4-1997

A new on-line cash check scheme

Robert H. DENG
Singapore Management University, robertdeng@smu.edu.sg

Yongfei HAN

Albert B. JENG

Teow-Hin NGAIR

A New On-Line Cash Check Scheme

Robert H. Deng, Yongfei Han, Albert B. Jeng, Teow-Hin Ngair
{deng, yfhan, jeng, teowhin}@iss.nus.sg

Institute of Systems Science
National University of Singapore
Singapore

Abstract: This paper presents a new on-line cash check scheme which guarantees payor anonymity and improves upon existing similar schemes in efficiency and security.

1. Introduction

Electronic commerce, which combines the efficiency of computerization and networking, is going to dramatically change the way in which business is conducted today. One of the key issues for the realization of electronic commerce is how payments be done in a secure, anonymous, and efficient manner [9]:

- *Security*: The bank, the customer and the merchant have different security requirements. The bank wants to prevent illegal creation, copying, and reuse of electronic money; the customer wants to prevent loss of electronic money because of eavesdropping or interception; and the merchant wants to make sure that the electronic money he receives is genuine and fresh.
- *Anonymity*: In the current cashless payment systems (such as check and credit card systems), banks could easily observe who pays what amount (and sometimes for what purpose) to whom and when. With the increasing digitization of payment systems and with the rapid emergence of a virtual marketplace, compiling dossiers on individual's spending patterns and whereabouts will become easy. Measures of law and jurisdiction alone are insufficient to prevent collection and use of these data, since infringements can hardly be discovered and restoration of privacy is impossible in most cases [2]. Hence, electronic payment systems, guaranteeing full privacy of payments are necessary. This implies that the payments of a customer are untraceable, even if the bank and the merchant collude.
- *Micropayment*: It is generally believed that a fairly large portion of transactions will be of low value, ranging from payments of a few cents (for accessing on-line newspapers, magazines, and electronic holiday brochures, etc.) to payment of a few dollars (for receiving video-on-demand services and computer games). Efficient payment systems incurring low transaction costs are highly desirable.

CCS 97, Zurich, Switzerland

Numerous protocols for electronic payment have been proposed in recent years [e. g., 1-10, 12, 14]. One way of classifying such systems is based on whether theiy require on-line or off-line clearing. Most of the anonymous electronic cash systems in recent literatures [e. g., 1, 6, 7, 10]] are off-line. Off-line systems in general are more scalable than on-line systems; however, to prevent double spending, off-line systems rely on tamper resistant hardware such as smart cards. This hardware requirment may be too costly for Internet based transactions. Another drawback of the tamper resistant hardware is that new technology might allow the compromise of such hardware, leaving users vulnerable to double spending.

No tamper resistant devices are necessary in on-line payment systems. So we may term such systems as "pure software" solutions. Examples of on-line systems including Chaum's electronic cash [3, 4] and cash checks [5], the NetCash [8], and the NetBill [14]. Chaum's electronic cash and cash checks provide unconditional payor anonymity, NetCash provides conditional payor anonymity, and NetBill provides no payor anonymity.

It is well known that operations in public key cryptosystems (PKCs) are computationally intensive. All the electronic payment schemes mentioned above make extensive use of public key operations in a payment transaction; therefore, they are not ideal for micropayment applications.

The objective of this paper is to devise a payment scheme that satisfies the requirements of *security, anonymity,* and *micropayment*. To this purpose we propose a new on-line cash check scheme which improves upon Chaum's on-line cash check scheme [5]. The security and anonymity requirements are satisfied by using blind signature schemes; while the micropayment requirement is met by allowing a single cash check to be used for multiple payments between a customer and a merchant.

The rest of the paper is organized as following. We review Chaum's on-line cash check scheme in Section 2. We then introduce our scheme in Section 3. In Section 4, we first compare the two schemes and then informally analyze the new scheme against our design requirements listed above. For easy of reference, notations used throughout the paper are listed in Figure 1.

C, M, B	The customer (payor), the merchant (payee), and the bank

K_x, K_x^{-1}	The long term public/private key pair of party x, where $x = C$, or B.
$\{m\}k$	A message m signed under a private key k
f	Secure one-way hash function

Figure 1: Notations used in the paper.

