduced fare for senior customers) and privacy-preserving data collection. We are able to create a new scheme based on Abe's that allows the encoding of user's attributes in the coins withdrawn while being efficient and provably secure [3]. We further show how to make variable-priced schemes as efficient as fixed-priced schemes [28] by using the idea of pre-payment with refunds. We give more details in Section 2.

On the implementation end, one challenge is imposed by the time constraints of transportation payment systems. To avoid congestion in front of turnstiles, a payment transaction should be completed in 200 to 300 ms. As we will see, this is a considerable challenge given the computational complexity of advanced payment protocols. The efficiency of withdrawing money, i.e., for charging a payment token, is less critical, but should also not take longer than a few seconds. Another set of challenges are related to the payment token itself. First, it should be based on inexpensive hardware due to the potentially very high volume and the need to replace payment cards frequently. Secondly, it should be able to communicate and work contactlessly and without battery. These two conditions are in conflict with the need to run very complex cryptographic operations. We demonstrate that through optimized implementation techniques it is

possible to realize advanced payment protocols even on RFID tokens with extremely limited computational resources. Such hardware platforms are close approximations of low-cost platforms that will be used in future payment tokens. Section 3 gives more details on the implementation techniques.

Thus, we are on the cusp of having RFID devices implementing cryptographically secure electronic cash. This opens up other application areas for which, up until now, cryptography was considered prohibitively inefficient.

1.1 Related Work

Heydt-Benjamin et al. [18] were the first to mention integrated transportation payment systems as an area of interest for cryptographic research and proposed the use of recent advances in anonymous credentials and e-cash systems for electronic payments in public transportation systems. Sadeghi, Visconti, and Wachsmann presented an efficient RFID-based e-ticket scheme for transit applications [30]. However, they assumed the existence of external trusted devices and their system did not protect privacy against a prying transportation authority but only against prying outsiders.

In 2009, Blass et al. [8] proposed an offline, pre-paid, "privacy-preserving" payment system for transit applications. The most remarkable feature of this approach is that the whole system solely relies on a 128-bit hash function (and lots of precomputed data: 18 TB stored at the backend and 1 GB per reader). However, again, a user's privacy in their system was not protected from the entity responsible for issuing transportation tokens.

Popa, Balakrishnan, and Blumberg [24] proposed a privacy-preserving payment system for location-based services such as mileage-based toll collection, insurance pricing based on driver behavior, etc. In 2010, Balasch et al. [2] presented PrETP which is a location-private toll-collecting system where the on-board payment units can prove that they use genuine data and perform correct operations while disclosing the minimum amount of location data. In 2011 Meiklejohn et al. [19] presented another location-private system for toll collection that was based on PrETP but fixed a serious problem that PrETP was vulnerable to colluding free-loading users. Their system, called Milo, doesn't reveal any information, even when drivers misbehave and collude.

All these schemes were developed for a scenario where users subscribe for a service and pay by the end of a billing period. In the transit scenario, we cannot assume that each user has access to a trusted PC to settle accounts: an untrusted vending machine is more realistic. PAYG therefore choses the e-cash route achieving real-time, secure payments that also guarantee location privacy.

2 Electronic Cash

E-cash is an electronic version of physical cash, which provides at least as much anonymity and security as physical cash transactions. A typical e-cash scheme has three types of players: the Bank $\mathcal{B}$, Users $\mathcal{U}$ and Merchants $\mathcal{M}$ who are involved in the following procedures: (a) *setup* where the public and private keys of $\mathcal{B}, \mathcal{U}$ and $\mathcal{M}$ are established; (b) *account opening* during which $\mathcal{U}$s and $\mathcal{M}$s register at the bank; (c) *withdrawal* where users obtain e-coins from the bank; (d) *spending* where users submit coins to merchants in exchange for goods/services; (e) *deposit* where merchants deposit coins back to their bank accounts; (f) other protocols to identify malicious behavior.

