Electronic Payments of Small Amounts

Torben P. Pedersen*
Computer Science Department, Aarhus University

Abstract

This note considers the application of electronic cash to transactions in which many small amounts must be paid to the same payee and in which it is not possible to just pay the total amount afterwards. The most notable example of such a transaction is payment for phone calls. If currently published electronic cash systems are used and a full payment protocol is executed for each of the small amounts, the overall complexity of the system will be prohibitively large (time, storage and communication). This note describes how such payments can be handled in a wide class of payment systems. The solution is very easy to adapt as it only influences the payment and deposit transactions involving such payments. Furthermore, making and verifying each small payment requires very little computation and communication, and the total complexity of both transactions is comparable to that of a payment of a fixed amount.

1 Introduction

The introduction of public key crypto-systems and digital signatures ([DH76] and [RSA78]) has led to the construction of many different types of payment systems. During payment at least one and often more digital signatures must be created and verified. This has the advantage that it can often be argued (although not always formally proved) that the payment system is as secure as the digital signature system. However, it also has the disadvantage that the efficiency of the payment system depends on the efficiency of the signature scheme. In particular, the time to perform a payment transaction is closely

*e-mail: tppedersen@daimi.aau.dk

related to that of verifying a signature and the storage requirements depend on the length of signatures.

Since the computations on the users side must be performed by small devices with relatively little computation power such as smart cards or electronic wallets (see [EG84] and [Cha92] for two different types of wallets), it is important that the users part of all transactions can be done very efficiently.

Although recent research has improved the efficiency of (privacy protecting) electronic payment systems, they are still too inefficient in certain applications. One such application is phone calls. Here the total amount to be paid is composed of many payments of small amounts, and each of these small amounts must be paid in real time upon receipt of a *tick*. Depending on the type of phone call (local, long distance, etc.) these ticks may come with relatively short time interval. The timing requirements imposed by this cannot possibly be met by publicly suggested payment systems if each tick is paid for by executing a payment transaction. Furthermore, such an approach would require quite a lot of storage at the recipient side increasing linearly in the number of ticks during the phone call. Although this may not be a serious problem considering todays cheap storage media, it does increase the cost of the payment system and it implies a large communication overhead. Other settings where similar payments could be useful include parking and access to electronic services. We shall in general denote such payments *tick payments*.

This note describes a method for adding tick payments to systems in which the payer during payment makes (something like) a signature on a message describing the recipient and the amount to be paid. The method is most easily explained as an application of Lamport's password scheme (see [Lam81]) to encode amounts in payments. An earlier application of repeated computations of one-way functions is described in [Mer90] and attributed to Winternitz (1979). In relation to payment systems a similar idea is used in [BC90] to encode amounts. The encoding suggested in the following is a simplification of that in [BC90] specifically designed for tick payments and can be used in many different systems.

The proposed solution is very easily applicable since only payment and deposit transactions involving tick payments are affected. A tick payment initially requires the same as a normal payment. Then the payment for the i'th tick requires at most $T-i$ computations of an easily computable function. Verification of each tick only requires one computation of this function. After the required number of ticks have been paid only a few hundred bits more than in a normal payment must be stored (the exact number depends on the actual choice of the function mentioned above).

The next section describes the general payment system considered and relates it to other proposed systems. Section 3 then presents the solution, and Section 4 describes a

few simple variations.

2 Electronic Payment Systems

2.1 The General System

We consider an electronic payment system in which an execution of the payment protocol can be described by a triple $(v, m, \sigma(m))$. Here $m \in \{0,1\}^*$ and $\sigma(m)$ are special messages, and v denote all other messages exchanged during the payment. The message, m, must describe the amount to be paid and it may contain additional information such as a description of the recipient and some bits chosen at random by the recipient. No part of v may depend on the amount paid. In most situations $\sigma(m)$ is a signature corresponding to a public key, which is either certified as part of v or can be recognised by other means. The verification of such a payment involves the verification that m contains the required information and that $\sigma(m)$ is correct with respect to m and v. The assumption that $m \in \{0,1\}^*$ is reasonable since a hash value of m and not m itself is normally used to compute $\sigma(m)$.