2. Chaum's On-Line Cash Check Scheme

In [3, 4], Chaum et. al. proposed an electronic coin scheme based on blind signature which supports customer's anonymity. An electronic coin is a digital signature on a random value, called "note number", signed by the issuing bank. The coin scheme supports different types of coins each with a fixed denomination value. A drawback of the coin scheme is that the number of separate coins needed for a payment is in general the Hamming weight of the binary representation of the payment amount. For example, with coins of denomination values 1, 2, 4, 8, and 16 cents, to pay an amount of 15 cents, the payor must spend a 1 cent coin, a 2 cent coin, a 4 cent coin, and a 8 cent coin.

A "cookie jar" on-line cash check scheme was presented by Chaum in [5] which achieves an average of an order of magnitude saving compared with the electronic coin scheme. As in the coin scheme, the cash check scheme is constructed based on a blind version of RSA signature method [11] in order to make cash checks untraceable. Moreover, it relies on a technique for encoding denominations in digital signatures (i. e., cash checks), and "devaluing" checks to the exact amount chosen at the time of payment.

An example denomination scheme for Chaum's cash checks is shown in Table 1, where it assigns the value of 1 cent to public exponent 3 in an RSA system, the value of 2 cents to exponent 5, 4 cents to exponent 7, and so on. Using this denomination scheme, a 3rd root of an image under the one-way function f (together with the pre-image modulo the bank's RSA composite) is worth 1 cent, a 7th root is worth 4 cents, a 21st root 5 cents, and so on. A signature on an image under f is "devalued" by raising it to the public powers corresponding to the cash values that should be removed. For example, a cash check having a 21st root could be devalued from its 5 cent value, to 1 cent, simply by raising it to the 7th power.

Table 1: Denomination encoding scheme

value (cents)	public exponent
1	3
2	5
4	7
8	11
16	13

In the example of Figure 2, two cash checks are withdrawn (to keep our notations compact, only residues modulo are shown). The e_i and r_i are random. The e_i are "note numbers" and the r_i "blind" the images under f. The bank's signature corresponds to taking the hth root, where $h = 3 \cdot 5 \cdot 7 \cdot 11$ assuming that the value of the checks are 15 cents each.

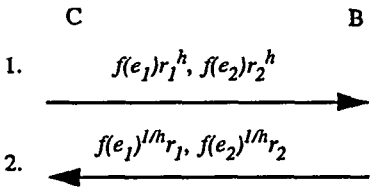

Fig. 2 Cash check withdrawal in Chaum's scheme

Upon receiving message 1 and identifying C (not explicitly shown in the figure), B signs the two received numbers by taking the hth root and returns the results to C in message 2. After receiving message 2 from B, C divides r_i out to obtain the cash checks $(e_i, f(e_i)^{1/h})$.

A payment example using the first cash check is shown in Figure 3, where the merchant M is an intermediary between C and B but is not indicated explicitly. Messages used to place and to deliver purchased items are also not shown in the figure. To spent 5 cents at M, C raises the signature in the cash check to the $(5 \cdot 11 =)$ 55th power to devalue it from 15 cents to 5 cents. The devalued cash check can be easily verified by B to be worth 5 cents. The second residue is a blinded "cookie jar", a blinded image under f of a randomly chosen value note number E. This cookie jar is modulo a second RSA composite that is only used for cookie jars. Once the bank verifies the cash check received and that e_1 has not been spent before, it signs and returns the blinded cookie jar with public exponents corresponding to the change due. Upon receiving message 2, C divides s out to obtain the cookie jar $(E, f(E)^{1/5 \cdot 11})$ which can be easily verified to be worth 10 cents. If more payments were to be made using the same cookie jar, all resulting signatures for change would accumulate. The cookie jar might conveniently be deposited during the withdrawal of the next batch of cash checks.

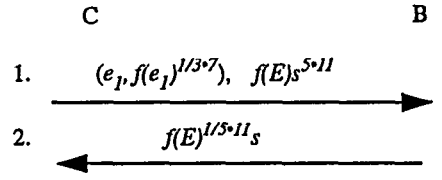

Fig. 3 Payment in Chaum's scheme

3. The New Cash Check Scheme

Our new cash check scheme is a modification to Chaum's scheme. It is also constructed based on a blind RSA

signature method in order to make cash check payment untraceable. Unlike Chaum's scheme, however, where value of a cash check is encoded according to Table 1, the new scheme encodes a fixed check value v into a fixed RSA public exponent, say 3, chosen by the issuing bank B. As a result, a cash check in the new scheme is of the form $(e, f(e)^{1/3})$. Another difference is that the note number e is not simply a random number, but a randomly selected one-time public key with the corresponding private key d kept by C. Refunds in the new scheme are collected in a cookie jar in exactly the same way as in Chaum's scheme based on the denomination encoding scheme shown in Table 1, except that the cookie jar note number E is a randomly selected one-time public key, instead of simply a random number as in Chaum's scheme.