In the electronic world, coins are simply digital data with unique serial numbers. So, in order to prevent users from creating coins by themselves we require the bank to digitally sign the coins. Regular digital signatures are not good enough for the e-cash setting. The problem is that anonymity is violated since the bank is able to recognize the signatures that were issued during withdrawal when the coins are deposited and thus trace how the users spend their coins. To circumvent this problem, we use a special type of digital

signatures called *blind signatures*. The idea is that the user can obtain the bank's signature on some message without the bank getting any information about the message being signed. This seems to solve the problem of users creating coins themselves, but what if a user simply tries to spend the same coin more than once? This could be easily solved, if we requested the bank to be on-line all the time (so that it could simply check if a coin being spend was used before). However, this doesn't seem very practical especially for the transportation setting. It turns out that security can be provided even in the off-line scenario. In that case we need to somehow encode the user's identity in the coins he withdraws so that if he tries to double spend a coin, the bank will be able to identify him.

In 1982, Chaum [14] came up with the idea of an e-cash system that allows anonymous, unlikable payments, secure against double spending in the off-line setting. Following Chaum's paradigm many schemes were proposed [11, 21, 17, 15, 6, 12, 13]. The scheme due to Brands [11], which is currently implemented by Microsoft in their U-Prove project [22], has the most efficient protocol when it comes to spending an e-coin and initially seemed like a good candidate scheme for the PAYG project. However, no proof of security has been given for his scheme. First, the best double-spending guarantee known, suggested by Cramer et al. [16], holds under the knowledge of exponent assumption (KEA) which is a very strong assumption (we do not want to base the security of cryptographic schemes on such strong assumptions). Second and more importantly, we showed that Brands blind signature (at the heart of his e-cash scheme) cannot be proven *unforgeable* using any of the currently known techniques [4]. Our result is even more general: we essentially ruled out all known approaches to proving security of a broader class of blind signatures ("generalized blind schnorr signatures") in the random oracle model.

We were able to resolve the second issue by providing a modification of Brands blind signature, which we prove to be unforgeable. The idea was to use Pointcheval and Stern's [23] suggestion and associate more than one secret keys to the same public key. So, in our modification of Brands, the signer (Bank) has a public key of the form $H = G_1^{w_1} G_2^{w_2}$ where w_1, w_2 the secret keys (refer to our paper [4] for the complete protocol and the security proof). However, similar to the original Brands [11], it is still an open problem whether provable guarantees against double-spending can be given for this modification.

As mentioned in the introduction we need schemes that provide (1) provable security and (2) high efficiency. Security in transactions plays an important role for the PAYG project and we are going to rule out any construction that doesn't provide a formal cryptographic proof of security (such as Brands). The scheme due to Abe [1] is provably secure and efficient enough for the PAYG requirements. It's drawback however, is that it doesn't allow the encoding of user's attributes in the coins/tokens withdrawn. This means that features like variable pricing and privacy-preserving data collection cannot be implemented with it. We gave a new e-cash scheme, based on the one due to Abe, which is provably secure, efficient and at the same time allows the encoding of user's attributes [3]. In our new e-cash the user commits to a set of attributes during the account opening phase, those attributes are later encoded in the coins he withdraws from the bank and then, during the spending phase, the user can choose to reveal a subset of those attributes when needed.

In Table 1 we provide a comparison of the above mentioned e-cash schemes. Compact e-cash [12] is another famous scheme that satisfies all the security requirements, but is significantly more expensive during the withdrawal and more importantly the spending phase which makes it difficult to be used in the PAYG without some modifications.

Even very efficient e-cash schemes require fairly large storage space for coins, and the protocols for withdrawing or spending a coin are computationally expensive. Hence, it is beneficial to limit the amount of coins that a user has to spend to execute a payment, having to spend only a single coin to make a payment would be ideal. In transportation payment systems, this conflicts with the necessity of allowing flexible prices. Fares should not be flat but arbitrary (and adjustable) monetary amounts. Setting the denomination of a coin to be one cent certainly allows for flexible pricing but users would need plenty of them to pay for a trip. Setting the face value to two dollars reduces the number of required coins per trip but severely restricts the system of fares. If we wanted to do a tradeoff and have e-coins for different monetary values we

