A Lightweight Buyer's Trust Model for Micropayment Systems

Samad Kardan and Mehdi Shajari
Department Computer Engineering and Information Technology Department
Amirkabir University of Technology (Tehran Polytechnic)
No. 424 Hafez St., Tehran, Iran, P.O. Box: 15875-4413.
IRAN
{skardan, mshajari}@ aut.ac.ir

Abstract: In this paper we present a trust model for an enhanced version of MR2 micropayment scheme. We named this scheme TMR2. TMR2's light-weight trust model is based on user polling, sales volume and vendor's reputation. We claim that TMR2 can solve the mind barrier problem, which will result in the expansion of micropayment usage. The new token proposed to handle trust is a certificate called Rating certificate. We propose a new Rating certificate for merchants to support users' trust. Our proposed purchase process needs validation of this certificate, therefore, compared to MR2, the proposed scheme needs only one extra on-line digital signature validation. This validation adds less than 4 percent computational overhead in the purchase process, which is acceptable, considering the benefits that users will gain.

Key-Words: User Trust Model, MR2 Micropayment Scheme, Micropayment Schemes, Computational Trust.

1 Introduction

After huge expansion in the population of web users, new subjects such as electronic commerce and consequently micropayment have emerged. The micropayment is a worthwhile idea, which has not become a profitable one yet.

The difference between a conventional payment system and a micropayment system is in the processing cost of each transaction. To form a micropayment system one has to minimize the fixed part of transaction cost, and if possible to cut it completely. For example a 30 cent fixed fee per transaction, which is reasonable for a conventional payment, is 60 percent of a 50 cent micropayment which is unacceptable.

In micropayment, one has to maximize the number of transactions processed by a server to divide the fixed cost of the facility among multiple transactions. This way, a fee of 10-20 percent of each transaction would become profitable; assuming low computational and communicational overhead.

Theoretically, many micropayment schemes have been proposed. These schemes achieved lower computational and communicational overhead, in comparison to conventional payment systems by degrading security.

The continuous increase in the processing power and decrease in the processing cost and emergence of new ideas in security area (such as online-offline signatures) make new micropayment schemes more practical. Universal aggregation of micropayments, using micro electronic checks is an interesting idea, proposed by Micali and Rivest [1]. MR2 is a practical scheme, which tries to solve the problems, which arise in practical implementations of former schemes [2]. For the above mentioned reasons we selected MR2 as the base scheme for our work.

Many attempts have been made to achieve a micropayment system with the above mentioned specifications, but most of them were unsuccessful. Various reasons have been proposed in different sources [3], [4], [5], [6], [7] for the failure of micropayment systems. One of the main reasons is the customers' mental cost for deciding to pay for something from unfamiliar merchant even if it is a very cheap item. The main approach to solve this problem is to define a trust structure that determines quality of a good or service provided by a vendor based on users' past experiences with that vendor. To do this we designed and tested a trust structure for our micropayment approach with minimum possible computational and communicational overhead.

The major contribution of this paper is the design of a lightweight trust model to create the sense of trust in the users of our proposed micropayment system. There is also an important contribution to enhance MR2 micropayment scheme that enhance the security and implementability of this scheme.

The remainder of this paper is organized as follows. Section 2 is a brief review of the micropay-

ment schemes and specially MR2 it also explains our enhancements to MR2. In section 3 we will discuss the reasons for failures of micropayment implementations. In Section 4 we will introduce the trust and trust management issues, and then we will explain our trust model. Section 5 is about the proposed scheme and its sessions. The section 6 discusses the implementation of TMR2 and explains the results. Finally in section 7 we discuss the conclusions and give a summary of the paper.

2 Micropayment Systems

As stated before, conventional payment methods are not applicable for handling high-volume low-value transactions of micropayments. There have been various methods with low computational and communicational overhead proposed for micropayments.

2.1 Theoretical Models

The theoretical methods proposed for micropayments can be classified by different factors. Here we explain some of them.