We now illustrate the scheme using a simple withdrawal example and two payment examples. In Figure 4, C intends to withdraw two cash checks from B, each worth v = 31 cents. The r_i are random numbers which blind $f(e_i)$. The e_i (note numbers) are one-time public keys randomly chosen by C based on a certain algorithm, such as RSA or the Schnorr signature scheme [13]. C keeps the corresponding private keys d_i to herself. Upon receiving message 1 and identifying C (not explicitly shown in the figure), B signs the received numbers by taking the *3rd* root (under the cash check modulus) and returns the results to C in message 2. C extracts $f(e_i)^{1/3}$ from message 2 and stores the checks $(e_i, f(e_i)^{1/3})$ together with d_i in her computer or digital wallet.

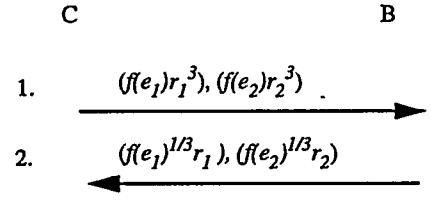

Fig. 4 Cash check withdrawal in the new scheme

C keeps a cookie jar inside her computer for accumulating refunds which can then be deposited to her account during the next withdrawal phase. For this purpose, C generates *a public and private key pair*, denoted as E and D, based on a PKC, for the cookie jar she wants to keep.

There are two options for C to spend a cash check at a merchant M. The first option is suitable for the case where C would like to have multiple shopping sessions with M using one cash check; while the second option is suitable for the case where C would like to have a single shopping session per cash check as in Chaum's scheme.

Briefly, the first payment option operates as follows. C sends M a cash check $(e, f(e)^{1/3})$, which M forwards to B for checking of double spending. If no double spending is detected, *M will establish a temporary account for C with a maximum spending cap equal to v, the value of the cash check.* The duration of the temporary account can be from hours to days or even weeks. During this period, C can place purchase orders at M with each order accompanied by a *payment token* signed under the one-time private key d. Any time before the temporary account expires, C may signal closing of the account by sending M a blinded cookie jar, which M forwards to B along with all previous received payment tokens. B can easily verify these tokens, credits the amount of payment authorized by C to M's account, puts the refund in the blinded cookie jar, and returns it to C.

An illustrative procedure for the first payment option is shown in Figure 5, where it is assumed that C would like to shop using her first cash check $(e_1, f(e_1)^{1/3})$ at a merchant M. She sends the cash check in message 1 to M who in turn forwards it to B. Once B verifies the received check and that e_1 has not been spent previously, it signs and returns message 2 to M. M checks the authenticity of message 2 and then opens a temporary account for C with e_1 as the account number, v = 31 cents as the spending cap, and a mutually agreed duration (say a week) beyond which the account expires. In message 3, C sends her payment token for an item she wants to purchase (the price negotiation messages are not shown). The payment token includes the amount of payment (e. g., 4 cents), e_1, and an information field I_1 (which may include M's identifier, timestamp, etc) and is signed under the private key d_1. M verifies the payment token using e_1, stores it with the temporary account, and then delivers the requested item in return. Upon receiving the item, C may close the current shopping session but leaves her account open at M.

The next day, C may want to have another shopping session with M. To do this, C simply sends her payment token for another item in message 4. M verifies this token and then replies with the requested item. C may continue the purchasing process by sending payment tokens in the current session or new sessions anytime before the account expires. A payment token is accepted and requested item delivered by M as long as the token is valid and the spending cap v is not exceeded.