In order to be credited the received amount the payee must show $(m, \sigma(m))$ and possibly v to her bank. This can either be done during the payment transaction or in a later deposit transaction. Thus no assumption is made whether the system is on- or off-line. Furthermore, such payment systems can be used in both pre- and post-paid systems. Examples of such schemes are

- Cheque-like systems, where the payer signs a message transferring the signed amount from the payer to the payee (see [EG84] for such an off-line system).
- Anonymous cheques with counters. In such systems the payer gets during withdrawal a number of cheques and a permission to spend a certain amount (by setting a counter). During payment the payer fills out the cheque, and the payee may be credited the amount. In pre-paid systems this can be done anonymously, i.e., without revealing the anonymity of the payer. See [BC90, BBC+94, Bra95] for examples of such systems.

The security of payment systems following this model depends on the infeasibility of creating a false pair $(m, \sigma(m))$ which will be accepted by the bank. This would enable the receiver to get extra money, and in pre-paid systems it would enable a payer to spend money that he didn't withdraw.

2.2 Different Off-line Systems

Other types of off-line electronic payment systems have been proposed in the literature. In an electronic coin system, the user gets during withdrawal a number of electronic coins with fixed denominations (guaranteed by a signature from the issuer). During payment the user must spend a number of coins whose values add up to the required amount (e.g., see [CFN90, Fer93, Bra94]). As this often requires a large number of coins in a payment, leading to inefficiency with respect to time, storage and communication, electronic cheques with refund were proposed in [CFN90]. Here, the user obtains (buys) a number of cheques during withdrawal. Each cheque has an upper limit and can be spent for any amount up to this limit. A possibly remaining amount of a cheque can later be refunded. This is obviously more flexible than coin based systems, but still less practical than encoding amounts as described above since the user must contact the bank in order to convert "refundable" money to "spendable" money.

Another proposal, in [OO92], is to use so called divisible coins. Here the user can split a coin arbitrarily (i.e., this is much like a cheque with refund, where the refund can be spend). These systems offer more flexibility than cheques with refund, but they have not been obtained with the same level of privacy protection and the proposed systems seem to be too inefficient for practical purposes.

In all these pre-paid systems a tamper resistant device (trusted by the issuer) is used to prevent that the user spends more money than allowed. Furthermore, even in payment systems protecting the anonymity of the user it is possible to identify persons who break this tamper resistant device and spend too much money. Very briefly, the price we have to pay for the greater flexibility implied by including the amount in m is less fall-back security when the tamper resistant protection is passed.

2.3 Limitation of the Systems

As noted above the enhanced flexibility is the main reason for encoding the amount in m. Furthermore, such a scheme can be relatively efficient since its complexity depends on the complexity of creating and verifying $\sigma(m)$.

However, in the application to tick payments this is not sufficient, because if a full payment is done for each tick then

- The user has to create and the payee has to verify and store a large number of pairs $(m, \sigma(m))$.
- The time needed to make and verify a payment may exceed the time interval allowed.

Thus the main advantage of these systems vanishes when it comes to tick payments.

3 Adding Tick Payments

In the following we shall denote the security parameter of the payment scheme by k. It will be assumed that T is polynomial in k. Another security parameter $n = n(k)$ will depend on k in such a way that n and k are polynomially related (e.g., $n = k^c$ for some constant $c > 0$).

3.1 The Idea

This section shows how a payment system as described above can be adapted to handle tick payments.

