EVALUATION OF MICROPAYMENT TRANSACTION COSTS

Ioannis Papaefstathiou
Institute of Computer Science (ICS)
Foundation of Research and Technology – Hellas
P.O. Box 1385, GR 711 10 Heraklion, Crete, Greece
ygp@ics.forth.gr

Charalampos Manifavas
Barclays Capital,
Canary Wharf, London E14 4BB, UK
harryman@otenet.gr

ABSTRACT

Electronic money, in general, is used for trading over electronic channels such as the Internet, or mobile networks. Micropayments, in particular, can be used for the purchase of low value intangible goods (i.e., non-physical assets like data and information). For these products, the use of payment instruments like on-line credit cards tends to be more expensive than the actual product. For this reason, a micropayment mechanism needs to keep the cost of the individual transaction low. There are many factors that contribute to the cost of running such a system and they come from the different disciplines that are involved in an electronic-payment (e-payment) system. Therefore, in order to design an efficient micropayment mechanism it is very important for the designer to consider and minimize each of these cost factors. In this paper, we identify the most interesting of the technological factors and we present ways to minimize them. We argue that one of the main reasons that the micropayment systems have not been widely used yet is that their cost has been forbiddingly high; this is probably due to the fact that there are certain cost factors that have been undervalued by the majority of e-payment developers.

Keywords : E-commerce, E-payments, micropayments

1. Introduction

Micropayment systems are used for the purchase of low value intangible goods, and have recently been reconsidered as a useful technology, mainly due to the growth of mobile networks. Independent service providers that operate mobile networks often charge users for access to low-value services in the fixed network but may not have long-term contractual relationships with their users [Ulema & Kozbe, 2003]. Other new applications that need a mechanism for charging for small transactions include storage systems [Roscoe & Hand, 2002] and news distribution [Abere & Wombacher, 2001].

Since micropayment systems are used for the purchase of extremely inexpensive items, it is vital for them to keep the cost of the individual transactions low. In general, providers of low-value goods or services should be able to use a micropayment system without the risk of having profit margins on small-value transactions wiped out by the cost of providing these transactions and without noticeable system performance loss.

Several papers on electronic money systems include a discussion of the characteristics the authors consider desirable when designing such systems [Odlyzko, 2003; Micali & Rivest, 2003; Herzberg, 2004; Kytöjoki & Kärpijoki, 1999]. However, none of them recommend the same characteristics, and, more importantly, none has provided metrics for the overall cost of such systems, even though this factor seems to be the most vital one. These other authors also select an arbitrary subset of the large number of possible criteria when they evaluate their proposed mechanisms.

In this paper we concentrate on the cost of the overheads involved in running an e-payment system, and in particular we concentrate on micropayment systems. This cost has to be a small percentage of the actual value of a product or service and, in the micropayment case, this value is expected to be very small itself. In order to design a mechanism with reduced costs, one needs to consider and minimize each of the cost factors. There are also other important issues which are business related and are not covered in this paper. The reader can refer to van Someren, et al. [2003] for a thorough description of those issues. We concentrate on the technical aspects, mainly, because we believe that this paper can be very useful to the developer of a micropayment system who will be responsible for implementing a framework according to a given set of operational specifications (i.e., assuming other business costs have already been minimized).

This paper also presents the different approaches to cost efficiency taken by various micropayment protocols and the consequences of their design decisions, and suggests ways of bypassing some of their problems. Finally, it presents an optimal micropayment model that minimizes the sum of these technical factors and thus increases the potential effectiveness of such a system.

In the rest of the paper, the different cost factors are categorized and described according to their underlying aspects. Within each cost category, the purely technical issues discussed in this paper are clearly distinguished from other issues involving the business itself and the overall operating environment.

2. Fixed Technical Costs

The fixed costs depend on the overheads of (a) the hardware (e.g., smartcards), (b) the software (e.g., electronic wallets), (c) the installation procedure, and (d) the maintenance of accounts (account openings, closures, fund transfers, balance enquiries, account statements, etc).

Account maintenance can be avoided by keeping the customer-merchant relationship transient. For one-time or infrequent interactions this will be more convenient and eliminate a potential performance bottleneck. Otherwise, merchants would have to create, maintain and bill accounts for possibly a large number of users, for a long period of time (see next section as well). Customers would have to do the same for each merchant. But since micropayments are almost always used on small but very frequent transactions, the cost of maintaining the account information might be a large fraction of the cost of the system and therefore it should certainly be taken into account when designing such a mechanism. Systems that have not paid attention to this issue and have finally faced severe problems due to it include FirstVirtual and Beenz [Hurwicz, 2001].

Moreover, a payment system should be able to accommodate advances at the protocol level as well as new payment types (e.g., plug-in software upgrades), because of the significant investment new infrastructure components might entail. A number of wallet designs that have been proposed [e.g., Kytöjoki & Kärpijoki, 1999] are typically targeted for specific financial instruments and operating environments. As a result they cannot accommodate discount e-coupons that merchants may introduce, and as they are frequently web-centric, their design cannot be applied to alternative devices (such as mobile network devices). Furthermore, some wallet architectures do not support the notion of a session, making thus the execution of micropayments more costly due to transaction parameter renegotiation [Bellare & Rogaway, 1997]. Both of those issues should be taken into account when designing a micropayment system so as to reduce the probability of significantly increasing the cost of the system when trying to “plug” them in afterwards.

3. Storage Costs

3.1 Legal and Financial

Business records, in general, are kept for the period during which transactions can either be challenged by the customers or audited by tax inspectors. For example, banks must keep records for a period of time set by the law [6 years in the UK: Anderson & Lee, 1999]. The large number of small transactions in a micropayment system suggests that huge logs might have to be kept by the participating entities (merchant, issuer, broker, etc.), which would significantly increase their overall storage requirements.

However, it should not be necessary to keep records of all tiny payments for a long time, especially if legal requirements for record keeping, particularly for micropayments, are relaxed. One remedy would be to allow for signatures to be exchanged on the agreed balance before the records are archived away (stored off-line). Off-line operation further reduces the issuer’s on-line storage requirements [Aberer & Wombacher, 2001].

3.1.1 Divisibility Costs

One of the many properties that have been proposed as crucial for any payment system is divisibility [Goh, Yip, & Ngo, 2003; Markose & Loke, 2003]. This is a feature of a coin-based e-cash system that allows a coin to be divided into lower denomination coins (the total value of which is equal to the value of the original coin). Divisibility is one way to address the problem of how to provide exact change for a payment.

Real cash is not divisible, but this is not a serious problem because cash is transferable. The pool of coins of different denominations, to a certain extent, can be kept replenished (in a shop’s till, for example).

In a system that does not support divisibility a payer would have two options. The first is to store a large reserve of coins of each possible denomination. Loss of interest and fear of this large number of coins being stolen or lost are the side effects of this approach. The second is to allow withdrawal from a central (bank) account of a coin of the exact amount for each payment. In this case, interaction with the bank would make all transactions on-line. Alternatively, the burden, in any of the above forms, could be put on the payee who would be responsible for paying back to the payer the difference. In that case, if coins are not transferable, the payer should be allowed not only to withdraw coins but also to deposit them in a bank.

Indivisible coins have to be aggregated to get the amount needed for a payment. If coins are in small denominations, a large number of coins may be required to buy anything even moderately expensive. Processing a large number of e-coins can introduce communication delays and computational costs. Additionally, if the coins are to be carried on a smart card, large numbers of coins require a card with a larger memory, which increases the fixed costs.

