An efficient micropayment system based on probabilistic polling

Stanisław Jarecki* and Andrew Odlyzko**

Abstract. Existing software proposals for electronic payments can be divided into "on-line" schemes that require participation of a trusted party (the bank) in every transaction and are secure against overspending, and the "off-line" schemes that do not require a third party and guarantee only that overspending is detected when vendors submit their transaction records to the bank (usually at the end of the day).
We propose a new hybrid scheme that combines the advantages of both of the above traditional design strategies. It allows for control of overspending at a cost of only a modest increase in communication compared to the off-line schemes. Our protocol is based on probabilistic polling. During each transaction, with some small probability, the vendor forwards information about this transaction to the bank. This enables the bank to maintain an accurate approximation of a customer's spending. The frequency of polling messages is related to the monetary value of transactions and the amount of overspending the bank is willing to risk.
The probabilistic polling model creates a natural spectrum bridging the existing on-line and off-line electronic commerce models. For transactions of high monetary value, the cost of polling approaches that of the on-line schemes, but for micropayments, the cost of polling is a small increase over the traffic incurred by the off-line schemes.

1 Introduction

Inexpensive goods, such as newspapers or candy bars, are almost always paid for in cash, since the costs of other systems, such as checks or credit cards, are too high for such small transactions. Electronic commerce is expected to lead to a dramatic growth in even smaller transactions, some for less than a penny, such as purchases of individual news stories. Most of the electronic payment systems that have been proposed are not adequate for handling micropayments because of computational or communications burdens. Therefore special electronic micropayment systems have been developed recently ([GS96, GMA+95, RS96, AMS96, Ped96, BGH+95, NM95, HSW96, JY96]). They do not provide all the desirable features of the conventional electronic payment schemes, but are more efficient, and should be adequate for the small sums involved in micropayments.

Secure digital signatures do exist, so it is possible to verify who created a document. However, the basic problem with all electronic payment systems is that "bits are bits," and are easy to copy. Thus the main problem for issuers of digital money is to prevent double spending of legitimate digital currency.

* MIT Laboratory for Computer Science, 545 Tech Square, Cambridge, MA 02139, USA. Work partly done during an internship at AT&T Labs - Research. Partly supported by a DARPA grant. Email: stasio@theory.lcs.mit.edu

** AT&T Labs - Research. Email: amo@research.att.com

Some systems, such as CAFE or Mondex, rely on tamper-resistant devices, which prevent double spending by keeping users from duplicating those devices or modifying the software in them. The disadvantage of these systems is that they require special hardware, and that if their chips are reverse engineered, the issuers could face a disastrous loss.

Previous Micropayment Schemes

Software-only electronic payment systems (which are the only ones we will discuss from now on) can be secure if they are fully on-line, so that the issuer participates in each transaction (as in *iKP* [BGH+95] or *NetCheque* [NM95]). However, that is precisely what is not acceptable in micropayments, since the computational and communication requirements this imposes are excessive when a purchase costs a fraction of a penny. Therefore, many micropayment schemes have been proposed (*Millicent* [GMA+95], *PayWord* [RS96], *NetCard* [AMS96], *Pedersen*'s proposal [Ped96], and *PayTree* [JY96]) which are off-line. In all these systems, to make purchases from vendors, a customer must receive a digital certificate from a bank or some other financial intermediary. The deficiency in these systems is that they either expose the issuer of the certificate to large losses or else they limit the user's flexibility. Issuer losses arise when the user spends up to her total limit at each of, say, 10,000 vendors. (This is a disadvantage of PayWord, NetCard, PayTree, and Pedersen's scheme.) Such losses can be prevented if the customer needs to obtain a separate certificate for each vendor she intends to deal with (as in Millicent), but these certificates must then sum up to no more than the customer's bank account balance. Similarly, *Micro*-iKP ([HSW96]), a protocol that is an adaptation of *iKP* to micropayments, prevents overspending by freezing in the customer's account the maximal amount that the customer can spend with a given vendor. Both of these solutions might be unacceptably restrictive when a customer might want to deal with any of thousands of online vendors and not know beforehand how much she might spend with each.

Basic Mechanism of Our Scheme

We propose a new scheme, in which a single certificate allows a customer to deal with all vendors in the system, and possible overspending is radically reduced by the mechanism of probabilistic polling of customers' purchases. The first transaction between a customer and a vendor is always registered with the bank, but for each consecutive transaction, the vendor uses a random number to decide whether to report that transaction to the bank or not. The probability of reporting each transaction is proportional to the amount involved in that transaction. The proportion constant depends on the customer's credit (or deposit) in the bank, and the bank's trade-off between communication costs of the system and the (always small) amount of overspending it can tolerate. This constant is adjusted to produce (with high probability) an accurate estimate of the customer's total spending. When that estimate reaches a certain limit, the bank sends messages to all the vendors dealing with that customer to stop any further sales to her.

Advantages of Our Scheme

Our use of probabilistic polling makes our scheme a hybrid between the conventional on-line and off-line electronic payment schemes. In our scheme, the expected amount of polling messages triggered by a customer who spends all her credit limit is a small constant (typically between 5 and 10). It is independent of the pattern of the customer's spending: whether she makes a few big transactions or many small ones, and whether she makes

them all with different vendors or with just one. This implies in particular that for large purchases (each on the order of one fifth of the customer's credit limit) our scheme generates approximately one message per purchase (approaching the cost of a fully on-line electronic credit-card payment system), while for micropayments, the amount of overhead incurred by a single transaction becomes negligible (approaching the cost of a fully off-line system like PayWord, NetCard, etc.).

Our scheme supports a variety of policies, depending on the desired trade-off between the communication overhead of the scheme and the amount of overspending the bank is willing to risk. The maximum overall expected amount of overspending of a single dishonest customer is proportional to the amount of credit given to that customer. This proportion constant must be bigger then 1, and the closer it is to 1 the bigger the communication cost of the scheme. However, the expected losses caused by overspending can be eliminated almost entirely, if in exchange for a certificate, the bank requires an extra security deposit in addition to the value of the electronic cash a user can spend with that certificate. For example, if a customer needed, say, $70 of electronic cash, she could go to the bank, deposit $100 into an account, and be allowed to spend her cash with any vendors, subject to the constraint that her total spending during the day should not exceed $70. (Any unspent money would be refunded to her the next day.)