C can close the temporary account at M any time before the account expires. C signals closing the account by sending to M the blinded cookie jar in message 5 which is then forwarded by M to B along with all the previously received payment tokens. B verifies the tokens and credits M's account with 4 + 10 = 14 cents. B then puts the amount of refund $31 - 4 - 10 = 17$ cents into the blinded cookie jar by computing the $(3 \cdot 13 =)$ 39th root of $f(E)s_1^{3 \cdot 13}$ (under the cookie jar modulus) and returns the result $f(E)^{1/3 \cdot 13}s_1$ in message 7 to M. Upon reception of message 7, M closes C's account and relays the message to C. C now extracts the cookie jar $f(E)^{1/3 \cdot 13}$ by dividing $f(E)^{1/3 \cdot 13}s_1$ with s_1. She can easily verify that her cookie jar now is worth 17 cents.

As stated before, the second payment option is designed for one shopping session per cash check. An illustrative procedure of this option is shown in Figure 6, where C would like to spend 26 cents using cash check $(e_2, f(e_2)^{1/3})$ at

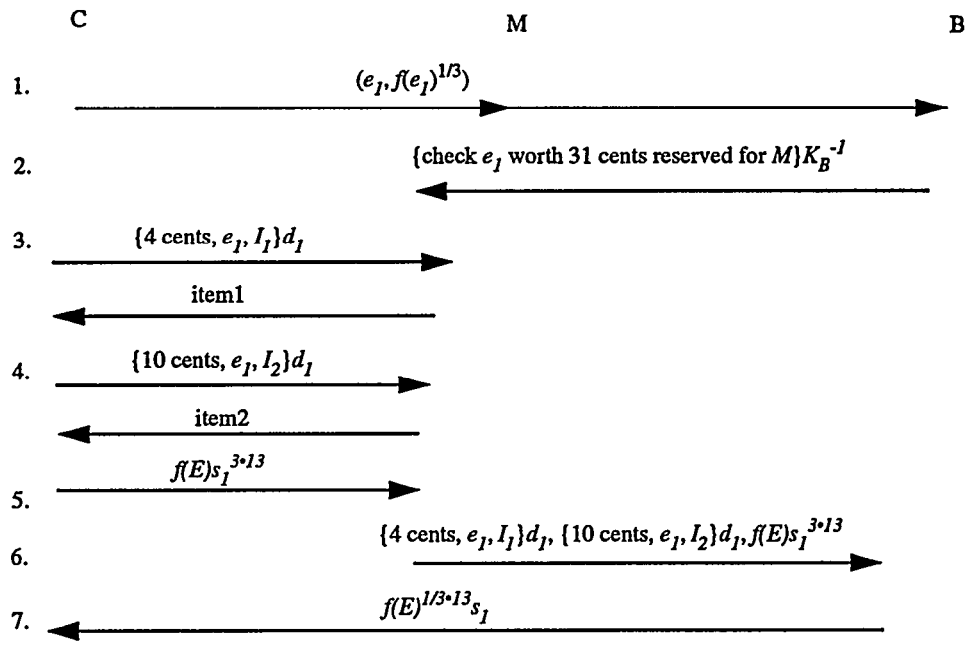

Fig. 5 Payment option 1 in the new scheme

M. *C* sends to *M* in message 1 a payment token signed under d_2, the cash check, and a non-empty cookie jar which has a value of 17 cents. *M* forwards the message to *B* for double spending checking and appropriate processing. Once *B* verifies that the check has not been spent before, it signs the blinded cookie jar (under the cookie jar modulus) with exponents corresponding the change due and returns the results to *M*. *M* then forwards the returned cookie jar along with the purchased item to *C*.

The cookie jar can be used for collecting refunds from other purchases and might be deposited, as shown in Figure 7, during the withdrawal of the next batch of cash checks. In Figure 7, the deposit message consists of the cookie jar and a *deposit token* signed under the one-time private key *D*. *B* first verifies the validity of the cookie jar and the deposit token, makes sure that *E* have not been deposited before, and then credits *C*'s account with the amount contained in the cookie jar.

4. Discussions

4.1 Comparison with Chaum's Scheme

There are a number of differences between our scheme and that of Chaum's. First, the denomination encoding scheme for cash checks in Chaum's scheme is more complex than ours. Second, the note numbers *e* for cash check and *E* for cookie jar in our scheme are one-time public keys with the corresponding private keys *d* and *D* kept in secret by the customer; while note numbers in Chaum's scheme are simply random numbers. Third, in Chaum's scheme, in order to spend a check, the amount of payment must be explicitly devalued from the check; while in our scheme the amount of payment is specified by a payment token signed under the private key *d*.

