ELSEVIER

Computer Communications 23 (2000) 425–428

computer communications

www.elsevier.com/locate/comcom

Research note

Date attachable electronic cash

C.-I. Fan[a], W.-K. Chen[b,*], Y.-S. Yeh[b]

[a]*Telecommunication Laboratories, Chunghwa Telecom Co., Ltd, 12, Lane 551, Min-Tsu Road Sec. 5, Yang-Mei, Taoyuan 326, Taiwan, ROC*
[b]*Department of Computer Science and Information Engineering, National Chiao Tung University, Hsin Chu 300, Taiwan, ROC*

Received 29 April 1999; received in revised form 17 September 1999; accepted 17 September 1999

Abstract

In this paper we propose a new untraceable electronic cash scheme which makes it possible for a payer to attach the desired date to his electronic cash during a transaction. With the aid of the date attachability property, the date on which an electronic cash is deposited in the bank cannot be forged in an electronic cash scheme. It is conducive to the unforgeability of the number of days for which the cash has been stored in the bank for some necessary purposes such as interest calculation. Our scheme not only keeps the attached date from being forged but also avoids two or more different dates being attached to the same electronic cash. Furthermore, the date attachment does not affect the untraceability property of electronic cash. Comparing with typical electronic cash schemes, the extra computation required for date attachment is just hashing.

Keywords: Untraceable electronic cash; Blind signatures; Cryptography

1. Introduction

Due to the fast progress of networking technologies, many advanced network services have been proposed in literature to take the advantages of these technologies. Among these services, electronic cash (e-cash) is a popular one since this service makes it possible for a payer in a remote site to pay his electronic cash through electronic communication networks [1–8].

A typical electronic cash scheme contains three kinds of participants (a bank, a group of payers and a group of payees) and consists of four stages (initializing, withdrawing, unblinding and depositing). In the initializing stage, the bank chooses its public and private keys. In the withdrawing stage, a payer withdraws an e-cash in a blinded version from the bank. In the unblinding stage, the payer unblinds his blinded e-cash to obtain a valid one. Finally, in the depositing stage, the payer sends his e-cash to a payee. After verifying the correctness of the e-cash, the payee forwards it to the bank for freshness checking (or double-spending checking). Then, the bank deposits the e-cash into the payee's account.

In typical electronic cash schemes proposed in the literature [1–8], the semantics embedded in an e-cash has to be determined before the bank performs the signing operation. In other words, the semantics embedded in an e-cash is fixed after it was issued by the bank. In practical situations, money deposited in a bank should be charged for interest, so that it is necessary for a customer to attach the date of depositing to his e-cash and the date cannot be modified by anyone else. This is referred to as the *date attachability* property. To prevent the attached date from being forged and ensure the correctness of interest charged, a modern untraceable electronic cash scheme is required to achieve the date attachability property. In this paper, we propose a new untraceable electronic cash scheme such that every payer can attach the desired date to his e-cash during a transaction. Our scheme keeps the attached date from being forged and avoids two or more different dates being attached to an e-cash. In addition, the attached date does not affect the untraceability property of e-cash. Especially, the additional computation for the date attachment is just hashing.

The rest of this paper is organized as follows. In Section 2, we review several basic preliminaries used in this paper. The proposed scheme is described in Section 3. In Section 4, we examine the security and discuss the date encoding in the proposed scheme. Finally, a concluding remark of this paper is given in Section 5.

2. Preliminary

In this section, we review the basic preliminaries and several correlative techniques used in this paper.

* Corresponding author. Tel.: +886-3-402-9538; fax: +886-3-402-9539.
E-mail addresses: chunifan@ms35.hinet.net (C.-I. Fan), weikchen@ms31.hinet.net (W.-K. Chen).

0140-3664/00/$ - see front matter
PII: S0140-3664(99)00189-9

2.1. Untraceable electronic cash

Untraceable electronic cash was introduced by Chaum [3]. In Chaum's e-cash scheme, there are three kinds of participants: a bank, a group of payers and a group of payees. A payer withdraws e-cash from the bank, and then pays the e-cash to a payee. The details of the protocol are shown as follows.