Our scheme shares several desirable features of cash. For users, there is some anonymity, in that the bank does not need to know the customers to accept their cash and open an account. (Users can pay the bank with anonymous electronic cash, and then the bank gains no knowledge about them, other than who they buy from.) The vendors learn only the customers' names as specified by the certificate from the bank, and even if a customer chooses to present herself as a new purchaser later in the day, the vendor has no way of finding that out. In addition, the bank does not know what the customers are buying from vendors (but does know which vendors they deal with, and how much they spend with them). When the customer is required to make a deposit, there is the added advantage for the bank that, just as with cash, the customer cannot refuse to pay because the vendor sold defective goods. This minimizes customer care costs, which can easily destroy profitability of a system that can collect only tiny amount from each of many small transactions.

Often a credit-type system can be adequate. For example, over 90% of the population of the U.S. has phone service, which allows unlimited credit in calling within a single billing period. (However, deposits are sometimes required of customers, and increasingly there are checks for unusual calling patterns, to detect fraud.) On the other hand, only about half of the U.S. population is regarded as eligible for regular credit cards, and there is careful tracking of transactions to ensure that credit limits are not exceeded. Others are required to make deposits to obtain credit cards, and can only spend up to the amount deposited. Therefore, in the coming world of electronic commerce, where there will be transactions across the world between strangers in a variety of goods, it appears advantageous to provide a flexible micropayment system that does not depend on too much trust in customers.

Probabilistic polling schemes have also been proposed by Gabber and Silberschatz [GS96] and Yacobi [Yac97]. Both of them use a fixed probability of reporting a transaction. Our proposal, which varies the probability according to the size of the transaction, has advantages in efficiency and flexibility.

Organization
We describe our system in detail in section 2 and analyze its performance and give guidelines to setting its parameters in section 3. In section 4 we present three interesting variations of the micropayment system based on polling, and we conclude in section 5. Appendix A contains a discussion of a technical issue of controlling the random numbers used by vendors to decide on polling customers' spending. Appendix B contains mathematical details and tables of data that are helpful in analyzing the performance of the system. Appendix C contains figures with data used in the performane analysis in section 3.

2 Polling System

2.1 The Underlying Off-line Micropayment Schemes

Our probabilistic polling mechanism can be added to all the off-line, single certificate micropayment schemes, such as Agora, PayWord, NetCard, Pedersen's scheme, and PayTree. The first three of these schemes use chains of hash values in a very similar way, while PayTree uses tree-structure of hashes, which has some computational advantages.

NOTATION: We will use symbol $\{M\}_X$ to designate a public-key signature of party X on message M. By convention, $\{M\}_X$ includes the message M itself. We designate the corresponding public key of party X by PK_X. In general, we will use subscripts, like A_X, to designate a variable of type A, but belonging, originated or destined to party X (the specific meaning will be clear from the context).

User Initialization
All the above payment schemes share the same basic structure: All vendors and users know the public key PK_B of the bank. Each user U has a certificate C_U, which contains the following data:

$$C_U = \{B, U, A_U, PK_U, Exp, Max\}_B$$

where B is the bank issuing the certificate, A_U is the user's address, PK_U is her public key, Exp is the expiration date of this certificate, and Max is the maximum amount that U is allowed to spend with any vendor.

First Payment
In all these schemes, the first payment of user U to some vendor V is more complex than the consecutive payments, hence we will call this first payment a "registration with a vendor". In all the schemes, this message conforms to the following pattern:

$$Reg_{UV} = \{C_U, V, T, \mathcal{P}^{(1)}\}_U$$

where T is the time of the purchase and $\mathcal{P}$ is a specific payment field, different for every scheme. For example, In PayWord, NetCard and in Pedersen's scheme, $\mathcal{P}^{(1)}$ contains a unit worth (in dollars) of future payments from U to V, and a value $x_n = h^n(x_0)$ which is a result of n applications of publicly known hash function h to a random value x_0 (see [RS96, AMS96, Ped96] for details). In PayTree, $\mathcal{P}^{(1)}$ contains a root of a tree of hash applications to random leaf values (see [JY96]). The future payment schemes might do it in yet another way.

On receiving Reg_{UV}, V checks B's signature on C_U, extracts PK_U from C_U, verifies U's signature on Reg_{UV}, and accepts the first payment [Pay] if it is well formed.

Consecutive Payments and Reimbursement
If everything goes well with the first payment, V will accept consecutive payments $\mathcal{P}^{(i)}$ of U during that day, until the maximum value Max is reached. At the end of the day, V submits to the bank all the payments of U and the bank reimburses V after checking that the payments are valid.

Unfortunately, if the user was dishonest and spent more than her available credit, the bank can reimburse the vendors only partially. The overspending of user U is potentially bounded only by the number of vendors in the system and the value Max in the certificate C_U. Hence, if there are many vendors in the system, the average loss caused by a dishonest user U per vendor can be equal almost to Max_U.

2.2 Adding a Probabilistic Polling Mechanism to PayWord

To control overspending, we add the following probabilistic mechanism to the underlying scheme described above:

Bank and User Initialization
The bank B picks constants c and M, where c is the expected number of polling messages triggered by a user who spends up to her credit limit, and M is the number of polling messages about a user that will cause the bank to suspect that this user is overspending her credit limit. These values are determined by the desired trade-off between the level of security and the computational complexity of the polling mechanism (We will examine this relationship in section 3). These constants can be different for different customers, depending on how much those customers have on deposit, or what their past record has been. The reasonable values for c are between 2 and 10, and for M between 5 and 30. For every user U, the bank stores the following information:

$$I = \{\mathcal{M}, x, L\}$$

where $\mathcal{M}$ is the dollar amount that B allows that user to spend[1], x is the counter of polling messages, initially zero, and L is a list of vendors, initially empty, that could be involved in transactions with that user. When user U buys $\mathcal{M}$ amount of electronic coins, B computes $f = \frac{c}{\mathcal{M}}$ and gives to U a modified certificate:

$$C_U = \{B, U, A_U, PK_U, Exp, f\}_B$$

First Payment: Registration with a Vendor
The payment protocol between U and V is the same as in the underlying scheme, except that the behavior of the vendor is modified as follows: On receiving Reg_{UV}, V checks whether $u_{UV} * f_U \leq 1$, where u_{UV} is the worth (in dollars) of this payment. If $u_{UV} * f_U > 1$, the vendor should not accept U's payment at all. (Alternatively, it could treat U's payments as a bundle of several separate payments, each of value less then $\frac{1}{f_U}$.) Otherwise, V forwards

[1] For simplicity we will describe the scheme as credit-based, but we can also require a deposit $\mathcal{M}'$ from each user.

