Towards Multiple-Payment Schemes for Digital Money

H. Pagnia, R. Jansen

Darmstadt University of Technology, D-64283 Darmstadt, Germany

Abstract. Recently, many payment schemes for digital money have been proposed. In most of these schemes money can be spent only once and must then immediately be returned to the bank. The purpose of this paper is to show the advantages of a scheme which allows the recipient of the money to use it directly for further purchases. We discuss why most existing schemes do not support such a payment scheme and make a proposal of how to overcome this drawback. Furthermore, we address the problem of achieving a fair exchange of money against service between the customer and the vendor. Few solutions to this problem have been published and all involve a trusted third party which actively supports the exchange. Using such a trustee has the disadvantage that – for high transaction rates – he easily constitutes a bottleneck. We present an alternative solution based on a 'passive' trustee thereby avoiding the former disadvantage.

1 Introduction

In the last few years many payment schemes for digital money have been developed. They all have different design goals. Some schemes are designed to provide maximum security [7] other schemes are low-cost schemes suitable for so-called *micropayments* [11, 17, 18]. Besides security another important feature of a payment scheme is the degree of anonymity it provides. Electronic payment with a credit card does not offer any anonymity to the customer whereas digital cash ideally is fully anonymous [3]. A precise survey of the requirements for digital money can be found in [14]. A requirement which most of the existing schemes do not meet is the *transferability* of the money. Nearly all schemes can be classified as *single-payment schemes* (e.g. [3, 9, 11, 17, 18]). This means that after the transfer of the money from a customer U_1 to a vendor U_2 the money must be immediately returned to the bank B (see Figure 1).

Since single-payment schemes require each party to frequently contact the bank heavy load is put on the bank. This load can be reduced by applying a *multiple-payment scheme* in which the bank is contacted only sporadically. A *multiple-payment scheme* is a scheme where the recipient of the digital money can spend it with another party without contacting the bank first. In such a scheme a banknote can be used for numerous payment transactions like "real" money and not just for a single transaction. Figure 2 shows how the money is transferred within a multiple-payment scheme.

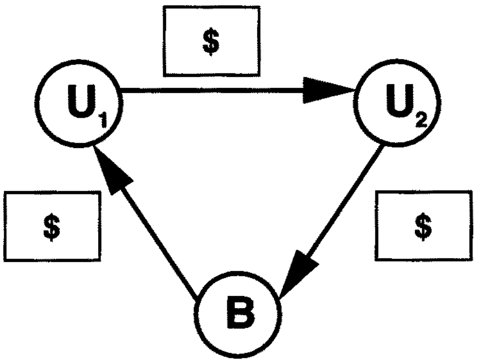

Fig. 1. A single-payment scheme

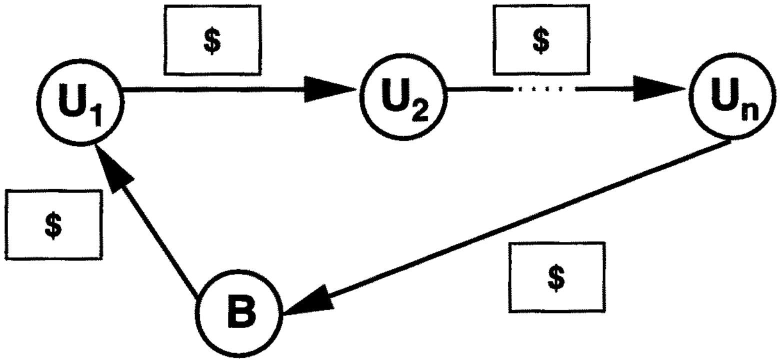

Fig. 2. Money transfer within a multiple-payment scheme

The fact that the bank does not need to be involved in every payment transaction has two advantages: First it leads to a reduced communication. Second it leads to an increased solvency of the users if the bank is temporarily unreachable. Suppose that after being paid five dollars from U_1, U_2 has to pay five dollars to U_3. Although U_2 owns the required amount, he cannot make the payment because the "used" money must first be exchanged against "fresh" money at the bank. Since multiple-payment schemes do not require such refreshing, U_2 can pay U_3 without any bank assistance.