Let f be a length preserving function $\{0,1\}^* \to \{0,1\}^*$ (i.e., f maps n-bit strings to n-bit strings for all $n \in \mathbb{N}$). Let f^i denote i evaluations of f for $i \in \mathbb{N}$. Thus, f^0 is the identity and $f^i = f \circ f^{i-1}$ for $i = 1, 2, \ldots$. For the moment f can be any efficiently computable function, but later we shall impose some restrictions on f for security reasons (one-way in a certain sense).

Let a payment system as described in Section 2.1 be given. In the following the system is extended to provide for efficient tick payments. This is done by providing a new way of encoding amounts in the message, m. In case of normal payments the encoding of amounts is unchanged. For tick payments the amount is encoded as follows. The message, m is computed as before except that it encodes the amount 0. Then a new message m' is computed as $m' = m \| T \| \alpha_0$, where $\|$ denotes concatenation, T is a possibly fixed parameter, and $\alpha_0 \in \{0,1\}^n$ is computed by the payer as $\alpha_0 = f^T(\alpha)$, where α is chosen at random in $\{0,1\}^n$ (recall that the message can be an arbitrary string of bits). In case of pre-paid systems, where a tamper resistant device manages a counter defining the amounts that may be spent, this device must do the computation of α_0 (see also Section 4). The payer then computes $\sigma(m')$ and sends $(m', \sigma(m'))$. This pair is verified as in a normal payment. Until now the payee has not received payment of any ticks. In order to get payment for the i'th tick ($i \geq 1$) the following takes place:

1. The payer sends $\alpha_i = f^{T-i}(\alpha)$ to the recipient.
2. The recipient verifies that $f(\alpha_i) = \alpha_{i-1}$ (α_{i-1} was received in the previous round, and α_0 was received in m').

This can continue for payment of at least T ticks. Thus an amount corresponding to t ticks is encoded as $\alpha_t \in \{0,1\}^n$ satisfying $f^t(\alpha_t) = \alpha_0$. Only α_0 (which is part of m) and the last received value, α_t, need to be remembered.

A simple variation of this scheme allows a combination of normal payments and tick payments, by just encoding a non-zero amount in m as described above. Furthermore, if one tick corresponds to one unit in the given currency, then u units can be paid in the i'th round by sending α_i satisfying $f^u(\alpha_i) = \alpha_{i-1}$.

3.2 Security of the Solution

It is now shown that if f is one-way in a certain sense then it is not feasible to obtain a larger amount than that received in the payment.

As mentioned in Section 2.1 the payment systems in question can be used in two ways. If the bank can identify the payer from the payment, it can always hold the payer responsible for the paid amount. In this case the main security concern is whether it is possible for a dishonest payee to change (increase) the encoded amount after the payer executed the payment transaction properly (i.e., following the given protocol).

In case the bank is not able to identify the payer, a tamper resistant unit trusted by the bank and used by the payer must decrement a counter corresponding to the amount in the payment. In this case, the main security concern is whether it is possible to encode a different amount than that used by the tamper resistant device, under the assumption that the tamper resistant device does its part of the protocol properly. Thus it can in both cases be assumed that α_0 is has been computed correctly.

The security of the extended system depends on the security of the given payment system and the properties of f. Consider the following property of a payment system as described in Section 2.1:

Definition 3.1 *Let a payment system with security parameter k be given. The system is said to be* unchangeable *if whenever a payee receives payment $(v, m, \sigma(m))$ from a device properly following the payment protocol she cannot deposit it as if she received $(v', m', \sigma_C(m'))$ with $m' \neq m$ except with negligible*[1] *probability in k. This probability is over all random choices during payment and deposit.*

Unchangeability ensures that amounts and other information encoded in m cannot be changed. The function f must be one-way on its iterates (see also [Lev85]):

[1] A function $g : \mathbb{N} \to \mathbb{R}$ is negligible if for all $c > 0$ and k sufficiently large $|g(k)| \leq k^{-c}$.