Reg_{UV} to B. When B receives this registration, it checks the signatures on it, adds V to the list L_U, and replies to V with an acknowledgment if $x_U < M$, or a rejection otherwise. These replies should contain some hashes of the message Reg that triggered them, and for increased security these replies can be signed by the bank.

Since registration with vendor V occurs at the time of first payment, V decides then whether to report this payment to the bank (see below). When V gets an acknowledgment from B, she can start selling her goods to U. Actually, she can start selling straight away, bearing a risk that if the bank eventually sends her a rejection, she will not be reimbursed for the sales. Each vendor can formulate a separate policy in this regard, depending on the amount of fraud encountered by that vendor so far, and the average delays the vendor is experiencing while waiting for the bank's acknowledgment.

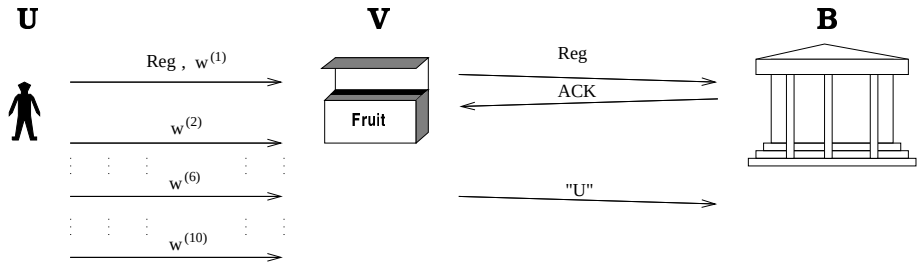
Payments and Polling

The selling protocol is the same as in the underlying scheme, except that at every payment V receives, she sends a message with U's name to B (f_U is the user-specific parameter in C_U) with probability

$$p = u * f_U \ (= \frac{u}{\mathcal{M}_U} * c)$$

where u is the dollar value of the payment.

Below is the picture giving an example of a payment session with the polling system. The user sends his registration (which includes a first payment $\mathcal{P}^{(1)}$) to the vendor, who relays the registration to the bank. The user makes an additional nine payments, $\mathcal{P}^{(2)}$ to $\mathcal{P}^{(10)}$, but the random number generator determines that only the sixth payment $\mathcal{P}^{(6)}$ triggers a polling message from V to B about U:



The first payment $\mathcal{P}^{(1)}$ which is contained in the registration message Reg_{UV} can also trigger a polling message from V to B. Such polling message should be piggybacked with the forwarding of the Reg_{UV} from V to B. Even then, V has a choice of waiting with the transmission of the goods to U for B's reply.

Every time the bank receives a report from some vendor about user U, it increments the x_U counter by one unit. When x_U reaches M, the bank broadcasts an alert message to all vendors on the L_U list. When a vendor receives such a message, it halts its dealings with U, and if V has sold anything to U that day (it might be that all that V received from U was Reg_{UV}), she sends to B all the payments she received from U on that day. This is the same type of message that she would send at the end of the day to get reimbursed for U's purchases.

Dealing with Overspending
The bank B, after receiving the data from all vendors in L_U, checks whether all the signatures are correct and computes z_U, the total value (in dollars) of the payments of U on that day. If $z_U > \mathcal{M}_U$, then U has overspent his credit (or deposit) and the bank can then freeze his account, assess penalty fees, and potentially prosecute him.

The losses caused by a dishonest user must be shared by the bank and the vendors in some way that they negotiated beforehand. The bank can set up various policies for reimbursing vendors who sold goods to an overspending user (see appendix A). We think that the best method is for the bank to provide a following partial reimbursement: divide $\mathcal{M}_U$ between the vendors in L_U, by paying to each $V \in L_U$, $\frac{x_{UV}}{x_U} * \mathcal{M}_U$, where x_{UV} is the number of reports that B received from V about U's purchases.[2] This assumes that B has to monitor not only the total number x_U of reports about U, but also how many reports came from a particular vendor. This policy requires that the bank monitor whether the random coins the vendors use for payment-polling decisions are not skewed. We will describe how and why the bank should do that in the appendix A.

If $z_U \leq \mathcal{M}_U$, then the alert about U was broadcast unnecessarily. In that case, the bank should set x_U to $\lceil c \rceil$ and broadcast to all $V \in L_U$ that the alert about user U is canceled.[3] After receiving such a message, vendors start selling to U again, following the same protocol as above. Since all transactions are electronic, all that the user notices is a delay in processing her transactions, and if the user had not overspent, so that the alert message was wrong, there is no embarrassment to the user (nor lawsuits against the vendor), as there is when detectives arrest a customer on suspicion of shoplifting, say.

3 Performance Analysis

In evaluating the performance of the micropayment scheme based on our probabilistic polling mechanism, we are interested in the following measures:

1. The average (expected) amount that a thief can spend without having money on deposit to pay for it. An important factor here is the ratio between the expected amount of purchases of a single user that triggers the bank to send alert messages, and the credit $\mathcal{M}$ given to that user. We will denote that ratio by k.
2. The volume of additional communication and computation.
3. The delays that are slowing down the micropayment protocol. One type of delay experienced by an honest user will be a halt in her transactions caused by the bank

[2] Optionally, the bank can require a deposit $\mathcal{M}'$ before it gives a credit $\mathcal{M}$ to a user. Then the bank would divide $\mathcal{M}'$, not $\mathcal{M}$ among the vendors.