Applied Technique: There are different cryptographic and none cryptographic technologies used in micropayment systems. We classify these various schemes into 6 groups as follows:

- Using the ID certificate and a database for minted coins to pay to a specific vendor.
  Such as: Millicent [8].
- Using an Account ID and balance database (phone card model).
  Such as: Subscrip [9] and BitPass [10].
- Using one-way functions
  - Using one-way functions to produce payment chains, payment trees and similar structures. Such as: PayWord [11], μ-iKP [9], PayTree [9], UBOT [9], General Pay-Word [12].
  - Using n-way collisions in hash functions to generate coins verifiable at vendors.
    Such as: MicroMint [11].
- Using conventional payment method to handle a group of micropayments in a centralized account system (payment aggregation by broker). Such as: Jalda [9], NewGenPay Micropayment [9].
- Using Probability theory to aggregate a group of micropayments to a conventional payment, with an equal probability among payers.
- Such as: Coin Flipping [13], LotteryTickets [14], MR Schemes [1] and Peppercoin [15].
- Peer-to-peer micropayments using transferable coins and distributed coin verification.
- Such as: KARMA [16], off-line KARMA [17], PPay [18] and CPay [19].

Connection: It explains whether the parties must be on-line or transaction handling can be done in an off-line mode. The off-line mode is preferred over on-line mode [20].

Fraud: It explains how a system encounters double spending or over spending. One possible action is to prevent and the other is to detect later. The former is safer but latter needs less computational and communicational cost. The later also enables off-line mode. In case of detection the offender is fined, or expelled from the system [21].

Broker: It explains whether there is a broker in the system or not and how is the broker involved in the transactions. There are distributed or centralized and active or aggregator brokers.

Token-based or Account-based: It explains the structure used for micropayments. In account-based systems, an account is assigned to each user and after authentication user can order a payment, and payment means an update in the payer's and payee's account balances. In this method, bank, broker, or the vendor himself is responsible for handling the transactions. The advantage of this method is that usually it does not need a client-side component, so payment is possible from everywhere.

In the token-based method, the asset of user is represented as sets of bits called tokens. User pays vendor by sending him these tokens. In this method a specific software id required for management and computation of these tokens. The tokens are stored on user's computers or dedicated hardware; therefore this method is less accessible compared to account-based method. On the other hand the computations are done partially in client-side which enables the servers to handle more transactions. The authors of [21] suggest that the account-based method is more common in newer micropayment systems, but comparing the two major implementations BitPass [10] and Peppercoin [15] and their outcome, it can be concluded that token-based approach is more compatible with micropayment requirements.

Aggregation Level: Aggregation of a certain number of micropayments to a conventional payment is done in the aggregation level. At the broker or vendor level, a broker or a vendor aggregates the micropayments made by a specific user or to a specific vendor accordingly and requests one macro (conventional) payment for them.

There is a higher level of aggregation introduced

in [1] called universal aggregation. At this level all actual payments are macro payments. A winning probability is assigned to each micropayment according to its value. A cryptographic selection function selects the micropayment that is upgraded to a macro payment. Statistically the value of the macro payment is equal to sum of the number of micropayments which caused one winner. We will discuss this type of micropayments in more detail in section 2.3.

This level of aggregation, reduces the cost of making a payment for customers, the processing overhead for banks, and also the fee paid by vendor for acquiring the micropayments.

2.2 Major Implemented Micropayment Systems

- BitPass: It was a rather successful micropayment system which had operated for 4 years. It is based on the phone card model. The advantage of this system was that the user did not need to login in the BitPass site to order a payment, instead the authentication was done in the seller's site. It was user-friendly and the installation process for sellers was relatively easy. This system had cash turnover of 13 Million Dollar, but it suspended its operations on 2007.
- PepperCoin: It is a system based on MR2 micropayment scheme. It was founded by Micali and Rivest. The main idea of this system is the universal aggregation as explained in the former section. The prevalent payments are done by a 10 cent coin. Each 100 payments are aggregated to a 10$ conventional payment.

  In this system the overhead of transferring cash to conventional financial systems is reduced, this enables the system to have a greater profit margin. The aggregation also is beneficial for the sellers; because they can have smaller transactions with customers (e.g. compare the 10 cent payment in PepperCoin with 25 cent payment in BitPass).

  This system had over 16 Million Dollar cash turnover in 2007. It is offering micropayment cards and the gift cards of the shops. In 2007 PepperCoin was acquired by Chalkstone Company, which is active in this sector. PepperCoin is currently the most successful Implemented micropayment system.