Law, Sabett, and Solinas [1996] presents three examples of off-line divisible e-cash schemes are presented. Measures are taken as to identify double-spending of parts of the coin but at the cost of a more complicated protocol, longer transaction time and additional storage. As a result, those systems operate in a less efficient manner. Although these schemes are untraceable, payments made from the same initial coin may be "linked" to each other, meaning that it is possible to tell if two payments came from the same coin and hence the same person. This does not reveal the payer's identity if both payments are valid but revealing the payer's identity for one purchase would reveal that payer's identity for all other purchases made from the same initial coin. This latter feature may or may not be desirable.

The divisibility property was mainly required to battle the considerable costs of a large and diverse wallet in the original coin model. Therefore, its main advantage is that it significantly limits storage and communication requirements. On the other hand, in order to make sure that the overall system cost is kept at a low level, the designer should use very simple underlying protocols that are as stateless as possible, both for anonymity and to control double-spending of the sub-coins. If those protocols are complex the designer faces the risk of undermining the system's efficiency.

3.2 Technical costs

In most proposals the minting entity maintains a list of spent coins (to catch double spenders), which grows with time. If the coins expire, then that entity can reuse the corresponding database entries from time to time and keep the whole database manageable. Some systems [e.g., Mondex®, 2004] allow private transactions between customers in which case there need be no central record of small transactions.

It is hard to estimate the cost per micropayment transaction, since databases (and the whole system) must be designed to absorb "rush hour" peaks gracefully. According to Yacobi [1997], if we assume that (a) the peak-to-average ratio is 100, (b) that very large databases cost about $10M annually, and (c) that they can do 150-500 transactions per second, the overall cost per transaction is about 10-20 cents, which is not cost effective for sub $1 transactions. When the frequency of incoming requests is high, the necessary access bandwidth may not be satisfactorily provided by secondary storage systems such as disks, since head contention becomes the limiting factor. On-line storage access should thus be, in some cases, effectively RAM. A caching mechanism could ensure that only the most frequently used data is maintained in RAM.

4. Computational Costs

If a transaction requires an enormous amount of computation, the cost of the transaction will be dominated by the cost of the computing time. Such operations will also result in additional delay, the factor that upsets the user most. Off-loading the server CPU is important because if the server application is less computationally intensive, the operators of the system will need fewer CPUs in order to implement an effective service center. But it is also important not to stress the client's CPU either, so that it can perform CGI and other user processing. Processing a transaction is essentially equivalent to making a contract so usually both sides will need to sign something and verify each other's signature. Public key computations are costly and should be prevented or at least kept to a minimum. These issues are addressed in PayWord and Micromint [Rivest & Shamir, 1996]. Such schemes try to aggregate many small payments into fewer, larger payments, so that the processing cost for the bank is relatively small.

In order to lower the processing cost and in a scenario where a customer makes many low value purchases to the same merchant, a signature re-use mechanism may be employed. This can be achieved with the use of a one-way, "collision resistant" hashing [Bellare & Rogaway, 1997]. Some protocols allow grouped payments to be made in batches. The hashes of multiple messages can be concatenated by a specific entity and then this same entity can sign the entire combined hash to prove that all the combined messages have been authenticated.

Additionally, computational costs can be lowered by carefully investigating which operations need not be performed every time a new transaction takes place. For instance, when a signature is verified against a certificate it is necessary to recursively verify the certificate itself. If this were done every time a connection was established it would add to the cost of processing. However, in practice many certificates are seen regularly. It is only necessary to verify each certificate once, as long as there is a secure way to make sure that the certificate has already been verified and that has not been revoked.

Finally, other optimizations can be made if the specific features of protocols like SSL are employed. For example, the SSL specification allows for session secrets to be reused for some time after an initial connection, thus removing the need for new key negotiation.

Another expensive part of any transaction, in terms of processing, is the settlement phase, where the actual value is transferred from the customer to the merchant [Herzberg, 2004]. A cost effective micropayment system should completely separate the settlement phase from the payment, so that a single settlement can be made for a large number of micropayments. Settlement can be done either before (pre-payment) or after (post-payment) a payment is made. Any conventional instrument may be used for the settlement (e.g., a credit card), and if the system is designed such that a large number of small transactions are grouped into just one "macrotransaction", the high computational complexity of the conventional payment systems can be tolerated. In general, pre-paid systems have the advantage of reduced fraud risk, which may make them cheaper to operate than post-paid systems.

5. Communication Costs

Many of the proposals for micropayment mechanisms have focused on completely eliminating public key operations. But according to Herzberg and Yochai [1997], public key operations are only a fraction of the delay and certainly not the bottleneck; the main performance bottleneck is the communication delay. Several proposals and mechanisms that eliminate public key operations, such as Micomint and FirstVirtual [Rivest & Shamir, 1996], do this at the cost of additional communication messages. The delay when establishing a connection to a third party is likely to cancel out the customer's gain from not having to compute a digital signature for the payment. However, the establishment of a third party connection may be a more efficient approach for the merchant, who is expected to be the bottleneck in such transactions.

The number of messages exchanged to complete a transaction has several implications on the speed, cost and complexity of the system. A protocol with a larger number of messages per transaction, such as the one used by Flooz, usually implies that a larger number of parties are involved (e.g., issuer, trust intermediaries, cooperating billing gateways, etc). Increasing the complexity of a transaction not only makes it more expensive, but it also increases the number of possible points at which the transaction can fail and the time it takes to find a problem and report the exact error. Moreover, the less the system depends on on-line involvement of third parties, the more resilient against denial of service attacks it is [Stallings, 2003] Increased usage of network resources will also add to the cost of the micropayment transactions. There are several ways to minimize the number and cost of messages.

- Off-line payments reduce communication costs by allowing the customer and the merchant to complete the transaction without having to consult a third party.
- Probabilistic schemes may not completely eliminate but they reduce the requirement to consult a third party. They can be divided into two categories: probabilistic checking/audit [e.g., Jarecki & Odlyzko, 1997] and probabilistic payment. A possible criticism of such schemes [such as the one in Wheeler, 1996] is that they are a weak form of gambling, which is forbidden by law in several countries [Lipton & Ostrovsky, 1998].
- Public keys and certificates can be pre-distributed. In this case the storage cost can be modest. Moreover, certificate revocation servers should not be engaged in every transaction (see also the previous section).
- Allow for withdrawals and deposits to be done less frequently. For small payments there is no need for the customer to request funds from the bank that holds the account. Each customer could have a daily spending limit and, as long as this is not exceeded, the merchant can be relatively sure that the bill will be paid [Herzberg & Yochai, 1997].
- Minimize the number of total bits transmitted per transaction between customers, merchants and the bank/broker. Depending on the network environment a few extra bytes may introduce additional message fragmentation at the lower network levels and increase the storage requirements. Provided that message length remains within reasonable bounds the overhead of a short message is similar to a longer message in the widely used TCP or UDP protocols.
- Smartcard technology has created a new opportunity. It enables "stored-value" applications that may reduce the needs for on-line authorization and its associated costs, as well as reducing fraud [Herzberg, 2004].
- Payment orders can be piggybacked next to the information requests [e.g., Gabber & Silberschatz, 1996]. If the characteristics of the link (such as the maximum and/or the optimal packet length) are known beforehand, this approach can save a large number of network packets.