[3] The counter is reduced to $\lceil c \rceil$ because if the user is going to overspend her credit afterwards, the expected amount she spends from that point before being caught is $\mathcal{M} * \left(\frac{M - \lceil c \rceil}{c}\right) \approx (k-1)\mathcal{M}$. Assuming the user spent almost her credit $\mathcal{M}$ before the first alert, her overall spending before being caught will be $\mathcal{M} + (k-1) * \mathcal{M} = k * \mathcal{M}$, which is the amount at which we want to catch all the overspenders (variable s in appendix B). On the other hand, with this choice of reevaluating the counter x_u the probability that an honest user will cause an alert twice in a row is absolutely negligible. These computations are easier to understand after reading the analysis in section 3 and appendix B.

which unnecessarily sends out alert messages about that user, even though the user has not overspent her credit. We will denote the probability of such event by d. We can tune our system so that this probability is a small number, like 0.03 or 0.001, but it cannot be ignored.

In section 3.1 we will present the relations between values k, c, M and d, which are the parameters that define the performance of our system. In section 3.2 we discuss the money losses caused by thieves. In section 3.3 we analyze the volume of the communication introduced by the system. In section 3.4 we examine the delays introduced by the polling mechanism. Finally in section 3.5 we give a procedure for setting up the system parameters so that the probabilistic polling micropayment scheme performs at an optimal cost under given conditions.

3.1 Relations between System Parameters

Since every user who overspends even by 1 cent is detected at the end of the day, we assume that if somebody decides to steal, they will try to steal as much as possible until the bank broadcasts the alert messages. In that case, we can ask for the expected amount of over-the-limit purchases of a thief U at the point where $x_U = M$. In appendix B we show that k, the ratio between the expected amount a thief has spent which triggered an alert (i.e. the expected amount a user must spend to generate M polling messages) and the amount of his credit $\mathcal{M}$ in the bank is $k = \frac{M}{c}$, which leads to the following relation between the parameters:

$$M = c * k. \tag{1}$$

The parameters d, M and c satisfy the following inequality, also computed in appendix B:

$$d \leq 0.8 \frac{c^M e^{M-c-1}}{\sqrt{2\pi}(M-c-1)(M-1)^{M-\frac{1}{2}}}. \tag{2}$$

The constant 0.8 in this equation comes from an inspection of the "goodness" of our approximation for the range of values that would make sense in our system, i.e. $1.3 \leq k \leq 5$, $0.001 \leq d \leq 0.1$. The ranges of c and M that correspond to these values of d and M are approximately $5 \leq c \leq 100$, $3 \leq M \leq 50$.

Notation
In the cost analysis below we will consider the following quantities given by external circumstances:

- U - the number of customers using the system
- V - the number of available vendors
- N - the average number of purchases per user per day
- W - the average number of vendors a single user makes purchases from in a day
- $t*U$ - the number of thieves (i.e. overspending users) encountered by the system in one day. We will assume that the number of thieves is a small proportion ($t \approx 10^{-2}$) of all customers.

We should also recollect the variables that we, the system designers, have a control over:

- $\mathcal{M}$ - the credit given to a customer
- $\mathcal{M}'$ - (optional) the deposit required of a customer
- c - the expected number of polling messages originated by a user who spends all her credit $\mathcal{M}$
- x_U - bank's counter of the polls about user U
- M - the threshold s.t. if $x_U = M$, the bank sends alert messages about user U
- k - a number s.t. $k\mathcal{M}$ is the expected value of purchases that will originate M polling messages, i.e. the expected amount of purchases which will trigger the bank to send alert messages about the user.
- d - probability that an honest user who spends up to $\mathcal{M}$ of electronic cash, will still cause M polls, and hence trigger the bank to send alert messages.

3.2 Average Expected Loss caused by Thieves

In all off-line and single-certificate cash schemes, every thief can spend up to his maximum credit with each vendor. Hence the total losses of these schemes caused by a single thief are bounded only by $V * \mathcal{M}$, where $\mathcal{M}$ is the average credit (or deposit) of a thief.

Thanks to the polling mechanism, potential losses can be greatly reduced. The expected amount a single thief can overspend can be expressed as $(k-1) * \mathcal{M} + E$, where E is the value of goods a thief U can purchase during the time between the moment when some vendor originated the polling message which caused bank's monitoring variable x_U to reach the threshold value M, and the time the alert message reached all vendors that U tried to buy something from. (They all must be on the L_U list of the bank.) If we require a deposit $\mathcal{M}'$ from each user, the expected loss caused by a single thief becomes

$$\text{financial loss per one thief} = k\mathcal{M} - \mathcal{M}' + E \tag{3}$$

We can minimize this number if we set $\frac{\mathcal{M}'}{\mathcal{M}} = k + \frac{E}{\mathcal{M}}$, but that could make $\frac{\mathcal{M}'}{\mathcal{M}}$ too big, especially for small $\mathcal{M}$. However, if we simply set $\frac{\mathcal{M}'}{\mathcal{M}} = k$, then the average loss due to a single thief would be E.

Hence the total loss caused by a thief in our scheme could be equal to *the amount one can buy during a delay between two messages flowing across the network* (the last poll that triggered the alert going from some vendor to the bank and the alert message going the other way), while in the off-line, single-certificate schemes (like PayWord), this cost is equal to *the amount one can buy during a whole day*. Obviously, this is a dramatic reduction.

Consider the delay between the moment when some vendor sends the poll that triggers the bank to broadcast the alert messages, and the moment when this alert reaches all the vendors that might be dealing with the overspending user. This delay will be on the order of the delay experienced by a vendor in waiting for a single acknowledgment, unless the number of vendors on the L_U list is exceptionally high. However, because users cannot register with vendors without making purchases, the thief can try to make the L_U list long only if he finds many vendors who are selling very cheap products and makes single purchases from them. It is not clear whether the thief can buy any time for himself in this way.