2.3 MR2 Micropayment Scheme

First, we introduce the MR2 scheme as described in [1], and then we will discuss our enhancement to the original MR2. In MR2 scheme there are three parties: Bank (B) which provides micropayment service, Merchant (M) and user (U). During the initialization process, each user and merchant is given a public-key. A specific digital signature method is also assigned to each merchant during this phase.

The payments are done by electronic checks (micro checks). To pay for electronic merchandise, user signs a micro check and sends it to the merchant. The checks have a serial number started with 1. This serial number is increased with each drawn check. The merchant verifies the user's signature and hashes the check to produce the digest, and then he signs the digest (SD_M).

The signed digest is an input to the selection function (F). The function F returns a decimal value between 0 and 1, with a selection probability of *s*. If F (SDM) < *s* then, this check is selected to be upgraded to a macro payment, and the check is saved. To acquire a selected check, merchant must send the check with its SDM to the bank. Bank verifies the check and SDM, and then, he credits the merchant's account with the amount of *1/s*. If the check's serial number, i.e. SN, is greater than highest serial number paid by this user, i.e. $MaxSN_U$, then bank charges user's account with the amount of $SN-MaxSN_U$ and sets the $MaxSN_U$ to SN.

Bank compares the date and serial number of the last check and compares it to the current check to detect user's possible reuse of the same serial number. Generally, any possible misuse of system is detectable by computing the payment rate, i.e. PR, which is the quotient of total amount paid by bank for that user, i.e. TP_U, divided by $MaxSN_U$. This rate must be close to *1/s*.

The user whose PR is much greater than *1/s* (e.g. *2/s*) is considered a malicious user and is fined or expelled from the system. The merchant who sends too much payable checks from malicious users is also fined or his contract is reconsidered. These measures and other statistical analysis, such as monthly or yearly usage patterns, enable bank to prevent great losses. Minor losses are tolerable because of the fines and membership fees paid by defrauding users and merchants.

2.4 Enhancements

During the implementation of the original MR2 scheme, to integrate our trust model, we faced some challenges that were not solvable solely build on the original specifications of MR2 [1]. Therefore, we made some modifications to the original scheme to satisfy our needs. These modifications are as follows:

- **Check Book**

Bank generates a hash chain for each user at the registration time using the same mechanism as PayWord [7], and sends the last token of the chain to the user in his certificate. Each month the user gets a new token of the chain. Each month the merchant agent requests the token from the user agent and hashes this token. If the result is the same as the last month's token, then the user certificate is valid for another month (or any other time period set). This token which is similar to a check book in the real world helps the bank to better controlling of the users and detecting of frauds.

We faced several challenges during our implementation, for example each user must spend adequate number of checks (about 1/2s) in a month (or any other period of time) so that with a good probability (0.5), one of his checks become payable and bank can charge him for his checks. Practically, this may not happen every month, so money paid by the bank is much more than money collected from the users. To solve this problem, each user agent sends its last serial number when it gets the check book token each month, so the bank can update the user account. Although malicious users may send an old serial number but majority of honest users, this will balance between amounts paid to merchants and amounts collected from user.

- **Variable value for checks with fixed payment**

In MR2 amount paid for each payable check is *1/s* with a fixed value for all checks and if the checks get variable values the penalty system most be changed to prevent false detection of malicious users. Here we used another approach in favor of the bank, we used a new F function that gets a base probability (s) and a multiplier (check value) and select with a chance of their product, but the payment is always *1/s* this way the higher the check value is, user himself pays more share of the payment made to merchant. To explain it better consider a 10 cent check in a micropayment system with 1 cent as default payment and basic payment of 10$ (s=1/1000). This check will be a 10$ payment with probability of 1/100 (0.01= 10*1/1000). This way we do not need to change the penalty system, besides merchants prefer our mechanism for payment because they are get just one check instead of many checks payments more than the default value of system (e.g. 10 checks with 1 cent value for above sample).

- **Keeping track of last serial number of each user by merchant**

In MR2, since only payable checks are saved a malicious user may send a check with same serial number to a merchant if he somehow finds out that the first check was not selected for payment, this way after detection of that user this merchant may also seem malicious to bank, to prevent that merchant saves the last serial number each user sent to him and reports the malicious users to bank.