	Brands [11]		Brands modification [4]		Abe [1]		New E-cash [3]		Compact E-cash [12]			
									RSA		Pairings	
Efficiency	B/M[1]	U[2]	B/M	U	B/M	U	B/M	U	B/M	U	B/M	U
Withdrawal[3]	2	13	4	19	6	12	7	13	10	8	15	14p
Spend	7	0	9	0	11	1	11	1	NC[5]		5+6p	9+6p
E-coin size	6 elem.		9 elem.		9 elem.		9 elem.		~ 0		~ 0	
Blindness	✓		✓		✓		✓		✓		✓	
Unforgeability	✗?		✓		✓		✓		✓		✓	
DS[4] **security**	✗?		✗?		✓		✓		✓		✓	
Attributes	✓		✓		✗		✓		✓		✓	

[1] Bank/Merchant, [2] User
[3] In number of exponentiations, [4] Double Spending, [5] Non comparable

Table 1: Comparison of e-cash schemes

would need to deal with overpayments and change in a privacy-preserving way. This is especially difficult in transportation payment systems where usually bank and merchants are the same entity.

To circumvent this problem we designed a payment system, called P4R [28] (**P**rivacy-**P**reserving **P**re-**P**ayments with **R**efunds), that is based on the concept of pre-payments with refunds: users deposit money to obtain a bundle of coins. When the payment is less than the value of the coin, the user obtains a refund. The system allows to aggregate these refunds in a single token, thereby saving memory and increasing privacy. P4R can be constructed from any anonymous e-cash/credential scheme which can be modified in a way that coins/credentials can be shown twice without revealing the ID of a user (e.g., Brands' scheme, or our new e-cash scheme [3]). In the next section we give more details on how P4R works.

2.1 P4R Payment System

As mentioned above P4R is not a typical e-cash scheme but a pre-payment scheme with refunds. It is composed of three subsystems the Trip Authorization Token (TAT), the Refund Calculation Token (RCT), and the Refund Token (RT) system. The TAT system is an offline system. Here ticket vending machines play the role of the "bank" issuing TATs and (offline) readers at the entry turnstiles play the role of a "merchant" where tokens can be spent. The RT system is an online system. Here roles are reversed compared to the TAT system, i.e., readers at the exit turnstiles issue refunds and the (online) vending machines receive the tokens and disburse the users. Some details of the different subsystems are given below.

A TAT (aka ticket) is a credential that authorizes a user to make exactly one trip in the transportation system. A user initially makes a deposit at a vending machine to obtain a number of TATs where the cost of a TAT equals the cost of the most expensive trip. Of course, to reduce the deposit for a TAT it is also possible to have different types of TATs for different sections or zones of the transportation system. The withdrawal of a TAT is done in a blind fashion such that a later use cannot be linked. The ID of a user (e.g., a credit card number) is encoded in each TAT to prevent a repeated use for entering the transportation system (double-spending detection). At the beginning of a ride a user presents an unused TAT to the reader at the turnstile at which he wants to enter the system. If it is valid and the user can show (using a zero-knowledge proof) that he knows the ID encoded in the TAT, access to the transportation system is granted.

At the end of the trip when the user leaves the system the actual fare is determined at the exit turnstile. This is done as follows: When entering the system a user also receives an RCT (aka stamped ticket), which

contains a MAC on the TAT, the date and time, as well as on the ID of the reader. When he leaves the system he presents this RCT to the exit turnstile reader, which calculates the trip cost based on this information. To obtain a refund the user also provides a blinded version of his RT (blank RT tokens are available from the vending machines) to the reader. To prevent a user from re-using an RCT and thus claiming the same refund several times in a row, the idea is to bind an RCT to the TAT, which has just been used to enter the system, and force him to again prove the knowledge of the ID encoded into this TAT when he leaves the system. An RT is re-used to add up several refunds instead of having a separate RT per refund. This saves memory and increases the privacy of a user. At some point the user may decide to cash the collected refund or to buy new TATs using this money. To do this, the user presents his RT to the vending machine, which redeems the RT, if this token is not already marked as cashed in the central database. This refund subsystem can be implemented based on a new variant of Boneh-Lynn-Shacham signatures [9] or a variant of RSA signatures [27].