1. *Initializing.* The bank randomly selects two distinct large primes p and q, and computes both $n = pq$ and $\phi = (p-1)(q-1)$. The bank chooses a large integer e at random where $1 < e < \phi$ and $\mathrm{GCD}(e, \phi) = 1$, and then computes an integer d with $1 < d < \phi$ such that $ed \equiv 1 (mod\ \phi)$. Finally, the bank publishes (e, n) and a one-way hash function H [9,10], and keeps (d, p, q) secret. In addition, let every e-cash issued by the bank worth w dollars.
2. *Withdrawing.* If a payer decides to withdraw an e-cash from the bank, he randomly chooses an integer r in Z_n^* which is the set of all positive integers less than and relatively prime to n. Then the payer computes and sends $\alpha = (r^e H(m) \bmod n)$ to the bank where m is a message selected by the payer. After receiving α, the bank computes and sends $t = (\alpha^d \bmod n)$ to the payer, and then deducts w dollars from the payer's account in the bank.
3. *Unblinding.* After receiving t, the payer computes $s = (r^{-1}t \bmod n)$. The tuple (m, s) is an e-cash in the scheme.
4. *Depositing.* To pay the e-cash (m, s) to a payee, the payer sends (m, s) to the payee. The payee examines the correctness of the e-cash by verifying whether $s^e \equiv H(m) (\mathrm{mod}\ n)$ or not, and then he calls the bank to check if the e-cash is fresh (or not double-spent). If the e-cash is correct and fresh, then the payee accepts this payment and deposits (m, s) into his account. The bank stores (m, s) in its database for double-spending checking, and adds w dollars to the payee's account.

Since the integer r is randomly chosen and kept secret by the payer, it is impossible for the bank to derive the link between the e-cash (m, s) and the instance of the withdrawing protocol which produces (m, s). This is the untraceability (or unlinkability) property in e-cash schemes [1–8].

2.2. Electronic cash based on partially blind signatures

Due to the feature of electronics, the e-cash is easily to be duplicated. Hence, it is necessary for the bank to store all spent e-cash in its database for double-spending checking. Hence, the bank's database will grow unlimitedly [4,5,7,8,11]. The technique of partial blindness makes it possible to prevent the bank's database from growing unlimitedly [5,11]. In an e-cash system based on a partially blind signature scheme, each e-cash issued by the bank contains an expiration date. All expired e-cash recorded in the bank's database can be removed, so that the size of the bank's database can be controlled [5,11].

An e-cash protocol based on the partially blind signature scheme of [11] is described in the following.

1. *Initializing.* The bank randomly selects two distinct large primes p and q, and computes both $n = pq$ and $\phi = (p-1)(q-1)$. It chooses a large integer e at random where $1 < e < \phi$ and $\mathrm{GCD}(e, \phi) = 1$, and then computes an integer d with $1 < d < \phi$ such that $ed \equiv 1 (\mathrm{mod}\ \phi)$. The bank publishes (e, n) and a one-way hash function H [9,10], and keeps (d, p, q) secret. Let every e-cash issued by the bank worth w dollars.
2. *Withdrawing.* If a payer decides to withdraw an e-cash from the bank, he randomly chooses two integers m and r in Z_n^*, and sends the integers $\alpha = (r^{ev} H(m) \bmod n)$ and v to the bank where v is a message chosen by the payer and it is in the predefined format negotiated and agreed by the bank and all of the payers in advance [11]. After receiving (α, v) and verifying that v is in the predefined format, the bank sends the integer $t = (\alpha^{d_v} \bmod n)$ to the payer where $d_v = (ev)^{-1} \bmod \phi$),[1] and then deducts w dollars from the payer's account in the bank.
3. *Unblinding.* After receiving t, the payer computes $s = (r^{-1}t \bmod n)$. The triple (m, s, v) is an e-cash in the scheme.
4. *Depositing.* The payee examines the correctness of the e-cash by verifying whether $s^{ev} \equiv H(m) (\mathrm{mod}\ n)$ or not where v has to be in the predefined format, and then he calls the bank to check if the e-cash is fresh (or not double-spent). If the e-cash is correct and fresh, then the payee accepts this payment and deposits (m, s, v) into his account. The bank stores (m, s, v) in its database for double-spending checking, and adds w dollars to the payee's account.