3.3 Volume of Communication Added by Polling

The volume of the additional communication caused by our polling system in a day can be counted as follows:

$$\begin{aligned} &\text{polling messages} + \text{regs. forwarded by vendors} \\ &+ \text{acks. from banks to vendors} + \text{msgs. in alerts caused by thieves} \\ &+ \text{msgs. in alerts caused by honest users} + \text{polling messages caused by thieves} = \\ &= cU + (1 - \frac{c}{N})UW + UW + 2tUW + 3dUW + tUM. \end{aligned}$$

Hence, the volume of the additional traffic per one user is:

$$\text{volume of traffic per user} = (c + tM) + W * \left(2 + 3d + 2t - \frac{c}{N}\right). \qquad (4)$$

We refer the reader to appendix B for better understanding of the mathematics of this system. We note that from the number of registrations forwarded by the vendors (which is equal to UW), we excluded the $\frac{c}{N}$ fraction of registration messages that carry a polling message piggybacked on them, since such messages are counted in the cU expression for the number of polling messages. In the above equation, we assume that every user spends every day all the coins she purchased (hence c is the expected number of polling messages caused by a single user). Also, we ignore the negligible possibility that an honest user can cause alerts twice on the same day. In the computations below we will fix t to be $.01$, but we note that the value of t in the range between 0 and 0.1 has a minimal influence on the results.

We will examine the cost function above for different values of W and N and for $k = 1.5$ and $k = 3$. It turns out that if we take M for which the cost function is minimized, values of d can be larger then the 0.01 that we assume in the computations in appendix B. Therefore, we will always compute two values of this cost function: one for M_{min} which gives minimal cost, and the other for the smallest M for which d is smaller then 0.01. The latter is a minimal communication cost subject to the constraint that $d \leq 0.01$. These values are included for reference in figure 2. Also for reference, we show in table 3 the values of M_{min} and d corresponding to the minimal costs shown in figure 2, and in figure 4 we give values of $M_{d=0.01}$, i.e. the smallest M s.t. $d \leq 0.01$, for different k's.

Figure 1 shows the relative increase in the communication costs of adding the polling mechanism to PayWord. The communication cost of PayWord itself is $2N$ messages per user per day (we count the messages carrying the purchased goods from the vendor to the user). For comparison, the communication cost (per user) of a fully on-line scheme like *iKP* is $4N$, because if the bank participates in every transaction, there is 100% increase in traffic. In figure 1 we give a range *from-to* of the increase in communication costs, where *from* corresponds to the minimal cost for $M = M_{min}$, and *to* corresponds to the cost for $M = M_{d=0.01}$. Table 1 simply shows the proportion between the added cost of the polling mechanism[4] and the communication cost of PayWord itself, i.e. $2 * N$.

[4] The volume of communication added by the polling mechanism is shown in table 2. The entries (the *from-to* range) were computed using equation 4. Value *from* corresponds to the choice of M_{min} (and the corresponding d's and c's as shown in table 3) while *to* is computed for $M_{d=0.01}$ (shown in table 4). Tables 2, 3 and 4 are in the appendix C.

k=1.5	$W=1$	$W=5$	$W=25$	**k=3**	$W=1$	$W=5$	$W=25$
$N=10$	$35-57\%$	$82-85\%$	NA	$N=10$	$17-22\%$	57%	NA
$N=70$	$6-21\%$	$14-25\%$	$46-49\%$	$N=70$	$2-3\%$	9%	38%
$N=500$	$1-3\%$	$2-4\%$	$7-8\%$	$N=500$	0.5%	1%	5%

Fig. 1. Increase of the communication costs due to the polling mechanism for $k=1.5$ and $k=3$

We observe the worst performance occurs for low N and high W. Since we assumed that the user always spends all the coins she purchased, a small number of transactions means that the amount of each transaction is high relatively to the amount of user's deposit (credit) in the bank. For $k=1.5$, value of $N=10$ is so small that $M_{min}=40$ lies beyond the range of possible threshold values M, because $M \leq kN = 15$.[5] This bound follows from the fact that $M=ck$, $uf \leq 1$ and $\mathcal{M}=\frac{c}{f}=uN$. If $N=\frac{M}{k}$, the amount of each payment is maximal allowed by the corresponding value f, and hence, each payment will invoke a polling message automatically. This explains why for a small number (or high money value) of payments, the polling system is least efficient. Similarly, a large value of W causes large user-vendor registration overhead, and hence leads to worse performance. If $W=N$, i.e. a user makes only one purchase from each vendor, then the increase of communication costs caused by the polling mechanism will be 100%, i.e., just like *iKP*.

It is hard to predict the values of N and W a real-life micropayment scheme might encounter, especially since we probably do not foresee all future uses of micropayments. However, the polling mechanism always gives less overhead than a fully on-line scheme, and for many patterns of usage, this overhead can be as small as 5%.

3.4 Delays Added by Polling

Since polling increases communications traffic, it will also lengthen the delay experienced by every single message. The increase of this average delay depends largely on whether the vendors are running close to their maximal throughput capability. To preserve the same performance, one would have to set up higher message-processing requirements for the vendor machines. The change would have to be roughly proportional to the traffic volume increases (as shown in table 1).

Other important delays come in if a vendor has a policy to wait with sales for an acknowledgment from a bank. This delay depends on the number of users U, number W of average user-vendor registrations per user, the communication capabilities of the bank, and the time it takes for the bank to process a single registration message. The arrival of vendor registration messages can be modeled as a Poisson process, and hence an average delay can be computed easily. We notice that the bank can decrease the registration-acknowledgment delay experienced by the vendors, by splitting its job into many separate servers. Instead of a single machine B, the bank can have several servers $B_1, \ldots, B_n$, each responsible for only $\frac{1}{n}$ fraction of the users. If the delays were caused not by the volume of traffic but by the

[5] Consequently, in figure 1, for $N=10$ and $k=1.5$, value *to* is computed not for $M_{d=0.01}=40$ but for $M=15$.

distance between vendors and a single central polling station B_i, these separate machines (they don't need to cooperate except in user registration) can be furthermore implemented by a distributed system of servers.

A delay experienced by an honest customer due to the probability d that she will cause a false alert and will be blocked from making purchases can be best taken care of by setting d as small as possible (without increasing M, and hence, the overall traffic volume too much). We should note, though, that the probability that an honest user will experience such a halt is smaller then d (as given by definitions and approximations in section 3.1). The highest possibility of unnecessary alert comes only if some user spends almost all the coins she purchased. This might not happen that often in the first place. We can furthermore imagine that the user's software will warn her when she has spent, say 85% of her deposit (credit), and would almost automatically contact the bank to purchase additional coins. Such a mechanism will decrease d even further (about twice on the average).