As a result, our scheme has several useful features not found in Chaum's scheme: 1) a cash check/cookie jar can only be spent/deposited by someone who knows the private key *d/D*. By keeping *d/D* in secret, the check/cookie jar can be sent in clear without having to worry them being seized by someone else; 2) since a check/cookie jar can be spent/deposited only if it is accomplished by an payment/deposit token signed under *d/D*, our scheme naturally prevents the bank from falsely accusing the customer double spending/depositing the check/cookie jar; 3) Our scheme provides two payment options suited for different payment/shopping scenarios. One option is designed for one shopping session per cash check and the other option allows a customer to open a temporary account with a merchant and then have multiple shopping sessions at the merchant. This latter option is not possible in Chaum's cookie jar cash check since it requires the customer to devalue a cash check to the exact amount of payment before sending it to the merchant.

4.2 Requirement Analysis

- *Security*: Assuming that the underlying PKCs are secure, then it is easy to see that no one except the bank can issue valid cash checks and collect refunds into cookie jars. Moreover, due to the use of public/private key paris (e, d) in a cash check $(e, f(e)^{1/3})$, the check can only be spent by the legitimate check owner, since only the owner can issue a payment token signed under the pri-

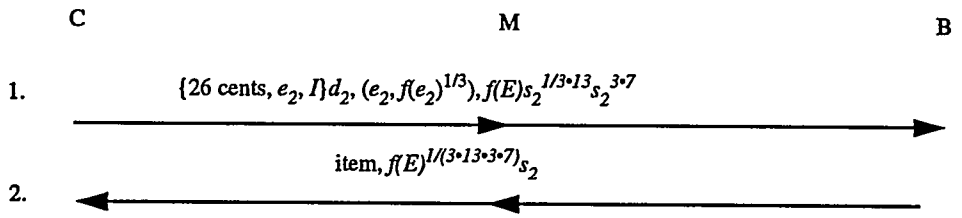

Fig. 6 Payment option 2 in the new scheme

vate key d. Similarly, due to the use of public/private key pair (E, D) in a cookie jar $(E, f(E)^{1/h})$, the cookie jar can not be deposited by anyone without knowing D. Double spending (double deposit) of a check (cookie jar) is detected by on-line clearing as in Chaum's scheme.

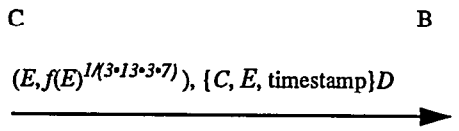

Fig. 7 Deposit of a cookie jar in the new scheme

- *Anonymity*: Cash checks and refunds in cookie jars in our scheme are produced based on blind RSA signature schemes. Following a similar argument as Chaum, it can be shown that the bank will not be able to trace a particular transaction and determine the payor's account, even if the bank and the merchant collude. That is, our scheme provides unconditional payor anonymity. It can also be shown that payments made by a payor using different cash checks are unlinkable.
- *Efficiency*: Efficiency of our scheme is achieved due to the following factors. First, only one cash check is normally sent and verified in a payment session; while in the electronic coin scheme [3], to make up a specific payment amount, the number of coins need to be payed is on average the Hamming weight of the binary representation of the amount. Second, processing and communication bottleneck in an on-line payment system is the bank server. A basic design goal of our scheme is to minimize the number of on-line messages handled by the bank. The multiple shopping session per cash check payment option allows a customer to open up a temporary account at a merchant and then have as many shopping sessions with the merchant as she desires so long as the accumulated amount of payment does not exceed the cash check value. Only two messages per account go to the bank, one for detection of double spending of cash check when setting up the temporary account and one for refund when closing the account. No messages are handled by the bank during shopping sessions. Therefore, our scheme is extremely efficient when a customer makes repeated requests from the same merchant. There will be many electronic shopping scenarios in which customers would make repeated purchases with a merchant, such as in web access to electronic newspapers, image banks, digital libraries, networked computer games, etc. Assuming that a customer opens a temporary account using a $5 dollar cash check with an electronic newspaper provider and assuming that each newspaper copy costs 20 cents, then the single cash check can be used for 25 newspaper shopping sessions. Since a maximum of two on-line messages need to handled by the bank per temporary account, the number of on-line messages per shopping session is only 2/25 = 0.08. In certain PKCs, signature generation is more expensive than signature verification. An example of this is RSA with small public exponent where signature verification is about 100 times faster that signature generation: on a typical workstation, one can sign two messages per second but verify 200 signatures per second [12]. Our cash check sheme can take advantage of such PKCs: a customer needs to perform an expensive signature generation operation while a merchant needs only to perform an inexpensive signature verification operation per payment. This arrangement improves system efficiency since the merchant rather than the customer is the system bottleneck in Internet based transactions.