Another important property of a multiple-payment scheme is the improved anonymity. Although the customer and the vendor do not know each other's identity, in many single-payment schemes the bank can identify both parties when debiting the customer's and crediting the vendor's bank account. If a multiple-payment scheme is used only the first party U_1 and the last party

U_n are known to the bank. Since all other parties in-between preserve their anonymity not a single payment transaction can be identified by the bank.

While digital money payment schemes are well discussed in literature, surprisingly few publications can be found concerning the fair exchange of digital money against an electronic service over a computer network. However, when the number of non-gratuitous electronic services increases in the near future this problem will become extremely important. Neither will a vendor tolerate that he obtains no payment nor will a customer accept to pay for a service which he did not receive. Without a trusted third party (called the *trustee*) either of both the customer or the vendor could defraud the other party by not delivering the payment or the service respectively. A trustee can support the exchange by first collecting the money and the service from both parties and then performing the exchange. The drawback of this protocol is that the trustee must buffer money and service until the exchange has been completed. Furthermore, if the money *and* the service have been received, the trustee is responsible to forward the service to the customer and the money to the vendor. Due to this active behavior a high load is put on the trustee making him a potential bottleneck. We propose a different protocol which allows the trustee to remain much more inactive and reduces the amount of storage needed by the trustee. Our protocol is easy to implement and can be combined with many payment schemes, e.g., with the multiple-payment scheme described above.

The paper is structured as follows. In Section 2 we explain why digital money is primarily designed for a single payment transaction. Section 3 presents a solution to the problem how a multiple-payment scheme can be designed providing anonymous payment. The problem of guaranteeing a fair exchange of money against service between a customer and a vendor is addressed in Section 4. Section 5 concludes the paper.

2 Single-Payment Schemes

The most important reason why existing schemes do not support multiple payment is the difficulty of preventing a user from keeping copies of already spent money and spending it once again (this is referred to as *double spending*). Because of this difficulty many payment schemes are on-line schemes. For example ECash [7] requires to contact the bank for every payment transaction. If communication with the bank is required anyway then multiple payment do not offer any advantages. During this communication the money might as well either be refreshed or credited to the receiving party's account.

Another system is NetCash [14] which is designed to be an off-line scheme. In NetCash double spending is prevented due to the fact that the bank *personalizes* the digital money by incorporating identifiers of both transaction partners into it. The recipient has to check that the personalized information within the money names him as the recipient. Double spending prevention is reduced to the simple check that the recipient did not already accept the same money. Both checks can be done locally without contact to the bank. Unfortunately, this technique cannot be extended to handle multiple payment, because for multiple payment identifiers of *all* recipients must be incorporated into the money. This would presume that the bank knows all recipients in advance which is an unrealistic assumption. Although in NetCash the payment itself is done off-line, the spending party has to order specially customized money from the bank beforehand. Since usually one does not know in advance where to spend money later, NetCash cannot truly be called an off-line scheme.

However, some schemes using tamper resistant hardware allow the transfer of money between two users without interference of the bank e.g. Mondex [15] [1]. Since these schemes rely on secure hardware, they can assume that the money is handled exclusively by a pre-defined untampered protocol which does not allow copying money. Therefore, double spending is assumed to be no problem.

In software-based off-line schemes where it is not generally possible to prevent double spending, mechanisms must be in place which allow the identification of fraudulent users in case double spending was detected. At first glance this seems to conflict with the requirement for anonymity. Chaum however has proposed a protocol [3] where the anonymous spending party must prove that he is the legitimate owner of the money thereby revealing his identity only in case of double spending and Chaum showed in [4] a method to extend payment systems with transferability. Unfortunately, the assertion process of his payment system is based on a costly challenge-and-response protocol.

3 Designing a Multiple-Payments Scheme

In this section we present our multiple-payment scheme. The basic idea is that the money contains personalized information containing identifiers for the current and all previous owners of the money. We are using a chained signature scheme described in [5] with extensions to protect the anonymity of the users. Initially, money consists of a money header including unchangeable information like the name of the issuing bank, its public key (signed by another authority), a serial number, the value, an expiration date, etc. For each money transfer a new entry called *transfer line* is appended which names the current owner.

Figure 3 shows how money can be passed on from one user to another. In order to withdraw some money from his account user U_1 sends his public key PK_1, his account number and the desired amount to the bank. The bank creates the header and appends the first transfer line consisting of the user's public key PK_1. Then the bank signs the whole note, delivers it to U_1 and finally charges his account with the appropriate amount. U_1 is now inscribed to be the owner of the money.