As for security, we can show that in P4R it is infeasible for malicious users to receive reimbursements, which exceed the overall deposit for TATs minus the overall fare of their trips. Note that this also covers fare evasion. In other words, the transportation authority does not lose money. With respect to privacy, we can show that any two users obtaining the same total refund amount cannot be distinguished. Furthermore, we argue that usually many sequences of trips should result in the same refund sum: The number of sequences actually equals the number of *integer partitions* of the refund sum. Hence, we are interested in the number $p_S(n)$ of partitions of integers n, where parts are restricted to come from a certain set S.

However, one should also be aware of the limits of this approach: In practice, we cannot guarantee that during a certain period of time many such sequences actually appear in the records of the transportation authority. For instance, it could happen that since the issue date of the refund token nobody but the owner of the token did trips that resulted in a particular refund amount (though in theory many sequences lead to this amount). Moreover, there might be a lot of different sequences of single trips leading to the same refund sum but one trip in all these sequences is the same. In this case, we know that a user redeeming this refund amount must have taken exactly this trip. Hence, the exact level of location privacy provided by our system depends on the frequency of individual trips and the set of single refund values, i.e., characteristics of the transportation system and user behavior. Nevertheless, we believe that for real-world transportation systems the limits described above are no real issues. In fact, a transportation authority could publish the set of possible trips and the corresponding refund values, which would allow to (partly) check the existence of such problems.

In summary, we ruled out Brands' e-cash, because of insufficient provable security guarantees [4]. We gave the first provably secure construction of e-cash with attributes, whose efficiency is comparable to that of Brands' e-cash [3], and showed how to construct P4R from e-cash with attributes [28].

3 Implementation

For the implementation, the PAYG project focuses on the user side, i.e. the payment devices. The turnstiles can be equipped with powerful hardware or connected to a back-end system, and hence do not represent an obstacle for the efficient execution of complex algorithms.

To keep the overall payment execution time low, payment devices that do not have to be physically connected to a machine are preferable, lending itself to the use of RF-communicating devices. As such, modern smartphones support NFC-technology, a standard that supports short range communication via RF-signaling. Additionally NFC-phones are equipped with powerful CPUs. However, it cannot be assumed that every customer possesses an NFC-capable smartphone. Furthermore, relying solely on battery powered devices is not desirable for a host of reasons, as for example cost and lifetime. Ideally the transportation authority would provide user devices for free or at least at a very low price. Passive RFID devices are the

emerging standard for low-cost tokens and are highly attractive for this application domain. However, they are also severely constrained in terms of computational capabilities and memory. Contactless smartcards allow short range communication, conform to NFC-technology, which enables the easy design of a hybrid system. Nevertheless for some setups, medium range communication, say over a distance of at least 1 m, is highly attractive. For example it can enable multimodal payment systems, including payments for toll roads. Pay-as-you-Go does not limit its investigations to one of those devices, but investigates advantages and disadvantages of the different device types.

Privacy-preserving payment protocols are based on public key cryptography, which requires the execution of computationally expensive algorithms. This conflicts with the need for very inexpensive payment devices that use only limited power. Modern contact-less smartcards provide hardware accelerators to support certain standard cryptographic protocols. But the usage of those hardware accelerators is stiff ([5], [7]), and an implementation of e-cash schemes on such devices leads to very long execution times.

A medium range passively powered RFID device that can be freely programmed is the Moo, a computational RFID chip designed at the University of Massachusetts Amherst [33]. Even though it is not a commercial product, it is a good representative for an extremely low-cost, medium-range payment token, which could be used in future transportation payment systems. This RFID tag is equipped with an MSP430F2618 microcontroller, a low-power microcontroller developed by Texas Instruments. The microcontroller uses a 16-bit RISC architecture. It has 8 KB RAM and 116 KB flash memory. 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. We implemented **Brands** and **Brands modif** from Table 1 for this device.