Definition 3.2 *Let $f : \{0,1\}^* \to \{0,1\}^*$ be a length preserving function. f is said to be one-way on T iterates if for every probabilistic polynomial time algorithm, A, the probability that A given $y = f^t(x)$ and t for $1 \leq t \leq T$ and a randomly chosen $x \in \{0,1\}^n$ outputs z such that $f(z) = y$ is negligible in n (the probability is over the choice of x and the random coins of A).*

The assumption that A also gets t as input is not important as it might as well guess this value whenever T is polynomial in k.

Given a length preserving function, f, consider the following game between two polynomially bounded parties A and B:

1. A chooses $x \in \{0,1\}^n$ at random, computes $f^T(x)$ and sends it to B.
2. For $i = T-1, T-2, \ldots$ until B decides to stop (B must do this at latest when $i = 1$): A sends $f^i(x)$ to B.
3. Assume $f^t(x)$ was the last value B received, where $t \in \{1, \ldots T\}$. Then B outputs $y \in \{0,1\}^n$.
4. B wins if $f(y) = f^t(x)$.

Lemma 3.3 *If f is one-way on T iterates (T is polynomial in n), then the probability that B wins is negligible in n and hence in k (the probability is over the choice of x and the random choices of B).*

Proof Assume that there is a polynomially bounded B, which is able to win with probability at least n^{-c} for some $c > 0$ and infinitely many values of n.

There is a t_0 such that the probability that B outputs a pre-image of $f^{t_0}(x)$ with non-negligible probability (over the choice of x and the random choices of B) is at least n^{-d} for some d and infinitely many values of n (since T is polynomial in n). Consider such n's.

Now consider the following machine for finding a pre-image of $y = f^t(x)$, given y and t:

1. First compute $z = f^{T-t}(x)$.
2. Simulate the above game.
3. If B stops for a value of i different from t, the algorithm outputs fails.

4. Otherwise if B stops for $i = t$ and outputs y_B, then the algorithm outputs y_B.

For $t = t_0$ the algorithm outputs y_B satisfying $f(y_B) = y$ with probability at least n^{-d}. This contradict the assumption that f is one-way on T iterates. □

From this technical lemma it is not hard to see that the encoding of amounts is secure if f is one-way on T-iterates. The details are given in the following proposition.

Proposition 3.4 *If the given payment system satisfies Definition 3.1, and if f is one-way on T iterates then the following holds except with negligible probability in k: If a payee receives the amount, a, in a payment transaction then, except with negligible probability in k, she can at most obtain an amount a during deposit.*

Proof A payment in the new system can either be a normal or a tick payment. In the first case the claim follows immediately from the assumption that the original system satisfies Definition 3.1.

Now assume that after N tick payment transactions a payee has been able to cash a payment received for t ticks as $t' > t$ ticks with non-negligible probability.

Then it is possible to construct a machine which wins in the above game, by simply simulating a payment system and using this payee to obtain the required pre-image. This machine for interacting with A in the above game works as follows:

1. Setup and simulate the entire payment system by selecting all keys and perform all protocols as described acting both as payer, payee and bank. Furthermore choose $l \in \{1, 2, \ldots, N\}$ at random.

2. Whenever a normal payment is performed follow the payment with the given dishonest payee).

3. If the payment is a tick payment and this is the l'th tick payment, then use the value $f^T(x)$ received from A as α_0. Furthermore, interact with A to get payment for the required number of ticks. Assume this payment is for t ticks.

4. If the payment is another tick payment then make this payment correctly (using the dishonest payee).

5. After all payments have been performed, the payee tries to deposit one of the received tick payments for more ticks than received.

If the payee is able to get money for $t' > t$ ticks for the l'th payment, then from the messages sent it is possible to find z such that $\alpha'_0 = f^{t'}(z)$, where α'_0 in contained in m' corresponding to the l'th payment. The machine outputs $u = f^{t'-t-1}(z)$.

Otherwise, the machine loses the game to A.