If we let v contain an expiration date of the e-cash (m, s, v), the storage of the bank's database can be controlled because all of the expired e-cash can be removed from the database [5,11].

3. Date attachable electronic cash

In addition to the expiration date of an e-cash, the date on which the e-cash is deposited into the bank is another important information we should attach to the e-cash for some necessary purposes such as interest calculation.

Note that the proposed date attachment method can be applied to almost all e-cash scheme in the literature [1,7,8,10]. In this paper, to simplify the description, we take Chaum's scheme [3] as an example to explain our idea. Based on Chaum's untraceable electronic cash scheme described in Section 2.1, we introduce a new untraceable

[1] Abe and Fujisaki [11] had introduced a method to choose the constant v such that $((ev)^{-1} \mathrm{mod} \phi)$ exists.

electronic cash scheme which makes it possible for a payer to attach the current date to his e-cash. In our scheme, the date is not required to be determined by the payer until the corresponding e-cash is really shown for verification, and anyone else cannot forge the attached date. Most important of all, the attached date does not affect the unlinkability property of e-cash and the extra computation for the attachment is just hashing.

First, we use (1 + (the two least significant digits of a year)) to denote the year such as year 2036 is denoted by 37. In addition, all of the 12 months in a year are numbered from 1 to 12, respectively. Finally, the days within a month are numbered from 1 to 28, 29, 30 or 31 depending on different months in a year. Besides, let H be a public one-way hash function [9,10] and define the following notations:

$$H^0(m) = m \quad H^i(m) = H(H^{i-1}(m)) \text{ for every integer } i \geq 1.$$

The proposed protocol consists of four stages: initializing, withdrawing, unblinding and depositing, shown as follows.

1. *Initializing.* The bank randomly selects two distinct large primes p and q, and computes $n = pq$. Through the same key generation as the initializing stage of the protocol shown in Section 2.1, the public key (e, n) and private key (d, p, q) of the bank are generated, respectively. In addition, let every e-cash issued by the bank worth w dollars.
2. *Withdrawing.* Let $\|$ be the string concatenation operator. A payer chooses a blinding factor $r \in Z_n^*$ and randomly selects six messages $x_1, x_2, x_3, x_4, x_5, x_6$ where r and x_is with $1 \leq i \leq 6$ are kept secret. The payer computes and submits $\beta = (r^e H(m) \bmod n)$ to the bank where

   $$m = H^{100}(x_1)\|H^{100}(x_2)\|H^{12}(x_3)\|H^{12}(x_4)\|H^{31}(x_5)\|H^{31}(x_6).$$

   Note that different one-way hash function H_i can be applied to different x_i. To simplify the presentation, we apply the same H to all of the x_is. After receiving the blinded message from the payer, the bank computes $t = (\beta^d \bmod n)$ and sends the signing result t to the payer. Then the bank deducts w dollars from the payer's account.
3. *Unblinding.* After receiving the signing result, the payer performs the unblinding operation to compute $s = (r^{-1}t \bmod n)$ which is the bank's signature on m. The tuple (m, s) is an e-cash in the scheme, and it can be verified by checking whether $s^e \equiv H(m) (\bmod n)$ or not.
4. *Depositing.* When the payer decides to attach the current date including the current year a, where a = (1 + (the two least significant digits of the current year)), the current month b, and the current day c to the e-cash (m, s), he performs the following operations. Initially, the payer computes $\alpha_1 = H^a(x_1)$, $\alpha_2 = H^{100-a}(x_2)$, $\alpha_3 = H^b(x_3)$, $\alpha_4 = H^{12-b}(x_4)$, $\alpha_5 = H^c(x_5)$ and $\alpha_6 = H^{31-c}(x_6)$. And then he sends the payee the date-attached e-cash (a, b, c, s, α) where $\alpha = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}$.