Another scenario that demonstrates the importance of communication costs in overall system effectiveness is the following. In the future, traffic may be prioritized depending on what application it comes from. Currently, the majority of Internet traffic is web traffic (more than 70% in US domestic links at the end of 1997 as described in [Thompson, Miller, & Wilder, 1997]). If a lower priority is assigned to web traffic than packet voice, for example, then micropayments based on web protocols will suffer extra delays or have to pay an additional cost to use a high-priority protocol. On the other hand, if the designer uses JAVA sockets, which very often carry time-critical data, for micropayments communications, these communications are less likely to be assigned a low priority in the future.

6. Administrative Costs

In this section, we present a number of factors that contribute to increased administration costs when running an e-payment system. We start with the issues related to the overall financial environment the micropayment system has to utilize.

6.1 Financial Environment - Billing

Billing is one example of administrative cost. Traditional billing using paper documents is inappropriate for small-value transactions since the associated cost is in the order of tens of cents [Daswani, Boneh, Garcia-Molina, Ketchpel, & Paepcke, 1998]. Thus, a way to minimize the billing cost is to have the banks using electronic billing, and attach a machine-readable list of transactions to each bill. Customers are then able to verify their bills by comparing the list of committed transactions with their own records. In this way, the cost of billing will be minimized.

6.2 Commissions

It is important to make a payment system profitable for those who administer it (i.e., the broker). However, prospective merchants should not find it expensive to participate in a given system. If it is inexpensive and easy, more people will become merchants, increasing the content available to customers. All of the early micropayment systems failed to gain a critical mass of merchants and therefore a critical mass of consumers. However, a micropayment system might come into widespread use if it is pushed by a large marketing power (e.g., a collection of banks, publishers, ISPs, etc.).

Banks or other intermediaries may charge a fee every time an account is accessed or a payment instrument used. A fee plays the role of an economic incentive (profit and benefit) for all the relevant parties to cooperate. Nevertheless, this will add to the overall cost of operating the system. Therefore, in order to reduce this factor, practical ways to manage commissions for small value transactions need to be found. Obviously, one approach is to be able to group multiple micropayment transactions to a single macropayment and pay the commission only once for each group of micropayments. Another, more revolutionary way, might be to push the standard credit cards operators, which have high profit margins due to limited competition, to reduce their fees for transactions that are executed entirely on-line.

6.3 Interoperability

It is likely that several forms of e-payment instruments (analogous to credit cards, personal cheques, real cash, etc.) will emerge. Customers will select the one that best suits their needs for a given transaction. Alternative forms will provide trade-offs with respect to the timing of the payment, performance requirements, auditing requirements, etc. A successful payment system implies a broad (ideally universal) acceptability, that is, the ability to support any transaction between any pair of buyers and sellers.

These instruments would probably be integrated into a common framework that would allow for interoperability[1]. In such an environment it is important that well-accepted and relatively fixed mechanisms (e.g., clearing stations) exist for converting the various forms of money from one form to another. In particular, compatibility between older and newer e-payment systems would be desired, as well as between older and newer versions of a specific system. One way to minimize the bottlenecks triggered by the need to support what it is called “backward compatibility” is to allow the conversions between the various forms of money to be done outside the system via, for example, bank accounts. This feature is extremely important for a micropayment system since the cost of each transaction should be kept to the sub-cent rate throughout the life of the system. As a result, each new version of a mechanism should add improved functionality without significantly increasing the cost of a transaction.

In a micropayment system, it is important to avoid restrictions on transactions between buyers and sellers, such as direct trust, physical proximity, and common providers. Any restriction would limit the customer base for a merchant. Most e-cash proposals use a single bank. In practice, multiple banks are needed for scalability, and because not all users will be customers of a single bank. In such an environment, it is important that other banks accept currency minted by any particular bank.

[1] Also, compatibility, convertibility, exchangeability.

A customer may use one type of e-cash and a merchant may use another. Either the two will be unable to transact or there should be a currency exchange mechanism. As currency exchange adds both time and cost, a surcharge will be added to the cost of the purchase effectively discounting the value of the currency. A way of bypassing this problem is to design the system so that it can be plugged in an existing currency exchange mechanism. This would be more cost effective than including the exchange program in the micropayment system itself, since the added cost of the converter might make the micropayment system prohibitively expensive.

Acceptability is a challenge to any new payment mechanism. However, deliberate incompatibility may be employed as a marketing tactic by some vendors.

6.4 Global environment

6.4.1 Patents

Patents present yet another cost factor at several levels. For example US patents 5191573 and 5675734 claim the concept of "selling electronically ... through telecommunications lines, the desired digital video or digital audio signals"—in short, pay-per-view over the Internet [Wayner, 1999]. Interested parties are currently evaluating what those patents mean to their plans for selling music over the Internet[2].

Especially in micropayment systems where the crucial factor for the success of the system is its cost, costs associated with patents should be carefully analyzed before designing such systems, and if possible anything already patented should be avoided.

6.4.2 Certification

A trusted key certification authority (CA) should be established to register and certify all principals who have public keys in the payment system. The CA also revokes lost or stolen keys and must make sure that revoked keys are immediately reported to all principals. Every principal involved in the payment has to check the revocation list whenever a transaction is processed. Any unavailability of the revocation list or delay in delivery of the revocation list will leave the principals open to fraud attacks.

The CA incurs costs in a number of ways. The cost of issuing certificates is proportional to the number of certificates issued. A large part of the issuance cost is the off-line validation of the information (at subscription time) that is to be included in the certificate. The CA may also incur costs in order to periodically re-validate the certificate information or re-issue certificates whenever certain attributes change. The CA must also maintain a certificate revocation list (CRL), which also incurs significant costs.

Merchants risk monetary loss by accepting certificates. A merchant who trusts an invalid certificate risks a loss (non-payment) from the transaction. While certificates remain valid (neither expired nor revoked) they may be used repeatedly, so a single "bad certificate" can be used in multiple transactions. Merchants also incur a cost when checking the CRL. A number of different access options are available to them, each with differing cost and risk implications.

The cost of operating the certificate infrastructure depends heavily on assumptions about how CRL accesses are handled. Two possible organization options are:

1. The merchant contacts the CA on every transaction to verify that a certificate has not been revoked which results in much higher infrastructure costs for the CA;
2. The CA regularly (e.g., once a day) sends a complete CRL to merchants.

In the case of micropayments, because customer spending is assumed to be smaller, the cost of fraud is less. Consequently, a micropayment system should probably choose the option of issuing shorter and less expensive certificates even if this increases the possibility of fraud. Based on this same fact, the certificate lifetimes may be kept long no matter what the effectiveness of frequent checking.

6.4.3 Exception Handling

For traditional payments, exception-handling costs can be as high as $30 per transaction. Exceptions include disputes and refunds. Exception handling costs involve both technological and human investments. In the case of micropayments, a single customer service call can easily wipe out an entire year's profit on that customer's account.

Several e-payment proposals put emphasis on different aspects of the exception handling problems while the majority fail to deal with issues such as interruptions and dispute handling. A real system certainly needs to handle interruptions. Although disputes are usually not mentioned in the literature, several security protocols have been invented to avoid disputes, while storage and verification of data needed for dispute handling should be included in any implementation.

[2] Today many people claim patents by taking some everyday occurrence and adding to it the phrase, "with a computer network." That's why no one ever thought of getting a patent on selling records for cash, but someone wants to patent a mechanism for selling audio files over the Internet.