If U_1 wants to transfer the money to U_2 he has to append another transfer line which consists of U_2's public key PK_2 which must be obtained from U_2 beforehand. Afterwards U_1 signs the whole note by using his own secret key SK_1.

[1] Detailed information about the protocol used in Mondex is not available to the authors.

By signing the money, U_1 has proven to be its legitimate owner since only with knowledge of the secret key SK_1 corresponding to the public key PK_1 (which is already incorporated into the previous transfer line) he can correctly apply the signature. Furthermore, only U_2 can spend the so transferred money, since he is the only one who knows SK_2.

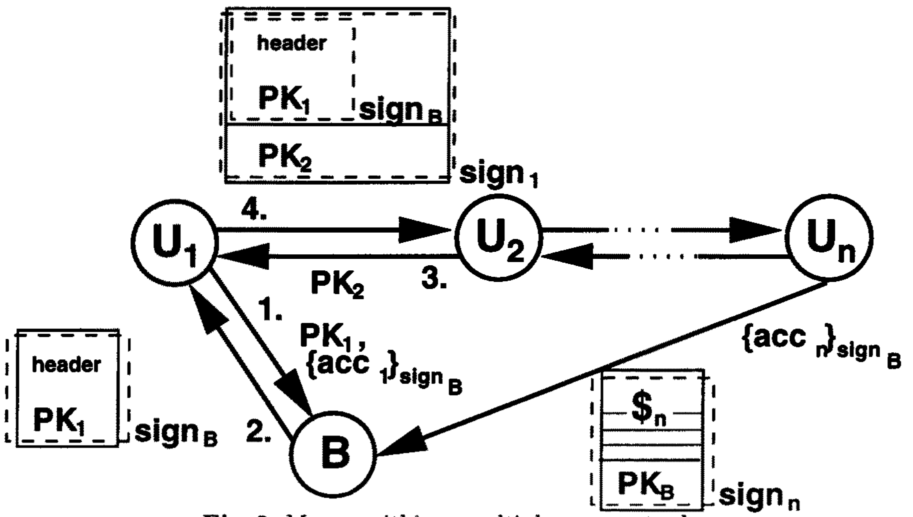

Fig. 3. Money within a multiple-payment scheme

If anonymity is desired users can generate new key pairs for every payment transaction. They do not need to reveal their identities, except for U_1 and U_n who both must submit their account numbers to the bank. However, even this can be avoided if they exchange digital money for digital money instead of making a withdrawal or a deposit respectively.

The scheme described so far provides multiple payment as well as anonymity but no double spending detection. Since the money does not contain any hints on the owner's identity, every user can spend the same money repeatedly to different partners without the risk of being traced.

In order to trace such a fraudulent user, the bank must somehow break his anonymity. Therefore, for every user $U_1, \ldots, U_n$ the banknote must incorporate some information which enables the bank to reveal his account number when double spending has been detected. The major problem that must be solved is to find a protocol which at the same time preserves the anonymity of honest users.

Instead of incorporating the account number a spender can use an anonymized authentication certificate (AAC). Such an AAC is a credential which can be obtained by a trusted Anonymity Server (AS). It consists of an arbitrary public key signed by the AS. Users might generate a public/secret key pair by themselves or they might obtain it from the AS which then acts as a public key server. In either case, the AS maintains a database which relates the signed public keys to the user identifiers. This requires that users must authenticate for the AS by using a valid 'passport' including, e.g., their bank account number. Figure 4 shows the communication between a user U_1 and the AS acting as a public key server. The signed public key $\{PK_1\}_{sign_{AS}}$ which is returned by the AS serves as an AAC for user U_1. Because the passport $\{acc_1\}_{sign_B}$ and SK_1 has to be kept in secret, the whole communication must be encrypted. It should be noted that each user can obtain numerous AACs from the AS. The AAC should be changed frequently in order to ensure maximum protection of the users' anonymity.