   The 5-tuple can be verified by checking if

   $$s^e \equiv H(H^{100-a}(\alpha_1)\|H^a(\alpha_2)\|H^{12-b}(\alpha_3)\|H^b(\alpha_4)\|H^{31-c} \times (\alpha_5)\|H^c(\alpha_6)) \pmod n.$$

   If the above formula holds, then the payee sends the 5-tuple to the bank for double-spending checking. After performing the double-spending checking, the bank also checks the above formula to examine whether the attached date is the current date or not. Finally, the date-attached e-cash (a, b, c, s, α) is deposited into the payee's account and stored in the bank's database. The bank adds w dollars to the payee's account.

4. Discussions

In this section we examine the correctness, unforgeability and unlinkability of the proposed scheme in Section 3, and discuss the date encoding methods in the scheme.

4.1. Correctness

In the unblinding stage of the proposed scheme in Section 3, the payer computes $s = (r^{-1}t \bmod n) = (H(m)^d \bmod n)$, so that $s^e \equiv H(m) (\bmod n)$. In addition

$$\begin{aligned}&(H^{100-a}(\alpha_1)\|H^a(\alpha_2)\|H^{12-b}(\alpha_3)\|H^b(\alpha_4)\|H^{31-c}(\alpha_5)\|H^c(\alpha_6))\\ &= (H^{100}(x_1)\|H^{100}(x_2)\|H^{12}(x_3)\|H^{12}(x_4)\|H^{31}(x_5)\|H^{31}(x_6))\\ &= m.\end{aligned}$$

Hence, if (a, b, c, s, α) is a date-attached e-cash produced by the proposed protocol in Section 3 where $\alpha = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}$, then we have, that

$$s^e \equiv H(H^{100-a}(\alpha_1)\|H^a(\alpha_2)\|H^{12-b}(\alpha_3)\|H^b(\alpha_4)\|H^{31-c} \times (\alpha_5)\|H^c(\alpha_6)) \pmod n.$$

4.2. Unforgeability

The proposed date attachable electronic cash scheme is based on Chaum's untraceable electronic cash scheme [3] shown in Section 2.1. Hence, the difficulty of forging a tuple (m, s) such that $s^e \equiv H(m) (\bmod n)$ depends on the security of Ref. [3].

Furthermore, given a date-attached e-cash (a, b, c, s, α) produced by the proposed protocol, the difficulty of deriving an e-cash (a', b', c', s, α') with another date (a', b', c') and $\alpha' = \{\alpha'_1, \alpha'_2, \alpha'_3, \alpha'_4, \alpha'_5, \alpha'_6\}$ such that

$$s^e \equiv H(H^{100-a'}(\alpha'_1)\|H^{a'}(\alpha'_2)\|H^{12-b'}(\alpha'_3)\|H^{b'}(\alpha'_4)\|H^{31-c'} \times (\alpha'_5)\|H^{c'}(\alpha'_6)) \pmod n$$

relies on the strength of the one-way function H [9,10]. On the other hand, if the payer himself constructs (a, b, c, s, α)

and (a', b', c', s, α') for different payments, then they can be detected by the bank after performing the double-spending checking through the common s, and the later one used is considered to be invalid.

4.3. Unlinkability

Comparing with Chaum's electronic cash protocol [3] shown in Section 2.1, the extra information attached to an e-cash is the date on which the e-cash is deposited in the bank. Clearly, the date is known to the bank after depositing even if the date is not attached to that e-cash. Therefore, the attachment does not affect the unlinkability property which an untraceable electronic cash protocol should possess. In other words, given a date-attached e-cash produced by the proposed protocol, the bank cannot derive the instance of the withdrawing protocol which produces that e-cash [3].