3.5 Guidelines for Setting up a Probabilistic Polling System

The simple guidelines for tuning the performance of this system can be stated as follows:

- k controls the financial losses incurred from thieves: we have to keep it small.
- The maximum expected number of polling messages from a single user is c. To reduce the processing time of the bank and the vendors and to decrease the volume of traffic introduced by the polling mechanism (and consequently the delays), we have to keep c small.
- d is the probability of the costly "alerts" caused by perfectly honest users. To decrease delays, processing time and overall traffic volume, we need to keep d very small.

However, as noted in section 3.1, these parameters are interdependent. So how shall one go about setting them up? We propose the following design loop (see the *notation* paragraph in section 3.1 for reference on variables):

1. Decide on the values of $\mathcal{M}, \mathcal{M}'$ and k for which it makes financial sense to run the system, taking the first estimation of value E (additional losses due to communication delays, see equation (3)) as, say, $\mathcal{M}$.
2. Estimate expected values of W, N and t.
3. Using equations (1) and (2), express M and d as functions of k and c. Substitute them into equation (4) for the amount of message traffic per user, and find c which minimizes that expression. This also sets the value of M.
4. Run experiments by modeling the purchasing decisions of the customers with Poisson processes, observe the average communication delays and estimate the value of E.
5. Estimate the losses again from equation (3). If something can be improved, adjust your $\mathcal{M}$,$\mathcal{M}'$ and k parameters accordingly and repeat from point 2.

When the system is up and running, the bank should monitor the values of U, W, t, N, E and d. To keep the optimal performance, the bank should adjust parameters $\frac{\mathcal{M}'}{\mathcal{M}}$, $\mathcal{M}_{min}$, $\mathcal{M}_{max}$, k and c (and hence M) according to this real data, instead of relying on modeling and on estimates.

4 Variations and Extensions

We present three variations of the probabilistic polling micropayment scheme we described in the sections above. First we show that our system can work without public key encryption. Second, we show a deterministic version of our scheme. Finally, we discuss a modification in which the vendors do not forward the customer registration information to the bank.

4.1 Using a Symmetric Key System

If every user-to-vendor registration is forwarded to the bank, and the vendor has a policy of waiting for the bank's acknowledgment with sales to the user, then the signature of the user on the Reg_{UV} message can be computed with a symmetric key (say, a DES or RC5 key) that is shared by the bank and the user. The vendor could not not check the authenticity of such a registration message, but would forward it to the bank. The bank would check the signature on Reg, and reply with acknowledgment or denial to the vendor, also signed with a symmetric encryption scheme using a shared vendor-bank key.

This change would necessitate increased trust in the bank, as the bank could now defraud the users. The users would then be at risk for the amounts $\mathcal{M}'$ they deposit in the bank. This change would also make it much riskier for vendors to proceed with the sales to customers without waiting for the bank's acknowledgment, as they would be unable even to check user's credentials and identity without waiting for the bank's response. However, if the vendors were to adopt the policy of always waiting for the bank's acknowledgment anyway, using symmetric encryption for authentication would *dramatically decrease the processing time of all parties*.

4.2 A Deterministic Scheme

Instead of deciding whether to report a payment by casting a random coin, the polling decisions of a vendor can be made deterministically in the following way. The i-th payment of user U to vendor V triggers V to send the polling message to the bank if and only if

$$i = \lfloor k^j \rfloor \text{ for some integer } j$$

Furthermore, the polling message would contain the exact dollar amount the user has spent since the last poll. If each payment represents u amount of money, then at i-th payment, if $i = \lfloor k^j \rfloor$ for some integer j, the vendor would notify the bank that the user spent $(i - i_{\text{previous}}) * u$, where i_{previous} is the payment number that originated the previous polling message (of course, for $i = 1$, $i_{\text{previous}} = 0$). Similarly, the bank's counter x_U represents not the number of the polling messages received, but the sum of money reported by vendors about the user U.

For example, if $k = 2$, the vendor will report the first, second, fourth, eight, and so on, payments of each user. When the bank receives these reports, it updates the estimates of the user's spending to u, $2u$, $4u$, $8u$ and so on. Clearly, at every point, the bank's counter x_U will be at least $\frac{1}{k}$ of the amount the user spent with all the vendors. The bank sends out an alert if x_U reaches $k * \mathcal{M}$. If the bank requires $\mathcal{M}'$ of deposit and gives $\mathcal{M}$ credit to the user, the expected loss caused by a single thief is exactly the same as for the original proposal, i.e.

$k\mathcal{M} - \mathcal{M}' + E$. However, the amount of traffic is different. In this deterministic system, the number of polls about a user depends on the number of payments she makes to the same vendor. In our main proposal, the number of polls originated by an honest user who spends all her money is always a constant (denoted by c). This means that if the user makes small purchases with many vendors, the deterministic system is likely to produce higher traffic. On other hand, if the user makes purchases of value u with only one vendor, then in the deterministic algorithm, the vendor will produce $log_k \frac{\mathcal{M}}{u}$ polls. This number will be smaller then c if $\frac{u}{\mathcal{M}} > \frac{1}{k^c}$. For values of M and k which we give as reasonable choices in table 3, k^c is between 3 and 25, except for $W = 25$ and $k = 1.5$, when $k^c = 200$ or 300.

In another variant of this system, all losses can be prevented. If $k\mathcal{M} = \mathcal{M}'$, and each time a polling message is sent to the bank, the vendor stops further sales to the user until the bank acknowledges the message, there is no way for the user to overspend her deposit. However, in this version, communications traffic is doubled, since the bank has to respond to each polling message.