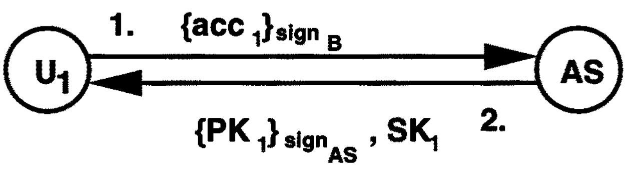

Fig. 4. Obtaining an anonymized authentication certificate from the Anonymity Server

When a user wants to obtain money from the bank, he can either withdraw money from his account or exchange it for some other money. In the first case the user reveals his identity while in the latter case the user remains anonymous.

Figure 5 shows the anonymized money transfer within our multiple-payment scheme. The user sends $\{PK_1\}_{sign_{AS}}$, a randomly chosen value R_0, and his account number (or some other money) to the bank. The bank generates the new banknote for U_1 with U_1's AAC and R_0 incorporated in the first transfer line. Finally, the bank signs the money and sends it to the user.

If user U_i wants to pass the money to user U_{i+1}, U_{i+1} sends $\{PK_{i+1}\}_{sign_{AS}}$ and a random value R_{i+1} to U_i. U_i generates the new transfer line consisting of U_{i+1}'s AAC and R_{i+1} and appends it to the hitherto existing money (denoted as '$\$_i$' in the figure). Next, U_i proves his legitimate ownership by signing the whole banknote using the secret SK_i which corresponds to his enlisted AAC $\{PK_i\}_{sign_{AS}}$. Finally, he sends it to U_{i+1}.

User U_{i+1} must now check if the money obtained is valid and if U_i was the legitimate owner. In particular, he must perform the following validations:

1. Is the money header correctly signed by the bank?
   This is essential to the scheme since nobody should be allowed to generate money by himself.
2. Are the enlisted AACs valid, i.e., are all $\{PK_k\}_{sign_{AS}}$, $k \leq i$, correctly signed by the AS?

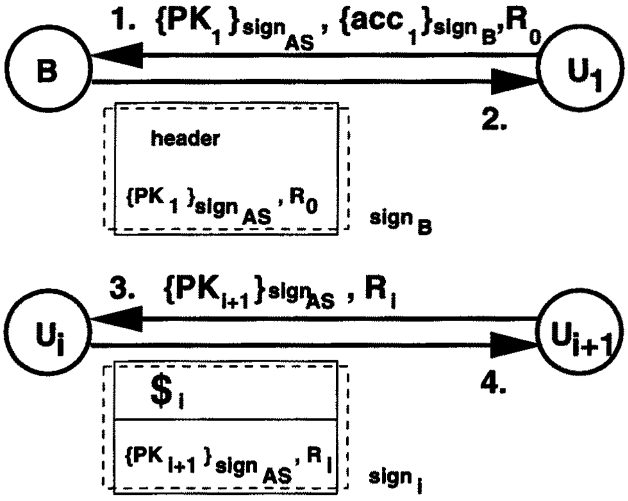

Fig. 5. Anonymized money transfer

This validation prevents the recipient from accepting money for which a previous – potentially fraudulent – owner cannot be detected by the bank later.

3. Have the transfer lines been signed correctly, i.e., do all signatures correspond to the public keys within the enlisted AACs?
   This ensures that all previous owners of the money were legitimate owners.
4. Is his own AAC $\{PK_{i+1}\}_{sign_{AS}}$ correctly incorporated into the current transfer line?
   This validation ensures that the recipient will able to spend the money to some other user.
5. Is R_i part of the current transfer line?
   This final validation is necessary because U_i might try to double spend the same banknote with still identical transfer lines to U_{i+1} at some time later. If U_{i+1} then accepts the money for the second time, he cannot spend it without being traced as a double spender by the bank. $Ui + 1$ has two options to avoid this: He can keep copies of all received banknotes in order to detect the situation described above. Alternatively, U_{i+1} must force U_i to generate two distinctive transfer lines by either incorporating a timestamp or a random value. Since keeping copies as well as checking timestamps in the presence of loosely coupled clocks would put additional burden on the users, we have favored the use of a random value R_i.

If all validations are successful then U_{i+1} accepts the money from U_i.