There are further issues need to be considered in practical implementations of our new scheme. Normally, generation of public/private key pair *e/d* is a time consuming process. However, *e/d* do not have to be generated on-line, they can be pre-computed. Moreover, since *e/d* are used to protect a specific cash check, they can be selected such that cost of breaking the *e/d* based PKC exceeds the value of the cash check. This implies that signature generation and verification based on *e/d* can be made more computational efficient.

The bank maintains a database of already spent note numbers for checking of double spending. Several techniques were suggested in [4] for reducing database storage requirement. One of them is to have cash checks encoded with expiration dates, so that they can be purged from the bank's list of spent note numbers as they expire.

Another issue is avoiding collision of note numbers. If two customers choose the same note number, then only one of them can succeed in clearing the check, since the bank ensures that no note number is spent more than once. The

probability that any two users independently generate the same number uniformly at random is the well-known "birthday problem". To guard against birthday attack, the domain of the random chosen public key e should be sufficiently large. Assuming that Schnorr signature scheme is used to generate payment tokens. Then the private key d associated with a cash check is randomly chosen from Z_q and the corresponding public key e is given by $g^e \bmod p$, where p and q are primes such that $q|(p-1)$, and where $g \in Z_p$ with order q (i. e., $g^q = 1 \bmod p$, $g \neq 1$). In this case, the size of q should be sufficiently large, say at least 128 bits, to prevent birthday attack.

REFERENCES

[1] S. Brands, "An efficient off-line electronic cash system based on the representation problem", *Technical Report CS-R9323*, CWI, Amsterdam, 1993.

[2] H. Burk and A. Pfitzmann, "Digital payment systems enabling security and unobservability", *Computer and Security*, 8(5): pp. 399-416, 1989.

[3] D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash", *Advances in Cryptology - Proceedings of Crypto'88*, pp. 319-327, Lecture Notes in Computer Science 403, Springer-Verlag, 1988.

[4] D. Chaum, "Privacy protected payments: unconditional payer and/or payee untraceability", *Smart Card 2000*, pp. 69-93, North-Holland, 1989

[5] D. Chaum,"Online cash checks", *Advances in Cryptology - Proceedings of Eurocrypt'89*, pp. 288-293, Lecture Notes in Computer Science 434, Springer-Verlag, 1989.

[6] D. Chaum and T. Pedersen, "Wallet databases with observers", *Advances in Cryptology - Proceedings of the Crypto'92*, pp. 89-105, Lecture Note in Computer Science 740, Springer-Verlag, 1992.

[7] M. Franklin and M. Yung, "Secure and efficient off-line digital money", In A. Lingas, R. Karlsson, and S. Carlsson, editors, *Automata, Languages and Programming*, 20th International Colloquium, ICALP 93, Lund, Sweden, Lecture Note in Computer Science 700, pp. 265-276, Springer-Verlag, 1993.

[8] G. Medvinsky and B. C. Neuman, "NetCash: A design for practical practical electronic currency on the Internet", *Proceedings of 2nd the ACM Conference on Computer and Communications Security*, Nov. 1994.

[9] B. C. Neuman and G. Medvinsky,"Requirements for networked payment: the NetCheque perspective", *Proceedings of the IEEE Compcon'95*, San Francisco, March 1995.

[10] T. Okamoto and K. Ohta, "Universal electronic cash", *Advances in Cryptology - Crypto'91*, pp. 324-337, Springer-Verlag, 1992.

[11] R. L. Rivest, A. Shamir, and L. Aldeman,"A method for obtaining digital signatures and public-key cryptosystems", *Communications of the ACM*, Feb. 1978.

[12] R. L. Rivest and A. Shamir, "Payword and micromint: two simple micropatment schemes", http://theory.lcs.mit.edu/~rivest/publications.html.

[13] C. P. Schnorr, "Efficient signature generation by smart cards," *Journal of Cryptology*, Vol. 4, No. 3, pp. 161-174, 1991.

[14] M. Sirbu and J. D. Tygar, "NetBill: An Internet Commerce System", http://www.ini.cmu.edu/netbill/ComCon.html.