4.3 No Registration of Vendors

In the system we presented, the vendor has to register with the bank the first payment of every user that contacts him. This allows the bank to maintain a list L_U of all vendors a user U can be dealing with. If the polling messages about user U reach the threshold M, it's enough for the bank to send the "alert" messages to all vendors on list L_U. Alternatively, we can require that the bank would broadcast the alert messages to all the vendors in the system. The vendors would then be required to keep hotlists of suspected users, and instead of registering the first payment of a new user with the bank, verify whether this user is not on their hotlist. This change will greatly increase the time it takes for the bank to broadcasts all the alerts. Consequently, there will be also a much larger delay before an interested vendor receives an alert and stops dealing with a suspected user. On the other hand, the vendors will no longer be required to forward the registrations from the users, and the bank will not have to process them and respond with acknowledgment or denial messages.

The increase of traffic in this scenario can be expressed numerically, similarly to equation (4), as:

$$\text{volume of traffic per user} = (c + tM) + V * (2t + 3d)$$

These costs do not scale well, assuming that our micropayment system is adopted globally by hundreds of thousands of vendors. Still, for some realistic values of t, d, c, M, N, W, the amount of traffic due to this alternative proposal will be the same as in the original proposal for large values of V.[6] Furthermore, the increased communication requirements imposed on the bank can be satisfied by distributing the job of the central bank (see section 3.4).

Because the alerts will be broadcast to all vendors, there will be a longer average delay between the transaction that triggered the alert, and the moment when most vendors receive the alert message. This would result in bigger loss caused by thieves, in addition to the increased communication requirements for the banks. Hence, this alternative scenario

[6] If $t = d = 0.01, c = 2.3, M = 7, N = 100, W = 25$, then for $V = 1000$ the traffic volume is the same as in the main proposal.

would increase the cost of the system, in exchange for cutting down the processing time and communication delays experienced by the users and vendors in their transactions.

5 Conclusion

We presented a new micropayment system, which prevents overspending by probabilistic polling of customer's spending. This technique enables a micropayment system which has small communication overhead and delays, maintains high cryptographic security, and reduces the costs of the system by minimizing the amount of possible overspending.

References

[AMS96] R. Anderson, C. Manifavas, and C. Sutherland. Netcard - a practical electronic cash system. In *Fourth Cambridge Workshop on Security Protocols*. Springer Verlag, Lecture Notes in Computer Science, April 1996. http:// www.cl.cam.ac.uk/ users/ rja14/.

[BGH+95] M. Bellare, J. Garay, R. Hauser, A. Herzberg, H. Krawczyk, M. Steiner, G. Tsudik, and M. Waidner. *iKP*- a family of secure electronic payment protocols. In *First USENIX Workshop on Electronic Commerce*, New York, 1995. http://www.zurich.ibm.com/Technology/Security/extern/ecommerce/.

[GMA+95] S. Glassman, M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro. The millicent protocol for inexpensive electronic commerce. In *Proc. 4th International World Wide Web Conference*, 1995. http://www.research.digital.com/SRC/millicent.

[GS96] E. Gabber and A. Silberschatz. Agora: A minimal distributed protocol for electronic commerce. In *Second USENIX Workshop on Electronic Commerce*, pages 223–232, Oakland, California, November 1996.

[HSW96] R. Hauser, M. Steiner, and M. Waidner. Micro-payments based on ikp. In *14th Worldwide Congress on Computer and Communications Security Protection*, June 1996. http://www.zurich.ibm.com/Technology/Security/publications/1996/HSW96-new.ps.gz.

[JY96] C. Jutla and M. Yung. Paytree: "amortized signature" for flexible micropayments. In *Second USENIX Workshop on Electronic Commerce*, November 1996.

[NM95] Clifford Neuman and Gennady Medvinsky. Requirements for network payment: The netcheque perspective. In *Proc. of IEEE COMPCON*, March 1995. ftp://prospero.isi.edu/pub/papers/security/netcheque-requirements-compcon95.ps.Z.

[Ped96] T. Pedersen. Electronic payments of small amounts. In *Fourth Cambridge Workshop on Security Protocols*. Springer Verlag, Lecture Notes in Computer Science, April 1996. Tech report: DAIMI PB-495, Computer Science Department, Aarhus University, August 1995.

[RS96] R. Rivest and A. Shamir. Payword and micromint: Two simple micropayment schemes. In *Fourth Cambridge Workshop on Security Protocols*. Springer Verlag, Lecture Notes in Computer Science, April 1996. http:// theory.lcs.mit.edu/ ~ rivest/ publications.html.

[Yac97] Y. Yacobi. On the continuum between on-line and off-line e-cash systems. In R. Hirschfeld, editor, *Financial Cryptography*, Anguilla, West Indies, February 1997. Springer Verlag, Lecture Notes in Computer Science.

A Reimbursing Vendors for Overspending Customers

The bank could set up a simple policy towards the unlucky vendors who were selling their goods to an overspending user: It could not reimburse them at all, or reimburse them fully. The first strategy would lead to higher then necessary average loss per vendor per thief. The second strategy seems to fail if dishonest vendors cooperate with dishonest customers and request excessive reimbursement. What we want to prevent is the possibility that a group of dishonest vendors refrains from sending polls about a dishonest user, the user "spends" her maximal amount $\mathcal{M}$ with each of those vendors, and then the vendors request reimbursement from the bank. What we need then is a mechanism in which the interests of a dishonest user and the interests of the vendor would not coincide.

The reimbursement procedure we propose at the end of section 2.2 meets this requirement: The vendors get reimbursed proportionally to the number of polls they produced. The vendors might try to improve their chances of reimbursement by "overpolling" the overspenders (i.e. producing too many polling messages). However, since the interest of the overspenders and the vendors collide, they will not cooperate, and hence a vendor would have to guess whom to overpoll, or to overpoll everybody. This would increase the communication cost of the system, and maybe derail the probabilistic monitoring mechanism altogether.

However, the bank can easily control whether the vendors produce too many polls on the average, and ban such vendors from the system. The bank could periodically check whether x_{UV} (number of polls produced by vendor V about customer U) falls within some error bracket of $\text{Exp}[x_{UV}] = (u_{UV} * f_U) * m_{UV}$ (m_{UV} is the number of same-cost transactions between U and V, and $u_{UV} * f_U$ is a constant that V should use to compute the probability of polling U's payments). The bank could normalize the difference $e_{UV} = \text{Exp}[x_{UV}] - x_{UV}$ by dividing it by u_{UV}. Then if in the long run the sum of normalized e_{UV} grows more then a random walk, this is an indication for the bank that the vendor is trying to slant the eventual reimbursement procedures by producing more polling messages than specified in the protocol.