3 Success and failure of implemented micropayment systems

There has been two generations of micropayment systems, implemented for business purposes and failed to continue progress. Many reasons in different sources have been proposed for micropayment failures [3], [4], [5], [6], and [7]. The more important and common ones are as follows.

- Complexity of use
- Poor implementations
- Inappropriate business models
- High prepayment for initializing an account
- Complex and time-consuming initialization process.
- Requirement of costly hardware and software for merchants.
- Lack of anonymity for users.
- Promotion of free-content sites which use advertisement as their income source.
- High cost overhead of macro payments for micropayment providers.
- Mental cost of making decision to pay each low value payment for user.

Many implementations tried to address some of these problems. For example in MR2, the way implemented in this research, except for lack of anonymity and mental cost for users, other problems are mostly solved. But none of the proposed micropayment schemes give any attention to the mental cost of making decision to buy from an unknown merchant even in a low value purchase from user's point of view. Only Foley and Quillinan in [22] proposed adding a trust management to a hash chain based micropayment to let parties define trust rules to other parties. This trust rules determines how often an imbursement must be done for payment tokens earned from each user (integration of micropayments) and it is completely manual. Our proposed trust model is completely different as it calculates and provides trust from customer's point of view based on his satisfaction of buying a product from a merchant and it is partially based on other users polling besides trust calculation and evaluation is

automated in our scheme.

4 Trust and Trust management

Trust is a complicated issue in virtual community, besides the mechanism of trust in humans is not clear, and even it can not be considered fully rational. Many computational models for trust, has been proposed in different areas of application (See [27], [28] and [29] for some examples). In [23] Sabater and Sierra present an in-depth review of computational trust and reputation models.

Trust management is a rather young subject but it has been applied to many different areas of application. In [24] you can find a basic survey of trust management. For a review of trust management in access control and its applications see [25]. As stated before in [22] Foley and Quillinan proposed using trust management for integration of micropayments made by a user based on the trust on that user.

4.1 Proposed trust model

Our approach is to present trust from customers point of view, so considering that the micropayment systems must have minimum overhead, the proposed trust model is mostly based on the customer's experience and rating. There is only one external parameter included in the trust formula, which is calculated by the bank (micropayment provider) and represented as a rating certificate for specific merchandise sold by a specific merchant. This rating is based on the rating of the other users who bought this merchandise from the merchant, there is a minimum volume size for the polling, and we will explain it later.

The main usage of the trust model used here is to decide if the merchandise is worth buying. The customer agent calculates the trust according to formula (1).

$$trust(x) = \text{max\_cost\_r ate}[cat(x)] * \alpha + \text{max\_trans\_ rate}[m(x)] * \beta + cert\_rate(x) * \cos t(x) * \chi$$

$$(0 < \alpha, \beta, \delta < 1), (\alpha + \beta + \chi = 1) \quad (1)$$

Here *trust(x)* shows the trust to buy *x* from the merchant *m(x)* with cost of *cost(x)* and is a value in the system currency unit (e.g. US Dollar). The category which this merchandise belongs to is *cat(x)*, max_cost_rate[c] is an array in currency units which stores the maximum cost*rate for the *c* category paid up to now (somehow, user's satisfaction of paying for this kind of goods). max_trans_rate[m] is an array which stores the maximum cost*rate paid to buy something from the merchant *m* (somehow, the user's satisfaction of buying some thing from this merchant), its values are also in currency units, and *cert-rate(x)* returns the rating value in the rating certificate. α, β, γ are weights to customize the three parts of formula (category-based, merchant-based and global rating), and are set by user.