Most systems deal only with the collection of payment evidence (e.g., a payment receipt in the form of a digital signature). In NetCard [Wheeler, 1996], for example, the combined hashes mentioned above constitute evidence that is retained in the event of a dispute. A successful micropayment framework should also include proofs that demonstrate that the collected evidence is enough to win any dispute. It is assumed that such evidence can be used in a dispute resolution procedure external to the system. This may not be practical if the system needs to make decisions based on the outcome of disputes. Obviously, such mechanisms should be incorporated in a system only if they are absolutely essential, because they significantly increase storage costs (see section 3).

Atomicity deals with how a protocol reacts when it is interrupted. A protocol is called atomic if a payment can only be completed in full or not completed at all. Interrupting the protocol should trigger the same result as if the transaction had never been started. A more severe form of atomicity deals with the delivery of the goods (transaction atomicity); payment is received if and only if the product is delivered. This is an awkward feature since there is always at least one step where one participant has received an output and another one has not. However, it should be guaranteed that nobody can draw advantage from interrupting transactions and that honest participants do not suffer permanent loss. Since the cost of the system should be kept very low, the latter feature should be guaranteed by a very simple mechanism (for example, by eliminating any transaction that has been interrupted and asking the user to re-execute it). Although this may upset the user, if it is infrequent enough and if the payment is finalized before the product or service is delivered, it will probably prove adequate.

Apart from disputes about interruptions, there can be disputes regarding previously completed transactions. There are several points where a dispute can arise. There may be disputes about ordering, about the subsequent payment, or about the delivery. For example:

- A customer claims that the merchant obtained the payment but has not delivered the goods or that the wrong goods were delivered (e.g., wrong size, etc).
- A customer claims that he or she has not authorized the bank to deduct money from his or her account for a payment.
- A merchant claims that the bank failed to add money to his account although properly instructed to do so.

A complete framework for handling all types of disputes involves technical, legal, and policy aspects. When there are no receipts in a transaction there is no method of guaranteeing that a customer gets the item that he has ordered, or even any item at all. Customers may complain to the merchant's broker, but the broker is unlikely to act until a significant number of customers have complained. A system is called "loss tolerant" if there are specific transactions for recovering money that has been lost. The payment mechanism should deter individuals from making false claims, and do that in a very cost effective way. This is one of the few cases that the developer might have to add costly components to the system so as to reduce those claims, otherwise the whole framework might be at a risk if someone finds the weak link of the system and asks for a large number of false claims. Obviously, the best way to cope with this issue is to analyze both the cost of the possible false claims (which might be extremely difficult, though) and the cost of the additional system modules and find the optimal point.

In pre-paid systems like Millicent [Manasse, 1995] all the unused part of the vendor scrip should be refunded back to the customer and this refund procedure can be assumed to be an exceptional case since it is not an integral or necessary part of any mircopayment transaction. The refunding process should be carried out by using a usual macropayment system and this will be a much more expensive protocol than the micropayment. It should be noted that both the vendor and the customer need to be involved in the refunding process and this complicates the protocol. Of course, this also increases the computational and communicating costs and thus it should be guaranteed that this refund cannot happen very often, otherwise it can wipe out the profit produced by the micropayment system.

6.4.4 Credit Cost

Another issue that affects the cost of a micropayment system is the adoption of a pre-paid or a post-paid model. In pre-paid systems (cash-like) the payer's account is debited before a payment can be initiated. The money has been paid in advance either to the bank or to the vendor. In post-paid systems (cheque-like, bank-card-based or account-based) the payer's account is debited after the payee's account is credited.

In pre-paid systems, vendors are protected since they actually receive the money before delivering the requested items, which in turn puts customers at a disadvantage. Post-paid systems offer more protection to customers; the risk in a post- paid system is the customer's credit. For micropayments this is not as crucial as it is for macropayments. However, credit risk and cost can be eliminated if the payment system decides to allow only pre-paid accounts, and this seems to be the optimal approach for the low cost micropayments.

7. Technical Environment

7.1 Availability

Continuous availability of global payment infrastructures would be necessary for the smooth running of the on-line economy. Availability encompasses several aspects. Reliability of the network in terms of speed and message integrity is one factor. Robustness of devices, like smartcards, is another factor. Card manufacturers guarantee a certain level of reliability in terms of card life and the number of transactions that can typically be performed.

Transaction protocols should be resilient to delayed or corrupted messages as a result of communication problems (e.g., accidents, vandals), or system interruptions (e.g., hackers, criminals). Payment systems should not have a single point of failure. For example contingency arrangements should be made for central systems that store cryptographic keys or perform other critical functions (e.g., on-line authorization, clearing).

In micropayment systems it is even harder to guarantee continuous availability of the infrastructure since:

1. There is an extremely large number of small transactions these systems process in any unit of time. As a result, they are often more vulnerable to system and network failures.
2. The cost of the whole system should stay low, and thus it is often much more difficult to acquire and maintain the necessary fault-tolerant hardware and software infrastructure
3. The cost of each transaction should be kept minimal and therefore the transaction protocols should stay as simple as possible. Thus, it is difficult to include mechanisms for handling efficiently any communication failures or system interruptions.

Unfortunately, there is no simple approach that can solve all the above problems, and more importantly the mechanisms that can cope with some of those issues, increase the complexity of the other ones. For example there are mechanisms that can handle the communication failures (3) at a very reasonable operational cost, but they need additional hardware so they will increase (2). Similarly, you can buy relatively inexpensive fault-tolerant systems that just use mirroring to cope with the problems in (2), but those systems increase significantly the number of transactions to keep all the mirrors up-to-date at any given time and therefore amplify the problems caused by (1). As a result, in order to find the optimal cost solution regarding availability, a system designer needs to analyze in parallel all three of the above factors and identify the most advantageous point.

8. Publishing Costs

For information products with very low intrinsic cost it is obviously important to minimize the cost of publishing and advertising. MiniPay (described above), offers a relatively cost effective solution. A tool parses a website and transforms it to a new format where paid links are visually accompanied by the appropriate fee. This allows content providers to use any HTML authoring tool to build their initial sites, and then MiniPay will change them to the appropriate micropayment-aware layout. In the early systems, per-fee links were not integrated in the web page itself, but instead, an additional 'pop-up' window was involved on which the transaction was confirmed. This latter approach increases the cost of the transaction, since it both increases the communication and the computational overhead while also making the user interface less friendly and more complicated.

There are two ways of assigning the pricing details to an intangible good like a piece of information. The first is to include the price within the traded object. The second is to provide a separate repository for this information. In the "shopping cart" model, where payments are separated from the traded goods, additional messages are needed (e.g., a "check-out" signal) to initiate the payment process. This is an effective approach for selling items that need the establishment of a secure session (e.g., SSL) in order for the payment to be completed. However, this model is not suitable for users who have specific items in mind that they would like to purchase, or where items are relatively cheap. Therefore, in the majority of the micropayment systems the former approach, where the price is included in the object itself, is the most cost-effective one.

Many of the existing micropayment systems employed on the Internet use their own proprietary method for creating a per-fee link and for encoding the information within this link. Today, a web-merchant willing to support multiple payment systems needs to embed, in each web page, payment information that is specific to each target system. For each micropayment system the pricing information is also encoded in a different way. Such encodings significantly increase the amount of information embedded in a web page. This situation motivates the need for a common markup supported by multiple payment systems. Michel [1999] specifies the encoding of the pricing information, in the case of per-fee links inside HTML pages, is specified. This specification does not, however, address security concerns related to the transmission of the linked web page from the merchant to the client, such as authentication of the parameters in the per-fee link (e.g., price) and confidentiality of the per-fee link. If such security concerns exist, the application should use other appropriate mechanisms (e.g., SSL). Unfortunately, this standard has never been formally completed. However, it is probably a good rule-of-thumb to follow the points specified in this standard when designing a micropayment system, since, as it is widely supported, this model will almost certainly be the starting point for any future standardizing processes, and conforming to a standard might be more effective in the long run than supporting only proprietary protocols.