We base the implementation of both schemes on elliptic curve cryptography (ECC). ECC is especially suitable in very constrained environments. In comparison to other public-key schemes, as for example discrete logarithm (DL) based schemes, ECC requires much shorter operand lengths to achieve the same level of security. As such, a 160-bit curve offers the same level of security as a 1024-bit DL-based scheme. Nevertheless, even though ECC has a lower bit complexity than DL, sophisticated implementation techniques are required in order to achieve acceptable execution times of the payment schemes on devices as the Moo. We conclude from [10] that a 160-bit elliptic curve presents sufficient security for a micro-payment system. We chose to implement the schemes based on the curve that is specified as secp160r1 in [25]. This curve is based on a special prime that allows for a very efficient implementation of the reduction step in the underlying prime field.

The most time critical operation, and hence the function that dominates the execution time of the different protocols, is the point multiplication, which is the equivalent to an exponentiation in DL-based schemes. We used the Montgomery algorithm for point multiplication [20] to implement this function. To give an idea of the computational complexity: The Montgomery ladder [20] requires 160 point additions and 160 point doublings, which when using Jacobian coordinates need $12M + 4S$ (where M stands for modular multiplication and S stands for modular squaring) and $4M + 6S$, respectively. On a 16-bit microcontroller one element in the underlying 160-bit finite field is represented as an array of ten integers. Thus using operand-scanning form, a modular multiplication requires 100 and a modular squaring 55 integer multiplications. Hence $344\,000$ integer multiplications are necessary to execute a single point multiplication. Especially useful for this implementation is the multiply-and-accumulate instruction of the MSP430, which supports a 16x16-bit unsigned integer multiplication and accumulation of the 32-bit results [32]. In our implementation the execution of one point multiplication takes 6.8 million cycles.

Table 2 summarizes the token-side execution times of the payment scheme for the withdrawal and spending phase. The user authentication step during withdrawal, where the user proves ownership of his account, is not included. This authentication step has to be executed once, before one or several coins can be withdrawn. The execution times are presented for two frequencies of the token CPU, 4 Mhz and 16 Mhz. The microcontroller that is integrated on the Moo operates at a frequency of 4 Mhz. However, it can be operated at a maximum frequency of 16 Mhz.

We note that 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 in the range of 10 ms are fully sufficient 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. There are approaches to improving the performance of the payment token. An interesting one is that during charging the payment device could be connected to the charging station, which supplies the passive tag with sufficient energy to clock it at higher frequencies, or to power additional hardware accelerators.

	Brands	**Brands Modif.**
Cycle count Withdrawal of a coin[1]	83	125
Cycle count Spending of a coin[1]	0.052	0.053
Execution time Withdrawal of a coin @16 MHz[3]	5	8
Execution time Spending of a coin @16 MHz[3]	0.0033	0.0033
Execution time Withdrawal of a coin @4 MHz[3]	21	31
Execution time Spending of a coin @4 MHz[3]	0.013	0.013
Code Size[2]	16570	17030

[1] in million cycles, [2] in bytes
[3] in seconds

Table 2: Timing results of user side protocol implementation

4 Conclusions and Future Work

The Pay-as-you-Go project has three main goals: (1) develop novel cryptographic algorithms for the transportation setting whose efficiency is suitable, and that allow for collecting useful data and yet guarantee privacy at the same time, (2) explore emerging hardware architectures and software for performing modern cryptographic operations on low-cost devices, and (3) test human factors and performance of privacy-preserving payment systems under realistic conditions of emerging transportation payment systems that must balance security and privacy with cost with usability. Our results so far include ruling out Brands e-cash scheme due to insufficient security guarantees, constructing new efficient and provable secure algorithms for the transportation setting that allow the encoding of user's attributes and can be used to construct a pre-payment with refunds (P4R) and implementing part of them. The next steps of the project include, in the crypto-end, the proposal of some security guarantees for the modified Brands protocol against double spending and exploring possible constructions that also allow private data collection. On the implementation end, the next steps will be the implementation of our new e-cash scheme with P4R as well as comparing different potential contactless payment devices.

5 Aknowledgements

This work was supported by NSF grant 0964379.

References

[1] M. Abe. A secure three-move blind signature scheme for polynomially many signatures. In *EUROCRYPT'01*.

[2] J. Balasch, A. Rial, C. Troncoso, B. Preneel, I. Verbauwhede, and C. Geuens. Pretp: privacy-preserving electronic toll pricing. USENIX Security'10.