4.4. Date encoding

In the proposed scheme of Section 3, we encode the date into a triple (a, b, c), and then form a date-attached e-cash (a, b, c, s, α). In such an encoding, $4(100 + 12 + 31) = 572$ hashing computations are performed to obtain and verify a date-attached e-cash, and the total length of all hashed values in $\alpha = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4, \alpha_5, \alpha_6\}$ is 6λ where λ is the length of the output of hash function H.

If we encode the current date into a tuple (a, u) with $1 \leq u \leq 366$ where the format of a is the same as that of Section 3 and the current date is the uth day in the current year. By performing a protocol similar to that of Section 3, we can obtain a date-attached e-cash (a, u, s, α) where $\alpha = \{\alpha_1, \alpha_2, \alpha_3, \alpha_4\}$, and it can be verified by checking if

$$s^e \equiv H(H^{100-a}(\alpha_1) \| H^a(\alpha_2) \| H^{366-u}(\alpha_3) \| H^u(\alpha_4)) (\mathrm{mod}\ n).$$

Thus, $4(100 + 366) = 1864$ hashing computations are required to obtain and verify a date-attached e-cash, and the total length of all hashed values in α is 4λ.

Evidently, the date encoding has a dramatic impact on efficiency. A longer encoding saves computation time of hashing but produces a longer e-cash and vice versa. The encoding of date in the proposed scheme of Section 3 is to make our idea more readable than a complicated encoding. However, adopting a shorter or longer encoding depends on the consideration for space or time in a practical implementation of the proposed scheme.

5. Conclusions

Different from embedding an expiration date into an electronic cash during withdrawing, the proposed scheme makes it possible for a payer to attach a date to an electronic cash when depositing. The attachment guarantees the unforgeability of the date on which the e-cash is deposited in the bank for some purposes such as interest calculation. Furthermore, only several hundreds of hashing computations are required to perform the attachment operation.

Acknowledgements

We would like to thank the anonymous referees of this paper for their valuable comments.

References

[1] S. Brands, Untraceable Off-line Cash in Wallets with Observers, Advances in Cryptology—CRYPTO'93 (LNCS 773), Springer, Berlin, 1993, pp. 302–318.
[2] J. Camenisch, J.M. Piveteau, M. Stadler, An Efficient Fair Payment System Protecting Privacy, Proceedings of ESORICS'94 (LNCS 875)Springer, Berlin, 1994, pp. 207–215.
[3] D. Chaum, Blind Signatures for Untraceable Payments, Advances in Cryptology—CRYPTO'82, Plenum Press, New York, 1993, pp. 199–203.
[4] D. Chaum, A. Fiat, M. Naor, Untraceable Electronic Cash, Advances in Cryptology—CRYPTO'88 (LNCS 403), Springer, Berlin, 1990, pp. 319–327.
[5] C.I. Fan, C.L. Lei, Low-computation partially blind signatures for electronic cash, IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences E81-A (5) (1998) 940–949.
[6] C.I. Fan, W.K. Chen, Y.S. Yeh, Blind Signatures with Double-Hashed Messages for Fair Electronic Elections and Ownership Claimable Digital Cash, Proceedings of First International Conference on Enterprise Information Systems 2 (1999) 612–618.
[7] N. Ferguson, Single Term Off-line Coins, Advances in Cryptology—EUROCRYPT'93 (LNCS 765), Springer, Berlin, 1994, pp. 318–328.
[8] T. Okamoto, K. Ohta, Universal Electronic Cash, Advances in Cryptology—CRYPTO'91 (LNCS 576), Springer, Berlin, 1992, pp. 324–337.
[9] M. Bellare, P. Rogaway, Random Oracles are Practical: a Paradigm for Designing Efficient Protocols, First ACM Conference on Computer and Communications SecurityACM Press, New York, 1993, pp. 62–73.
[10] A. Menezes, P. van Oorschot, S. Vanstone, Handbook of Applied Cryptography, CRC Press LLC, 1997.
[11] M. Abe, E. Fujisaki, How to Date Blind Signatures, Advances in Cryptology—ASIACRYPT'96 (LNCS 1163), Springer, Berlin, 1996, pp. 244–251.