The dishonest payee will output u in the last step with non-negligible probability since l was chosen at random (and N is polynomial in k). By the property of unchangeability, $\alpha'_0 = \alpha_0$, except with negligible probability, and hence the machine will output a number u winning the game against A. By Lemma 3.3 this contradicts that f is one-way on T iterates. □

This proposition shows that the recipient cannot increase the received amount. Of course she can decrement it (corresponding to losing coins from a purse), but that should not be of her interest. Thus the extended system is not unchangeable, but still secure.

4 Variations

4.1 Possible Choices of f

In the previous section it was shown that if f is one-way on its iterates then the solution for tick payments is secure.

In a practical implementation f could be derived from a hash function $\{0,1\}^* \rightarrow \{0,1\}^n$ by restricting the input to n-bit strings (e.g., [Riv91, SHS92, BBB+93]). Such a function would be efficient and it often comes for free in the sense that it is needed anyways in the implementation of the payment system. If the hash function distributes its input sufficiently randomly it can be assumed to be one-way on its iterates for practical values of T.

Alternatively, a one-way *permutation* can be used. Such a function is one-way on T iterates whenever T is polynomial in k since y in Definition 3.2 is uniformly distributed. This again leads to the possibility of choosing f as a *trapdoor permutation*. If only (the tamper resistant device of) the payer knows the trapdoor information (i.e., the secret key needed to compute f^{-1}), then the payer can avoid T and have $\alpha_0 = \alpha$. If α_{i-1} has been used to pay for tick $(i-1)$, then $\alpha_i = f^{-1}(\alpha_{i-1})$ can be used to pay for the i'th tick. This has the advantage, that a priori no upper limit on the number of ticks must be determined. Furthermore, it could lead to better efficiency although this is not clear if f is the RSA function with small public exponent and T is at most a few hundreds (see also

[Lam81] for some suggestions for speeding up the repeated computations of the one-way function).

Finally, we mention that if $\{0,1\}^n$ is replaced by a group A_n (with group operation, say, $\odot$) a one-way *homomorphism*: $A_n \to A_n$ could be a good choice in wallets with observers (again the RSA function is a candidate). In that case observer and user could choose $\alpha \in A_n$ mutually at random as follows

1. Observer chooses $\beta \in A_n$ at random and sends a commitment to $\beta_0 = f^T(\beta)$ to the user.
2. The user chooses $\gamma \in A_n$ at random and sends $f^T(\gamma)$ to the observer.
3. The observer opens the commitment to $f^T(\beta)$
4. The observer and user compute $\alpha_0 = f^T(\beta) \odot f^T(\gamma)$ and include it in m'.

To pay for the i'th tick the observer sends $\beta_i = f^{T-i}(\beta)$ to the user and the user verifies that $f(\beta_i) = \beta_{i-1}$ and sends $\alpha_i = \beta_i \odot f^{T-i}(\gamma)$ to the recipient.

The advantage of this approach is that the observer can control the number of ticks paid and the user can blind the numbers sent for each tick.

4.2 Variations in On-line Systems

If the proposed method is applied to on-line systems, two alternatives are possible:

1. $(v, m', \sigma(m'))$ is verified on-line, but the payment for each tick is verified off-line.
2. $(v, m', \sigma(m'))$ as well as the payment for each tick is verified on-line.

Here the first alternative is most attractive as it obtains the advantage of on-line security while keeping the communication requirements independent of the number of ticks paid.

5 Conclusion

It has been shown that tick payments can be done very efficiently for payment systems where amounts are encoded in a special message during payment. It seems to be contradictory to the notion of coins to obtain the same property for coin based systems, but it is an interesting question if a similar hack can be used for cheque systems with refund as described in [CFN90].

6 Acknowledgements

I would like to thank all members in the protocol group of the ESPRIT project CAFE for discussions about this method. In particular, Michael Waidner for urging me on to work on this, Ronald Cramer for showing how an initial suggestion could be made independent of withdrawal, and both Ronald Cramer and Berry Schoenmakers for their cooperation on the extension to the wallet with observer setting.