[3] F. Baldimtsi and A. Lysyanskaya. Anonymous credentials light. Cryptology ePrint Archive, Report 2012/352, 2012. http://eprint.iacr.org/.

[4] F. Baldimtsi and A. Lysyanskaya. On the security of one-witness blind signature schemes. Cryptology ePrint Archive, Report 2012/197, 2012. http://eprint.iacr.org/.

[5] L. Batina, J.-H. Hoepman, B. Jacobs, W. Mostowski, and P. Vullers. Developing efficient blinded attribute certificates on smart cards via pairings. In *CARDIS'10*.

[6] M. Belenkiy, M. Chase, M. Kohlweiss, and A. Lysyanskaya. Compact e-cash and simulatable vrfs revisited.

[7] P. Bichsel, J. Camenisch, T. Groß, and V. Shoup. Anonymous credentials on a standard java card. In *ACM CCS'09*.

[8] E.-O. Blass, A. Kurmus, R. Molva, and T. Strufe. Psp: private and secure payment with rfid. In *WPES'09*.

[9] D. Boneh, B. Lynn, and H. Shacham. Short signatures from the weil pairing. *J. Cryptology*, 17(4):297–319, 2004.

[10] 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. 2009.

[11] S. Brands. Untraceable off-line cash in wallets with observers. In *CRYPTO'93*.

[12] J. Camenisch, S. Hohenberger, and A. Lysyanskaya. Compact e-cash. In *EUROCRYPT'05*.

[13] A. Chan, Y. Frankel, and Y. Tsiounis. Easy come - easy go divisible cash. In *EUROCRYPT'98*.

[14] D. Chaum. Blind signatures for untraceable payment. In *Crypto'82*.

[15] D. Chaum, A. Fiat, and M. Naor. Untraceable electronic cash. In *CRYPTO '88*.

[16] R. Cramer, I. Damgard, and J. B. Nielsen. On electronic payment systems. In *http://www.daimi.au.dk/ ivan/ecash.pdf*, 2010.

[17] N. Ferguson. Single term off-line coins. In *EUROCRYPT '93*.

[18] T. S. Heydt-benjamin, H. jin Chae, B. Defend, and K. Fu. Privacy for public transportation. In *PETs 2006*.

[19] S. Meiklejohn, K. Mowery, S. Checkoway, and H. Shacham. The phantom tollbooth: privacy-preserving electronic toll collection in the presence of driver collusion. In *USENIX 2011*.

[20] P. L. Montgomery. Speeding the pollard and elliptic curve methods of factorization. In *Mathematics of Computation*, 1987.

[21] T. Okamoto and K. Ohta. Universal electronic cash. In *CRYPTO'91*.

[22] C. Paquin. U-prove cryptographic specification v1.1. In *Microsoft Technical Report*, 2011.

[23] D. Pointcheval and J. Stern. Provably secure blind signature schemes. In *Asiacrypt'96*.

[24] R. A. Popa, H. Balakrishnan, and A. J. Blumberg. Vpriv: protecting privacy in location-based vehicular services. In *USENIX security symposium*, 2009.

[25] C. Research. Sec 2: Recommended elliptic curve domain parameters. 2000.

[26] P. F. Riley. The tolls of privacy: An underestimated roadblock for electronic toll collection usage. *Computer Law & Security Review*, 24(6):521 – 528, 2008.

[27] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

[28] A. Rupp, M. Fischlin, and C. Paar. Pre-payments with refunds: A new cryptographic approach to privacy-preserving payments for transportation systems. In *Manuscript*, 2012.

[29] R. Russel, Z. Anderson, and A. Chiesa. Anatomy of a subway hack. In *DefCon*, 2008.

[30] A.-R. Sadeghi, I. Visconti, and C. Wachsmann. User privacy in transport systems based on rfid e-tickets. In *PiLBA*, 2008.

[31] B. Saderson. E-z pass could take toll on right to privacy. In *New York Post*, 1996.

[32] Texas Instruments Incorporated. MSP430x2xx Family User's Guide (Rev. H). 2011.

[33] H. Zhang, J. Gummeson, B. Ransford, and K. Fu. Moo: A batteryless computational rfid and sensing platform. 2011.