Validation 2 and 3 can be performed either to all entries or solely to the previous one. Checking all entries within every payment transaction is helpful in order to detect dishonest users as soon as possible: U_i might steal a valid AAC

$\{PK_x\}_{sign_{AS}}$ and submit it to U_{i-1} who writes it to the transfer line. Next, U_i double-spends the money to himself by using two correctly obtained AACs $\{PK_{i'}\}_{sign_{AS}}$ and $\{PK_{i''}\}_{sign_{AS}}$ Note that U_i cannot sign the money correctly, since he does not know the corresponding secret key to $\{PK_x\}_{sign_{AS}}$. However, because he is trying to defraud, U_1 in the role of $U_{i'}$ does not validate the signature but immediately spends the money to U_{i+1}. If U_{i+1} checks only the previous transfer line he will accept the money. Nevertheless, U_i's fraud will not be successful. When the money is finally returned to the bank it will detect the double spending and determine that $U_{i'}$ and $U_{i''}$ did not refuse the incorrectly signed money. Therefore both can be made responsible for the double spending. (Remember that both are synonyms of U_i.)

If, in contrast, all previous transfer lines are always checked, neither $U_{i'}$ nor $U_{i''}$ can spend the money. The inconsistencies will be detected immediately by any potential recipient.

The bank must keep copies of all banknotes which are returned by the users until they eventually expire. If a banknote is returned more than once then the bank can determine the AAC of the fraudulent user by comparing the transfer lines. The AAC certificate which is enlisted previous to two divergent transfer lines names the fraudulent user. The bank can forward this certificate to the AS for disclosure of the corresponding bank account number. The bank should document the fraud by additionally submitting the divergent banknotes. The anonymity revocation can also be used by law enforcement to proof a suspect of other crimes e.g. money laundering. A more detailed description of possible crimes and prevention methods can be found in [2, 13].

In the next section we will discuss the problem of a fair exchange of money against service.

4 Purchasing Electronic Services

A crucial problem when doing electronic business is to ensure that both, the customer and the vendor, fulfill their duties, i.e., the customer pays the vendor and the vendor delivers the desired service.

Suppose that due to a transmission failure the customer does not receive the service after having already paid for it. Since he is usually not known to the vendor, he will face difficulties to prove that he has already completed the payment. The situation becomes even worse in the presence of a fraudulent vendor: the vendor might take the money but denies to deliver the service. The customer simply cannot prove that the vendor did not send the service. Requiring the vendor to send the service prior to receiving the payment is not a solution. It only gives the customer the possibility to cheat. Obviously, the problem cannot be solved without the help of a trusted third party.

Only few solutions to this problem have been published so far. In NetCash [14] a protocol is described which assures that a spender is been given a receipt for the payment. The protocol needs specially customized money, namely coin triplets with each part valid within a certain window of time. A currency server serves as the trustee.

A solution without trustees is described in [12]. The protocol splits the money into two parts. The first part is transferred to the vendor before he releases the service. The second part of the money is transferred afterwards, leaving an advantage to the customer, because he might refuse to submit the second part. In this case the vendor is unable to identify the customer or to deposit the money at the bank. Nevertheless, the buyer looses his money, because he cannot spend it again without being detected as a double spender.

In [19] a protocol is described which uses the NetBill server as the trustee to actively perform the exchange of money against a receipt.

4.1 Active Trustees

Figure 6 shows a protocol solving the problem by involving an active trusted third party.

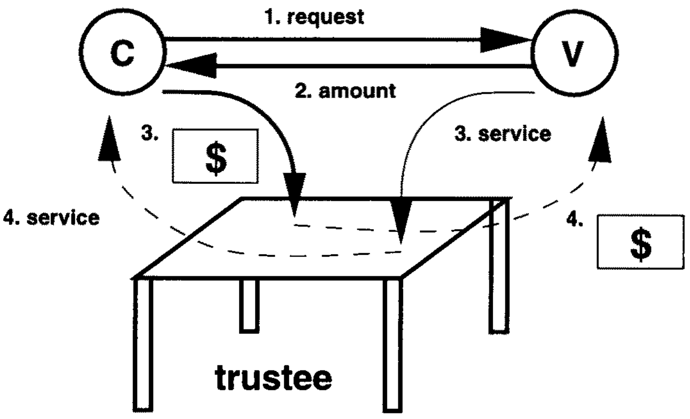

Fig. 6. Fair exchange of money against service

The customer C tells the vendor V which service he wants to buy and negotiates the price with the vendor. After receiving and inspecting the service from the vendor and the money from the customer, the trustee performs the exchange. This protocol and some variants are discussed in [6, 10]. A major advantage of the protocol is that both, the vendor and the customer, can remain anonymous to each other.