Publishing does not only involve low value everyday goods (such as magazine articles) but also more specialized items like stock quotes. Anderson and Lee [1999] discuss the issue of secure publishing is discussed and a security model for publishing. They argue that insecure publishing may result in serious financial consequences or safety hazards: for example, a stock market investor tricked into relying on bogus news, or a doctor fooled into relying on falsified on-line reference data for drugs. Secure publishing involves integrity of content, persistent reference, revision history, evidence of authorship and copyright protection. Without doubt, the necessary support for secure publishing will further add to the overall cost of a micropayment system, and therefore the developer should use such security features only when they are absolutely necessary.

9. Case Study

In this section, we present an architectural model for a micropayment system, which is based on one of the most cost-effective systems available today. We evaluate the associated costs of this system and propose ways of further minimizing some of those costs, so as to reach the optimal point where the full cost of the system would reach what we consider to be the absolute minimum.

The base of our reference system is one heavily discussed recently in the Linux arena. This “Web Bug-based Micropayment Model” is a very simple scheme [Yeargin, 2003]. As we have mentioned in the previous sections the three parties of a transaction are:

- **The Merchants**, which in our case are the participating websites’ owners
- **The Broker**, which is a micropayment clearing house
- **The Users**, who are the participating web surfers

The roles of these three types of participants are as follows:

A web user purchases a user-account at the micropayment clearing house and funds it with real money. The web surfer can now log into the account from any computer and be issued a funded surfing cookie.

A website owner opens a website account with the clearing house. The website owner receives account information and instructions from the broker together with a template for web bugs. The webmaster updates the website with code to display the web bug on the priced documents. The information containing that website’s account number is embedded in the web-bug’s code. For example, each website could be given a different URL for the image file at the clearing house’s web bug server in order to identify the website associated with a certain request.

The web user tries to access the priced content at a participating website. At that point the user is prompted to authorize the micropayments to that website. The surfer authorizes the payments and surfs into the priced area of the website. There, each page contains a web bug, which causes the user to fetch the invisible image from the clearing house. The web bug causes a counter to be incremented in the web surfer’s account and another counter to be incremented in the website’s account. The web bug is also responsible for sending from the broker to the website a small confirmation message with an IP address and a referring URL to authenticate the web surfer as a micropayment member with surfing funds available.

Since the user has a cookie from the broker and the web bug code identifies the website, the clearing house has all the necessary information to move a cent—or ten—from the user’s account into the merchant’s account. The clearing house, of course, takes a piece of the micropayment—the house cut.

Let us now assume that this system is used for purchasing articles in a magazine that cost from \$0.05 to \$0.20 each (depending on their context). This application seems to be one of the most demanding for micropayments since the cost of the “goods” is extremely low. On the other hand, the same system can easily be employed in the very (in)famous operation of purchasing music over the Internet, but in this case the purchasing items cost a lot more (about \$1) and therefore it is less crucial to minimize the micropayment system’s cost. We also assume that the web-magazine has 100,000 active users each day and the clearing-house, which is a really famous one, has 10,000,000 customers, of which an average of 1,000,000 are “active” [Pew Internet & American Life Project, 2002]. Based on those assumptions, we will then analyze the various system cost-factors belonging to the above-mentioned categories.

Regarding the fixed technical costs on the side of the website operator, there is no need to keep accounts for the clients since the payment takes place in the clearing house, therefore the operator does not have to worry about who the client is. As a result, the account cost can be zero (the operator may want to keep some info about who the client is for marketing reasons). Moreover, the clearing-house has to maintain the account info, but since it needs just one account per user no matter how many websites a user may want to visit, the account overhead is the minimum possible. If we assume each account record is about 10KB, then the broker needs about 10GB of storage, which today costs just $10. In terms of future enhancements to the system, their costs are practically negligible since it is based on web bugs, which are small software programs that can very easily be altered so as to support future HTML or IP protocols, for example.

Moving to storage costs, the merchant can minimize its legal and financial costs, since it can ask the clearing house to provide it with just one record for each client, each day/week or month, where only the total money spent and the number of articles purchased will be listed. This record seems to be enough for any kind of financial auditing and can also give the magazine operator a way to ensure that the clearing house does not perform any kind of fraud (since the operator can have a very simple mechanism for measuring the total number of articles purchased and then compare this number with the one provided by the broker). For the magazine's 100,000 customers, we need, in the worst case, 100,000 records a day, and assuming that each record is 1KB, we need 100MB/day. Therefore, for a year we need 40GB, which costs today about $40, or about $120 including maintenance [as calculated in Gartner Group, 2000]. On the other hand, the clearing-house should keep every single transaction somewhere, in order to be able to defend itself when a transaction is challenged by either the customer or the merchant. If each user purchases an average of 4 "products" a day, there would be a need for 4GB of storage per day (given that each record is just 1KB). If the clearing house has a policy that it does not accept any challenges for its transactions 1 month after the monthly statement is received by either the user or the website operator, then it has to keep those records for at most 2 months. As a result, it needs at most 250GB of storage, that can be reused when the transactions it contains "expire". Therefore, the overall cost is in the order of $700. The only permanent storage needed, which consists of the simple 1K records mentioned above that are sent to the merchant and/or user and can be used for possible financial auditing, is in the order of 20GB (a $60 cost).

Regarding the very important technical storage costs, the merchant does not have to keep any kind of database, and/or initiate transactions, therefore no such cost factor is associated with it. The broker however does need a database for the customers' money and their transactions. If, again, each of the broker's active users perform 4 transactions a day, of which half are during the standard (for entertainment websites) 18:00-21:00 peak hours [Sparkes, 2003], then it would have to cope with about 2M transaction in those 3 hours or, in other words, about 200 transactions a second. According to section 3.2 above, if the broker uses a standard database system to handle those transactions, this will yield a cost of at least $0.10 per transaction, which is obviously prohibitively high! However, if the details of the customers are pre-loaded in on-line storage before those peak times, (and thus accessible at a much lower rate), and then the transactions details are "cached" before being written back to the database (after those peak times), the database system used can be a "standard" PC-based one, which according to Microsoft [2004] costs only $500K a year or $0.00125 per transaction.

With respect to the computational costs, the system authorizes a user each time it accesses a website. Therefore, the possible public key or other authorization mechanism would be employed in a minimal way. If each of the website operator's 100,000 clients purchases an average of just 4 "items", the web-server should be able to accommodate about 5 authorization transactions in a second during peak hours. Given that today's high performance PCs can execute up to 10 such transactions per second [Intel Corporation, 2001], it seems that the operator may need just one additional server for authorization. Given the maintenance costs of such a server [Sparkes, 2003], which can be up to $30,000 a year, each transaction's computational cost will be about $0.000001. Regarding the computational costs of the intermediate broker, these are dominated by the settlement phase in which the money is transferred from the customer's account to that of the seller. Given that we have implemented, for benchmarking reasons, a simple money transfer protocol with less than 10,000 x86 assembly instructions, a 3GIPS (Giga Instructions per Second) standard PC today can perform 50,000 such transfers in a second (including the I/O and memory overheads), or in other words two of those PCs (or maybe even one if it is pushed to its limits) can easily satisfy all the 2M transfers in the 3 peak hours. Therefore, the associated cost is in the order of $6,000 a year.