References

[BBB+93] B. den Boer, J.-P. Boly, A. Bosselaers, J. Brandt, D. Chaum, I. Damgård, M. Dichtl, W. Fumy, M. van der Ham, C.J.A. Jansen, P. Landrock, B. Preneel, G. Roelofsen, P. de Rooij, and J. Vandewalle. RIPE Integrity Primitives, Final report of RACE 1040. Reports CS-R9324 and CS-R9325, Centrum voor Wiskunde en Informatica, April 1993.

[BBC+94] J.-P. Boly, A. Bosselars, R. Cramer, R. Michelsen, S. Mjølsnes, F. Muller, B. Pfitzmann, P. de Rooij, B. Schoenmakers, M. Schunter, L. Vallé, and M. Waidner. The ESPRIT Project CAFE — High Security Digital Payment Systems. In *Computer Security — ESORICS'94*, volume 875 of *Lecture Notes in Computer Science*. Springer-Verlag, 1994.

[BC90] J. Bos and D. Chaum. SmartCash: A Practical Electronic Payment System. Technical Report CS-R9035, CWI, August 1990.

[Bra94] S. Brands. Untraceable Off-line Cash in Wallet with Observers. In *Advances in Cryptology - proceedings of CRYPTO 93*, Lecture Notes in Computer Science, pages 302 –318. Springer-Verlag, 1994.

[Bra95] S. Brands. Off-Line Electronic Cash Based on Secret-Key Certificates. In *Proceedings of LATIN'95*, 1995. Also available as CWI technical report, CS-R9506.

[CFN90] D. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. In *Advances in Cryptology—CRYPTO '88*, volume 403 of *Lecture Notes in Computer Science*, pages 319–327, Berlin, 1990. Springer-Verlag.

[Cha92] D. Chaum. Achieving Electronic Privacy. *Scientific American*, pages 96–101, August 1992.

[DH76] W. Diffie and M. E. Hellman. New Directions in Cryptography. *IEEE Trans. Inform. Theory*, IT–22(6):644–654, November 1976.

[EG84] S. Even and O. Goldreich. Electronic Wallet. In *Advances in Cryptology - proceedings of CRYPTO 83*, pages 383 – 386. Plenum Press, 1984.

[Fer93] N. Ferguson. Single Term Off-Line Coins. In *Advances in Cryptology - proceedings of EUROCRYPT 93*, Lecture Notes in Computer Science, pages 318 –328. Springer-Verlag, 1993.

[Lam81] L. Lamport. Password Authentication with Insecure Communication. *Communications of the ACM*, 24(11):770–772, 1981.

[Lev85] L. A. Levin. One-Way Function and Pseudorandom Generators. In *Proceedings of the 17th Annual ACM Symposium on the Theory of Computing*, pages 363 – 365, 1985.

[Mer90] R. C. Merkle. A Certified Digital Signature. In *Advances in Cryptology - proceedings of CRYPTO 89*, volume 435 of *Lecture Notes in Computer Science*, pages 218–238. Springer-Verlag, 1990.

[OO92] T. Okamoto and K. Ohta. Universal Electronic Cash. In *Advances in Cryptology—CRYPTO '91*, volume 576 of *Lecture Notes in Computer Science*, pages 324–337, Berlin, 1992. Springer-Verlag.

[Riv91] R. L. Rivest. The MD4 Message Digest Algorithm. In *Advances in Cryptology - proceedings of CRYPTO 90*, volume 537 of *Lecture Notes in Computer Science*, pages 303–311. Springer-Verlag, 1991.

[RSA78] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Communications of the ACM*, 21, 1978.

[SHS92] Specifications for a Secure Hash Standard. Federal Information Processing Standards Publication, 1992.