4.2 Blackboard Business

However, still some problems remain with the protocol described above. First, the service cannot be completely checked by the trustee; the customer might still complain about its quality later. Second, the trustee must be absolutely trusted to keep the neither service nor the money. Finally, since the trustee is an active party, a high transaction rate results in an enormous load making the trustee to become a bottleneck. This can be avoided if the trustee is "passive" which means the he simply logs the transactions. In the following, we assume that the customer knows the vendor but stays himself anonymous. We believe that this restriction is tolerable because it is in accordance with a normal customer-vendor relation.

Figure 7 shows a protocol with a passive trustee, called a *blackboard*, which allows fair exchange of money against service.

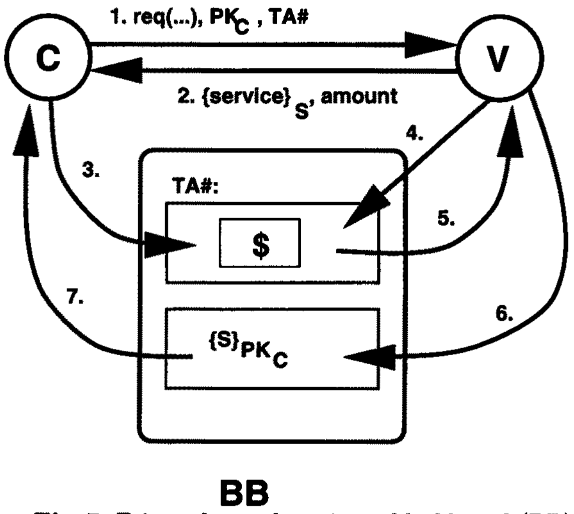

Fig. 7. Fair exchange by using a blackboard (BB)

During the first step the customer requests the service from the vendor and delivers a public key PK_C together with a unique transaction number TA#. The transaction number serves as the key for the blackboard entry. The customer can obtain TA# from the blackboard or he can randomly generate it by himself. The vendor returns the desired service encrypted with some key S and the service price to the customer (step 2). The customer now has obtained the desired service. However, the service remains useless to him unless he knows S. In step 3 the customer writes the entry index TA# and the money to the blackboard. This requires the money to be somehow personalized so that only V can spend it again. This personalization can be achieved e.g. by using the multiple-payment scheme as described in the previous section or by encrypting the money. If the amount paid matches with the price (step 4), the vendor copies the money from the blackboard (step 5) and writes S to it (step 6). S should be encrypted with PK_C so that only customer C can apply it to the encrypted service. In the final step the customer retrieves S from the blackboard and decrypts the service.

Properties of the Blackboard Protocol

Only if the customer receives the encrypted service from the vendor, he gives away the money. He can always prove that he has delivered the money since he has written it to the blackboard which is publicly readable.

The vendor is protected from fraudulent customers since he receives the money before delivering the key S. If the money does not meet his expectations he refuses to deliver S and aborts the transaction. In this case the vendor must not use the money since it is still owned by the customer who can prove it with the help of the incomplete blackboard entry for index TA#.

If the service (or key S) delivered by the vendor does not satisfy the customer, the latter can complain to the vendor and if necessary sue him using the blackboard entry as a proof. This is possible due to the assumption that the customer knows the vendor.

The blackboard is completely passive. It can only be read or written. It keeps all log entries until they eventually expire. The expiration period should be chosen according to the regulations which are made by the law. By request the blackboard might also give signed receipts for any transaction logged.

5 Conclusion

We have proposed a multiple-payment scheme for digital cash and a new method which provides a fair exchange of money against service. Our multiple-payment scheme offers anonymous payments with double spending detection. In contrast to existing single-payment schemes it has the advantage that contact to the bank is needed if two users transfer money from one to another. The money is personalized in a way that only the designated owner can spend it to another user or return it to the bank.

The proposed money/service exchange method by using a blackboard as a trustee has the advantage that the blackboard only has to store messages but it does not need to support the exchange process actively. This heavily reduces the load on the trustee and avoids that he becomes a bottleneck at high transaction rates.

To extend our payment scheme to double spending prevention, so called observers as described in [1, 8] can be included into the protocol. Observers are tamper-proof devices which increase the level of security by adding additional hardware protection.