Moving to the very important communication costs, the baseline for our computations is that one Mega-Byte of transmitted data costs about 2 cents [JANET, 1998]. Given that the size of the web-bug is less than 1KB, the communication overhead of the web-operator, which has to transmit the bug to the consumer, is about $0.00002 per item. However this cost can grow if a new authorization page has to be acknowledged by the client for every article bought. Then, for each item the operator has to transmit a new page, which will probably cost it about $0.0004 per item, or about $15,000 a year. However, this additional feature has the advantage of overcoming any uncertainty about how much a surfer was spending. On the other hand, some users may find the pop-up windows frustrating. Moving to the clearing house's communication costs, we calculate the sizes of the various messages needed to complete a single transaction to be in the order of 2KB-4KB for each transaction. Therefore, we claim that the communication costs for the broker is in the order of $0.0004-$0.0008 per transaction or $600,000 - $1,200,000 a year. It is clear that the size of the transaction-messages, which can be easily undervalued by a micropayment designer, can be the factor that differentiates a successful from an unsuccessful system.

In terms of the publishing costs, the reference system is claimed to be optimal, since the only overhead incurred is the addition of a couple of lines of code at the first HTML page of each item, which will cost almost nothing for the operator. However, there are slight problems with the security aspects of the web-bug based mechanism, such as the authentication of the parameters in the link (e.g., price), or its confidentiality. As was described in Section 7 above, in order to support secure publishing, you have to add various new dimensions to the micropayment system. However, in this case study, we assume that security would not be required for the magazine articles being sold by our hypothetical website operator, and therefore we believe do not include any additional security costs in our measurements. Regarding the broker, the proposed system has the additional advantage that the payment intermediary does not have to manage any of the publishing aspects of the system.

The last large category of cost factors is the administrative costs. As described in Section 6 above, this includes the costs implied by the financial environment, such as billing, commissions and interoperability. Regarding the billing issues, the merchant is not directly implied in any kind of billing to the user, as the broker is responsible for both billing the user and the merchant. Since, as was illustrated above, the billing is done only once a month and, according to MailArmony Corporation [2002], each email costs about $0.005 if it is created automatically, the overall billing cost for the broker would be about $6,000 a month or $70,000 a year. If the bills needed any kind of customization requiring human intervention, the cost per email would increase to $1 [MailArmony Corporation, 2002], and therefore the overall monthly cost would be $1,100,00, which would probably make the broker unprofitable! Regarding the interoperability issue, it is hard to estimate the cost for both the broker and the merchant. If we assume that the additional system the merchant uses is similar to the reference one, then the only cost that is added is that of another 0.5KB of data at the end of a page, since all the described mechanisms can work for a new system as well. Similarly, the broker can support similar web-bug based instruments at almost no additional cost. If, on the other hand, either the merchant or the broker support micropayment mechanisms of a different philosophy, the cost of interoperability could be very high and cannot easily be estimated. If we look at the patent costs of the scheme, one of the main advantages of it is that there is no risk of "hidden" associated patents, since the whole system depends on freely available methods and tools. Regarding certification costs, the website need not necessarily be certified, using a certification authority, since the payment transaction takes place between the broker and the client. However, in order for the customer to feel safer, it is advisable to have the website certified. This certification cost is now about $250 a year [Innova.NET, 2001-2002]. The broker should certainly be certified, and at the highest level, and therefore a certification cost of about $2,000 a year should be added to the yearly maintenance cost of the system.

Regarding the exception handling issues, since digital goods cannot be returned and there is no physical delivery involved, the exceptions that can arise are very limited, and they have mainly to do with the fact that the protocol can be interrupted. This can happen, for example, if the magazine web-server breaks down after the user has clicked on the article and has been charged, or when the broker's system breaks down before the actual money transfer has taken place and after the client has downloaded the article. In the first case, the system will over-charge the customer, while in the second it will under-charge him or her. Therefore, the broker should have a mechanism for the customers' complains, staffed by the appropriate personnel which, in every case, would check the log-files of the operator's web-server and the broker's system and identify whether the claims are valid or not. However, since the value of the goods is so low, the complaints will be minimal (there are no many customers that will fill-in a form so as to have $0.10 returned to their account). Therefore the "complaints department" of the broker, even if there are 10,000,000 customers (of which just 1,000,000 are active) could be staffed by as few as 20 entry-level employees (or even part-time teleworkers). Therefore, we believe that the total associated cost will be in the order of at most $1,000,000 per year, and this includes also the cost for any other complaints that can arise from customers (such as problems with the cookies being lost, their new browser not working properly, etc). Another aspect of exception costs arises when fraud is taken into account. In the reference system, the broker is responsible for taking the actual money from the client, and for giving it to the merchant. Users would normally be allowed to deposit money in their accounts using credit cards. If the broker wanted to minimize fraud costs and did not accept credit cards, and instead only accepted wire transfers or checks, etc., these kinds of payments would probably need greater human intervention and therefore, 10-20 more employees may be needed at a cost of $1,000,000.

Moving to credit/commission costs, if the broker is the bank itself (which is probably the best option), then the credit and commission costs will be minimal (or nothing). If the broker is an independent authority, and credit cards are used, then credit/commission costs would wipe out about 2-3% of their transaction revenue, which means that the broker's commission to the merchant should certainly be significantly higher than this percentage. Given the 1,000,000 active users that are doing 4 purchases a day at a mean cost of $0.1 each, the total amount of money transferred to the broker, in a year, will be about $150,000,000, and thus the credit cost will be in the order of $3,000,000-$4,500,000 a year. Since, in case the broker is not the bank itself, the cost of the employees that will be responsible for the money transfers is probably much lower than the commission cost, and moreover, money transfers are safer as described above, the broker should certainly go for the money-transfer option. On the other hand, if a bank is acting as the broker, the commission costs will probably be negligible. But even in this case, the bank may prefer the wire-transfer option, since, as it was described above, the costs due to fraudulent credit card transactions are likely to be much higher than the labor cost triggered by the money-transfer option.

Finally, the availability costs will depend on the quality of service both the merchant and the broker want to offer. Since the broker, especially, will need its systems to be available 24 hours a day, 7 days week—otherwise it will lose its credibility—it will probably have to pay the required cost to purchase the necessary hardware and software to offer availability as close as possible to 100%. This means that the above infrastructure costs would have to be increased by 25%-50% [Sparkes, 2003]. Since the sum of all the infrastructure costs described in the above paragraphs is in the order of $500,000, then we can safely assume that the additional availability costs will be $125,000-$250,000 depending on the actual availability rate demanded. Similarly, if the merchant wants its part of the micropayment process to be available almost 100% of the time, it has to add a comparable overhead, or in other words, up to $15,000 a year. (Note that this overhead has nothing to do with the infrastructure employed by the merchant so as to provide the actual content; we look only at the payment mechanism's costs.)