Notice that the same mechanism can be used to verify that the vendors are not "under-polling" the purchases of users, trying to save on communication in this way.

As an additional control on the vendors, one could require that they obtain from the bank (during the registration transaction for each user) the seed for the random generator that determines when to report a transaction. A dishonest vendor could still cooperate with a thief by testing each seed to see which ones require few polling messages. However, such behavior would usually be easy to detect, since each seed requires a purchase and a new registration.

B Computations for Performance Analysis

In this section we analyze the relations between parameters in the probabilistic polling system. The reader should refer to the notation in section 3.1 for reference about the meanings of the variables.

Let's assume for a moment that all purchases of some user have the same unit worth u. Let $m = \frac{\mathcal{M}}{u}$, i.e. the maximal number of purchases at the unit cost u that the honest user

can make. The probability of forwarding a single purchase that the vendors should use is then $p = f * u = \frac{c}{\mathcal{M}} u = c\frac{u}{\mathcal{M}} = \frac{c}{m}$. Let $g(z)$ be the probability that x_U becomes M at the z-th purchase of user U:

$$g(z) = \binom{z-1}{M-1} p^M (1-p)^{z-M}$$

Notice that $\sum_{z \geq M} g(z) = 1$. Now we can express the probability that $x_U \geq M$ for the honest user, i.e. for the user who spent up to m of u-valued coins:

$$d = \sum_{z=M}^{m} g(z) \tag{5}$$

Approximating k

The expected number of purchases at which the dishonest user is caught is:

$$s_u = \frac{1}{1-d} * \sum_{z>m} (z * g(z))$$

(The dollar amount of spending will be $s = u * s_u$). We multiply the expectation by the $\frac{1}{1-d}$ factor because we take the expectation over only the users who make more than m transactions. Since we want only small d's, i.e. at most 1%, this factor is between 1 and 1.01. Since, $\sum_{z \geq M} (z * g(z)) = \frac{M}{p}$, we approximate:

$$s_u = \frac{1}{1-d} \left(\frac{M}{p} - \sum_{z=M}^{m} z * g(z) \right) \simeq \frac{M}{p}$$

We assume that the important parameter one will want to minimize is the proportion between the expected amount of money a thief can spend, and the amount of the thief's credit $\mathcal{M}$ in the bank:

$$k = \frac{s}{\mathcal{M}}$$

which we can approximate now as:

$$k = \frac{u * s_u}{u * m} \simeq \frac{M}{pm} = \frac{M}{c}$$

Approximating d

We derive our approximation from the summation formula (5). First notice that for $0 \leq i \leq m - M$:

$$\binom{m-i-1}{M-1} \leq \left(\frac{m-M}{m-1} \right)^i \binom{m-1}{M-1}$$

which leads to the following bound on d (we substitute $i = m - z$ in the definition of d):

$$\begin{aligned} d &= \sum_{i=0}^{m-M} \binom{m-i-1}{M-1} p^M (1-p)^{m-i-M} \\ &\leq p^M (1-p)^{m-M} \binom{m-1}{M-1} \frac{m-1}{p+M-pm-1} \end{aligned}$$

Taking $p = \frac{c}{m}$, and using the inequality

$$\binom{n}{k} \leq \frac{1}{\sqrt{2\pi k}} \left(\frac{ne}{k}\right)^k \tag{6}$$

which holds for all k and n, we derive:

$$d_{ap} \leq \frac{c^M e^{M-c-1}}{\sqrt{2\pi}(M-c-1)(M-1)^{M-\frac{1}{2}}}$$

Comparing the direct summation and the approximation formula above, we observe that for the variable ranges that we are interested in, i.e. $k \in \{1.3, \ldots, 5\}, c \in \{5, \ldots, 100\}$, the actual d was always no more then $0.8 * d_{ap}$, as expressed in the resulting equation (2) in section 3.1.

Although we derived equation (2) by fixing the unit u (and consequently the amount m and the probability of forwarding p) of a single transaction, the values k and d, depend only on the choice of c and M, by equation (1) and (2). This is not entirely true, in the sense that the approximation (6) which led to equation (2) works best for $n \rightarrow \infty$, i.e. in our case for large m, and hence for small u. However, this inequality is sharper for smaller n, which means that the bigger the u, the smaller the proportion $\frac{d}{d_{ap}}$. In other words, if the user's transactions are large fractions of $\mathcal{M}$ (say, $\frac{u}{\mathcal{M}} \geq \frac{1}{4c}$, and, hence, $m \leq 4c$), the probability d that an honest dealer will be halted by a false alert is much smaller then d_{max} from equation (2).

C Tables

We supply the tables with the data used in the analysis of increase in communication volume (section 3.3).

k=1.5	$W=1$	$W=5$	$W=25$
$N=10$	$7-11.5$	17	NA
$N=70$	$8-29$	$20-35$	$64-68$
$N=500$	$8-29$	$20-37$	$68-77$

k=3	$W=1$	$W=5$	$W=25$
$N=10$	$3.5-4.25$	11.5	NA
$N=70$	$3.5-4.5$	12.5	53
$N=500$	4.5	12.5	53

Fig. 2. Communication Costs of Polling (number of messages per user)

k=1.5	$W=1$		$W=5$		$W=25$	
	M_{min}	d	M_{min}	d	M_{min}	d
$N=10$	6	.21	12	.11	NA	
$N=70$	6	.21	9	.15	21	.05
$N=500$	6	.21	9	.15	18	.06

k=3	$W=1$		$W=5$		$W=25$	
	M_{min}	d	M_{min}	d	M_{min}	d
$N=10$	3	.08	6	.016	NA	
$N=70$	3	.08	5	.027	8	.006
$N=500$	3	.08	5	.027	8	.006

Fig. 3. Values of M_{min} and their corresponding d's

k	1.5	2	3	5
$M_{d=0.01}$	40	15	7	4

Fig. 4. Minimal values of M for which $d \leq 0.01$, for different k's