An interesting extension to our multiple-payment protocol is to make a banknote divisible, i.e., a banknote can be split into several parts each of them spendable separately. Such an extension allows a user to spend a ten dollar note can be spent e.g. as two five dollar notes or as ten one dollar notes. To handle this, in our scheme an additional number V_i must be incorporated into all transfer lines stating the banknote's current value. Of course, $V_i \geq V_{i+1}, \quad i \geq 0$ must hold, with V_0 being the banknote's initial value as issued by the bank. The bank however must now keep logging information of all returned parts of each banknote. With the help of this logging information the bank can detect fraudulent users if the total amount spent exceeds the initial value of the money.

We are currently investigating how divisible money can be efficiently implemented for our multiple-payment scheme. We are planning to integrate both mechanisms, the multiple payment scheme and the money/service exchange method, into a framework for distributed electronic services which is based on the architecture described in [16].

References

1. S. Brands. Untraceable Off-line Cash in Wallets with Observers. In *Proceedings of Crypto 93*, pages 302–318, 1993.
2. Jan Camenisch, Jean-Marc Piveteau, and Markus Stadler. Fair Anonyme Zahlungssysteme. In *GISI 95 - Herausforderungen eine globalen Informationsverbundes für die Informatik, Informatik aktuell*, pages 254–265. Springer Verlag, 1995.
3. D. Chaum, A. Fiat, and M. Naor. Untraceable Electronic Cash. *Advances in Cryptology - CRYPTO '88, Lecture Notes in Computer Science*, 403:319–327, 1990.
4. D. Chaum and T.P. Pedersen. Transferred Cash Grows in Size. In *Advances in Cryptology ˚Eurocrypt '92*, pages 89–105. Springer-Verlag, 1992.
5. CitiBank and S. S. Rosen. Electronic Monetary System. US Patent Nr. 05453601, Sept. 1995.
6. CitiBank and S. S. Rosen. Trusted agents for open electronic commerce. US Patent Nr. 05557518, Sept. 1996.
7. DigiCash. World's First Electronic Cash Payment over Computer Networks. Press Release, May 27th 1994.
8. N. Ferguson. Extensions of single-term Coins. In *Proceedings of Crypto 93*, pages 292–301, 1993.
9. S. Glassman and M. Manasse et al. The Milicent Protocol for Inexpensive Electronic Commerce. In *Proc. of the 4th International World Wide Web Conference*, December 1995.
10. H. Bürk and A. Pfitzmann. Value Exchange Systems Enabling Security and Unobservability. *Computers & Security*, 9(8):715–721, 1990.
11. R. Hauser, M. Steiner, and M. Waidner. Micro-Payments based on iKP Technical report, IBM Research Division, Zürich Research Laboratory, 8803 Rüschlikon, Switzerland, January 1996. URL: http://www.zurich.ibm.com/Technology/Security/publications/1996/HSW96.ps.gz.
12. M. Jakobsson and M. Young. Ripping Coins for a Fair Exchange. In *Eurocrypt '95*, pages 220–230, 1995.
13. M. Jakobsson and M. Young. Revokable and Versatile Electronic Money. In *3rd ACM Conference on Computer and Communications Security*, pages 76–87. ACM, 1996.

14. G. Medvinsky and B.C. Neuman. NetCash: A design for practical electronic currency on the Internet. In *Proc. of the First ACM Conference on Computer and Communications Security*, November 1993.
15. Mondex. Home Page. URL: http://www.mondex.com/, 1996.
16. H. Pagnia, C. Liebig, and O. Theel. Mutual Trust between Customers and Providers of Distributed Services. In *Proc. of the IASTED International Conference on Networks*, January 1996.
17. T.P. Pedersen. Electronic Payments of Small Amounts. Technical Report DAIMI-PB-495, Aarhus University, Computer Science Department, August 1995.
18. R.L. Rivest and A. Shamir. PayWord and MicroMint: Two simple micropayment schemes. Technical report, MIT Laboratory for Computer Science, 545 Technology Square, Cambridge, Mass. 02139, May 1996.
19. M. Sirbu and J.D. Tygar. NetBill: An Internet Commerce System Optimized for Network Delivered Services. In *IEEE Compcon '95*, March 1995.