In Table 1 all the cost factors of the proposed system are listed. As can be clearly seen, by slightly altering some of the parameters, even of a well-analyzed system such as the reference one we used in our example, the overall cost can be reduced by almost a factor of 10! Given that the annual turnover for the broker will be about $150,000,000 and the micropayment mechanism's costs can be as low as $3,200,000, (or in other words about 2% of the turnover), the broker can probably make a profit if its micropayment commission is in the order of 3%-4%. Obviously, if it makes the wrong design choices, it may have to charge as much as 20% for commission in order to be profitable. Similarly, the merchant would have a turnover of about $3,000,000 and, given a commission of even 4%, the costs associated with the micropayment's mechanism would be in the order of only 5%-6% of its turnover, depending on the mechanism's design parameters.

Note that the fact that some of the costs are negligible in this system does not necessarily mean that they are negligible in every micropayment system. For each one of the factors mentioned, we can probably find at least one real system that has a non-negligible associated cost.

Table 1: Reference Model's annual costs based on 1.5 Billion transactions per year

	Merchant's costs (in $)		Broker's costs (in $)	
	Lower-Limit	Higher-Limit	Lower-Limit	Higher-Limit
Fixed Technical	0	0	300	300
Storage (Legal & Financial)	60	60	760	760
Storage (Technical)	0	0	500 K	10 M
Computational	30K	30K	0	3 K
Communication	7.5K	15K	600 K	1.2 M
Administrative Commissions	Depend on broker	Depend on broker	0	4.5 M
Administrative-Billing	0	0	70K	14 M
Administrative-Patents	0	0	0	0
Administrative-Certification	250	250	2.5 K	2.5 K
Administrative-Exception Handling	0	0	1 M	1 M
Administrative-Credit Cost	0	0	1 M	1 M
Availability	0	15K	125K	250K
Total	**38 K + commissions**	**61 K + commissions**	**3.2 M**	**27 M**

In general, since in micropayment mechanisms even a difference of much less that a tenth of a cent per transaction can distinguish a successful from an unsuccessful system, and in this case-study we have shown that the wrong design choices can increase the transaction cost by a factor of 9, we argue that it is extremely important for the micropayment designer to thoroughly analyze each of the cost factors listed in this paper to make sure that the optimal point is found where the overall micropayment overhead expense is indeed minimized.

10. Conclusions

Micropayments have recently resurfaced as a payment option due to the growth of low-cost mobile network services and other new applications such as distributed storage systems. It is well known that a micropayment mechanism needs to keep the cost of individual transactions low. Therefore, in order to design an efficient micropayment mechanism it is very important for the designer to consider and minimize each of its cost factors. In this paper we identified the most important of the technical cost factors and presented ways to minimize their associated costs. This paper also demonstrated the cost efficiency of various micropayment protocols and the consequences of their design decisions. Additionally, it proposed ways of reducing some of the costs of a number of existing micropayment protocols.

It is essential for all the micropayment system designers to be able to identify all these technical cost parameters. We argue that existing payment schemes have several features that are potentially unnecessary when their benefits are traded-off against their costs, since these features may often be more appropriately delivered by external systems that provide equivalent or superior functionality at a much lower cost.

Acknowledgements
We thank the reviewers, and especially the corresponding editor, for their useful comments in restructuring this article.

References

Aberer, K., and A. Wombacher, "A Language for Information Commerce Processes," In *Third International Workshop on Advanced Issues of E-Commerce and Web-Based Information Systems*, San Jose, California, USA, June 21-22, (available online: http://lsirpeople.epfl.ch/aberer/GMD-PAPERS/WECWIS2001.pdf, last accessed May 10, 2004), 2001.

Anderson, R. J., and J.-H. Lee, "Jikzi: A New Framework for Secure Publishing", *Seventh Cambridge Workshop on Security Protocols*, Cambridge, UK, April 6-8, (available online: http://www.ftp.cl.cam.ac.uk/ftp/users/rja14/jikzi.pdf, last accessed May 10, 2004), 1999.

Bellare, M., and P. Rogaway, "Collision-Resistant Hashing: Towards Making UOWHFs Practical", *17th Annual International Cryptology Conference*, Santa Barbara, California, USA, August 17-21, (available online: http://www-cse.ucsd.edu/users/mihir/papers/tcr-hash.ps.gz, last accessed May 10, 2004), 1997.

Cardis, "Home Page of Cardis-International," (available online: http://www.cardis-international.com, last accessed May 10, 2004), 2004.

Cybercash, "Home Page of Cybercash® Internet Payment Services," acquired by VeriSign, (available online: http://www.cybercash.com, last accessed May 10, 2004), 2004.

Daswani, N., D. Boneh, H. Garcia-Molina, S. Ketchpel, and A. Paepcke, "SWAPEROO: A Simple Wallet Architecture for Payments, Exchanges, Refunds, and OtherOperations", *Proceedings of the Third USENIX Workshop on Electronic Commerce* (pp. 121-129), Boston, Massachusetts, USA, August 31-September 3, 1998.

Gabber, E., and A. Silberschatz, "Agora: A Minimal Distributed Protocol for Electronic Commerce", *Proceedings of the Second USENIX Workshop on Electronic Commerce* (pp. 223-232), Oakland, California, USA, November 18-21, 1996.

Gartner Group, *Total Cost of Storage Ownership: A User-oriented Approach*, Public Report, Gartner Group IT Management Consulting, February 16, 2000.

Glassman, S., M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro, "The MilliCent Protocol for Inexpensive Electronic Commerce", *Fourth International World Wide Web Conference*, Darmstadt, Germany, December 10-13, (available online: http://www.w3.org/Conferences/WWW4/Papers/246/, last accessed May 10, 2004), 1995.

Goh, A., K. W. Yip, and D. C. L. Ngo, "Flexibly-Configurable and Computation-Efficient Digital Cash with Polynomial-Thresholded Coinage," in A. Lioy & D. Mazzocchi (Eds.), *Communications and Multimedia Security—Advanced Techniques for Network and Data Protection*, Lecture Notes in Computer Science (Vol. 2828, pp. 181-193), Proceedings of the 7th IFIP TC-6 TC-11 International Conference, CMS 2003, Torino, Italy, October 2-3, 2003.

Herzberg, A., *Introduction to Applied Cryptography for Secure Communication and Commerce*, Prentice-Hall, in preparation, 2004.

Herzberg, A., and H. Yochai, "Mini-Pay: Charging per Click on the Web," *Sixth International WWW Conference*, Santa Clara, California, USA, April 7-11, (available online: http://decweb.ethz.ch/WWW6/Technical/Paper099/Paper99.html, last accessed May 10, 2004), 1997.

Hurwicz, M., "The Return of Micropayments: Will Tiny Payments Finally Make Their Big Debut?" *New Architect Magazine*, (the only reference to some of the dead micropayment systems, available online: http://www.webtechniques.com/archives/2001/12/hurwicz/, last accessed May 10, 2004), December, 2001.

Innova.NET, "Innova - Web Hosting: eCommerce", online document (available at: http://www.innova.net/products/ecommerce.htm, last accessed May 10, 2004), 2001-2002.

Intel Corporation, "The Itanium® Processor Family - High Performance on Security Algorithms", white paper::19909 (available online at: http://www.intel.com/cd/ids/developer/asmo-na/eng/microprocessors/itanium/optimization/19909.htm, last accessed May 10, 2004), March, 2001.

Internet Dollar, home page (was available at: http://www.internetdollar.com, last accessed January 2004, listed at: http://www.onlinepaymentsystems.com/list.html, last accessed May 10, 2004).

JANET Organization, "JANET Network Charges", news circular (available at http://www.jisc.ac.uk/index.cfm?name=news_circular_3_98, last accessed May 10, 2004), March 1, 1998.

Jarecki, S., and A. Odlyzko, "An Efficient Micropayment System Based on Probabilistic Polling", in R. Hirschfeld (Ed.), *Financial Crypography: First International Conference (FC '97)*, Anguilla, British West Indies, Feb. 24-28, Lecture Notes in Computer Science (Vol. 1318, pp. 173-192), 1997.

Kytöjoki, J., and V. Kärpijoki, "Micropayments - Requirements and Solutions", in A. Karila (Ed.), *Proceedings of the Seminar on Network Security: Security in Electronic Transactions*, Helsinki University of Technology, Fall (available at: http://www.tml.hut.fi/Opinnot/Tik-110.501/1999/papers/micropayments/micropayments.html, last accessed May 10, 2004), 1999.

Law, L., S. Sabett, and J. Solinas, "How to Make a Mint: The Cryptography of Anonymous Electronic Cash", white paper, National Security Agency, Office of Information Security and Technology, Cryptology Division, June 18 (available at: http://jya.com/nsamint.htm, last accessed May 10, 2004), 1996.

Lipton, R. J., and R. Ostrovsky, "Micro-Payments via Efficient Coin-Flipping", in *Proceedings of the Second Financial Cryptography Conference (FINANCIAL CRYPTO-98)*, Anguilla, British West Indies, Feb. 23-27, Lecture Notes in Computer Science (Vol. 1465, pp. 1-15), (available at: http://www.argreenhouse.com/papers/rafail/33.pdf, last accessed May 10, 2004), 1998.

MailArmony Corporation, "MailArmory Secures Innovation Award", press release, Fort Collins, CO, July 26 (available at: http://www.mailarmory.com/resources/news/07262002.html, last accessed May 10, 2004), 2002

Manasse, M. S., "The MilliCent Protocols for Electronic Commerce", in D. E. Geer (Ed.), *Proceedings of the First USENIX Workshop on Electronic Commerce (EC 95)*, New York, NY, July 11-12 (available at: http://www.usenix.org/publications/library/proceedings/ec95/manasse.html , last accessed May 10, 2004), 1995.

Markose, S. M., and Y. J. Loke, "Network Effects on Cash-Card Substitution in Transactions and Low Interest Rate Regimes", *Economic Journal*, Vol. 113, Issue 487: 456-476, April 2003.

Masciarotte, O., "Can You Spare a % of a Dime?", *Mix Professional Audio and Music Production*, May 1 (available at: http://mixonline.com/ar/audio_spare_dime/, last accessed May 10, 2004), 2003.

Micali, S., and R. L. Rivest, "Micropayments Revisited", in B. Preneel (Ed.), *Progress in Cryptology—CT-RSA 2002*, San Jose, CA, February 18-22 (available at: http://theory.lcs.mit.edu/~rivest/MicaliRivest-MicropaymentsRevisited.pdf, last accessed May 10, 2004), Lecture Notes in Computer Science (Vol. 2271, pp. 149-163), 2003.

Michel T. (Ed.), "Common Markup for Micropayment per-fee-links", W3C Working Draft, online document (available at: http://www.w3.org/TR/Micropayment-Markup/, last accessed May 10, 2004), August 25, 1999.

Microsoft Corporation, "Access Home: How to Buy", online, http://www.microsoft.com/Office/Access/howtobuy/default.mspx, last accessed May 10, 2004), October 20, 2003.

Mondex®, "Mondex", now part of the global OneSMART™ MasterCard® program, online document (available at: http://www.mondex.com, last accessed May 10, 2004), 2004.

Murray, M., and R. Narayanaswamy, "Some Free - Some Fee: the Emerging Business Model for e-Content Web Sites", *Journal of Internet Banking and Commerce*, vol. 8, no. 2, online (available at: http://www.arraydev.com/commerce/JIBC/0311-04.htm, last accessed May 10, 2004), November, 2003.

Odlyzko, A. M., The Case Against Micropayments", in R. N. Wright (Ed.), *Financial Cryptography: Seventh International Conference (FC 2003*), Guadeloupe, French West Indies, January 27-30, Lecture Notes in Computer Science (Vol. 2742, pp. 77-83), 2003.

Pew Internet & American Life Project, "The Rise of the E-Citizen: How People Use Government Agencies' Web Sites", online report (available at: http://www.pewinternet.org/reports/toc.asp?Report=57, last accessed May 10, 2004), April 3, 2002.

Rivest R., and A. Shamir, "PayWord and MicroMint: Two Simple Micropayment Schemes", in M. A. Lomas (Ed.) *Proceedings of the 1996 International Workshop on Security Protocols*, Cambridge, UK, April 10-12 (available at: http://theory.lcs.mit.edu/~rivest/RivestShamir-mpay.pdf, last accessed May 10, 2004), Lecture Notes in Computer Science (Vol. 1189, pp. 69-87), 1996.

Rivest, R., "Electronic Lottery Tickets as Micropayments", in R. Hirschfeld (Ed.), *Financial Crypography: First International Conference (FC '97)*, Anguilla, British West Indies, Feb. 24-28, Lecture Notes in Computer Science (Vol. 1318, pp. 307-314), 1997.

Roscoe T., and S. Hand, "Transaction-Based Charging in Mnemosyne: A Peer-to-Peer Steganographic Storage System", in *Proceedings of the International Workshop on Peer-to-Peer Computing at Networking 2002*, Pisa, Italy, May 10-12, 2002.

Sparkes, H., "The True Cost of Commodity Servers", online report, Unisys, UK (available at: http://www.cfoeurope.com/media/pdf/unisys_consolidation.pdf, last accessed May 10, 2004), 2003.

Stallings, W., *Cryptography and Network Security: Principles and Practice*, 3rd ed., Prentice Hall, Upper Saddle River, 2003.

Ulema, M., and B. Kozbe, "Management of Next-Generation Wireless Networks and Services", *IEEE Communications Magazine*, vol. 41, no. 2: 86- 87, 2003.

Thompson, K., G. J. Miller, and R. Wilder, "Wide-Area Internet Traffic Patterns and Characteristics", *IEEE Network*, vol. 11, no. 6:10-23, 1997.

Van Someren, N., A. Odlyzko, R. Rivest, T. Jones, and D. Goldie-Scot, "Does Anyone Really Need Micropayments?", in R. N. Wright (Ed.), *Financial Cryptography: Seventh International Conference (FC 2003*), Guadeloupe, French West Indies, January 27-30, Lecture Notes in Computer Science (Vol. 2742, pp. 69-76), 2003.

Wayner, P., "The Torrent of Patents for E-Commerce Schemes Raises New Questions About an Old-fashioned System", *Salon Magazine*, online (available at: http://www.salon.com/21st/feature/1999/03/09feature.html, last accessed May 10, 2004), March, 1999.

Wheeler, D., "Transactions Using Bets", *Proceedings of the 1996 International Workshop on Security Protocols*, Cambridge, UK, April 10-12, Lecture Notes in Computer Science (Vol. 1189, pp. 89-92), 1996.

Yacobi, Y., "On the Continuum Between On-line and Off-line E-cash Systems", in R. Hirschfeld (Ed.), *Financial Cryptography: First International Conference (FC '97)*, Anguilla, British West Indies, February 24-28, Lecture Notes in Computer Science (Vol. 1318, pp. 193-202), 1997.

Yeargin, R., "A Web Bug-based Micropayment Model", *Librenex*, online document (available at: http://librenix.com/?inode=2706, last accessed May 10, 2004), January 13, 2003.