For each purchase after rating by user, the max_cost_rate[c] and max_trans_rate[m] are updated to reflect this rating. The algorithm that updates the max_cost_rate[c] is shown in algorithm (2). Here the trust value is the amount that user agent can buy without asking user about it, so the update strategy is to increase cautiously but decrease fast. This way whenever user is not satisfied with the purchased merchandise (rate< ACCEPT_RATE) system lowers the trust for automatic buy, so that user agent will not buy a poor quality merchandise automatically. The max_trans_rate[m] is updated same way. The ACCEPT_RATE and RECOMMEND_RATE are constants set by user.

```
Update_ max_cost_rate(cat as Category, cost as Cost, rate as Rate)
 If rate<ACCEPT_RATE then
  If max_cost_rate[cat]>=cost then max_cost_rate[cat] = cost*rate;
 If cost*rate>max_cost_rate[cat] then
  begin
   If rate>RECOMMEND_RATE then max_cost_rate[cat] = cost*rate;
   If ACCEPT_RATE <= rate < RECOMMEND_RATE then
       max_cost_rate[cat] = max_cost_rate[cat] + 0.5*(cost*rate - max_cost_rate[cat]);
  end;
```

(2)

Rating certificate used here, is a key factor for protect users from merchants who sell low quality or fake goods, or do not have adequate infrastructure to assure that user gets the product file complete and correct (e.g. poor file server). To obtain a rating certificate for a specific merchandise x (mostly a file), merchant sends a request with the product file to the bank; bank gives him a temporary certificate called Rating Discount Certificate (RDC) which includes an Electronic Merchandise Identification Code (EMIC) and hash of the product file, and sets a discount for instance 10 percent for who attends in the rating process.

When user's agent gets RDC instead of Rating Certificate notifies the user, user rates the product and the agent sends the rating to bank. Merchant receives checks from users but can not cash them because the EMIC in the checks is not approved yet so he stores them all. When ratings of users get to a specific volume (e.g. 10000 rating for each cent of product's value), bank requests the checks withdrew by the users who bought the good with the EMIC. Merchant sends these checks to bank, if the rate by users is acceptable bank continues the process, otherwise the users get refund and that merchandise is no longer salable.

The rest of process is as follows. Bank applies the discount for attending in rating process. It is done by randomly selecting 10 percent (discount rate) of the checks and crediting the check value to corresponding account this way statically, users get 10 percent discount. Then bank applies the normal F function to checks and selects the upgradeable ones, subtracts discount value and then pays the merchant. Finally bank issues a rating certificate to merchant for that merchandise and approves the corresponding EMIC.

If the selling volume of the merchandise gets to the required volume, but the users do not attend in rating, so that the ratings at bank are not as much as checks gained by merchant, merchant can request cashing them from bank, in this case bank applies discount for users attended in rating and pays the merchant same as above but do not issue rating certificate and holds on for more ratings.

5 The Proposed Scheme

In this chapter we combine the proposed trust model and enhanced MR2 scheme and present the complete TMR2 protocol. This protocol consists of 7 sessions shown in figure 1. In this diagram R and B are rating and financial agents of the micropayment provider (Bank), M is the merchant's agent, F is the merchant file server which sends the product files, and U is the user's agent. The sessions are specified with numbers in the figure 1. We explain the session in the same order.

5.1 Request for membership

User completes the membership form and sends his information to the bank. Bank processes the requests and perform the legal checks then sends a notification email to the user. User downloads the user agent via SSL from banks site. User agent in first run gathers random number from user's system and produces public and private key pair. Then it encrypts the user's identity and his public key with bank's public key and sends them to bank. Bank signs the public key of user with his private key and sends the resultant user certificate (UID certificate) to user agent.

5.2 Purchase from an unknown merchant

User agent sends UID certificate to the new merchant and requests registration, merchant agent validates the certificate and registers user to his customers list. Then he sends the merchant identification certificate (MID Certificate). User agent verifies this certificate and adds a record for this merchant. Also adds this merchant in max_trans_rate[m] array. The process continues same as normal purchase.

5.3 Normal purchase

First user selects the product he wants to buy. The link provided for each product contains the price, category and EMIC of the product. As user clicks on the link, user agent is activated and sends a purchase request to merchant agent. Merchant agent sends the Rating certificate for the specified product to user agent; user agent receives the certificate and verifies it.

Then user agent uses the rating certificate to calculate the trust on buying this product, if the trust value is greater than the product's price the agent will automatically buy the product otherwise it asks the user about buying it. In the prompt the price, rating and calculated trust for the product is shown to user and user decides to buy it or not. To continue the purchase process user agent sends a micro check to merchant agent. Merchant agent verifies the check and redirects user agent to file server so that it receives the product file. Optionally user agent can hash the file and compare it to the hash of the file in the rating certificate.

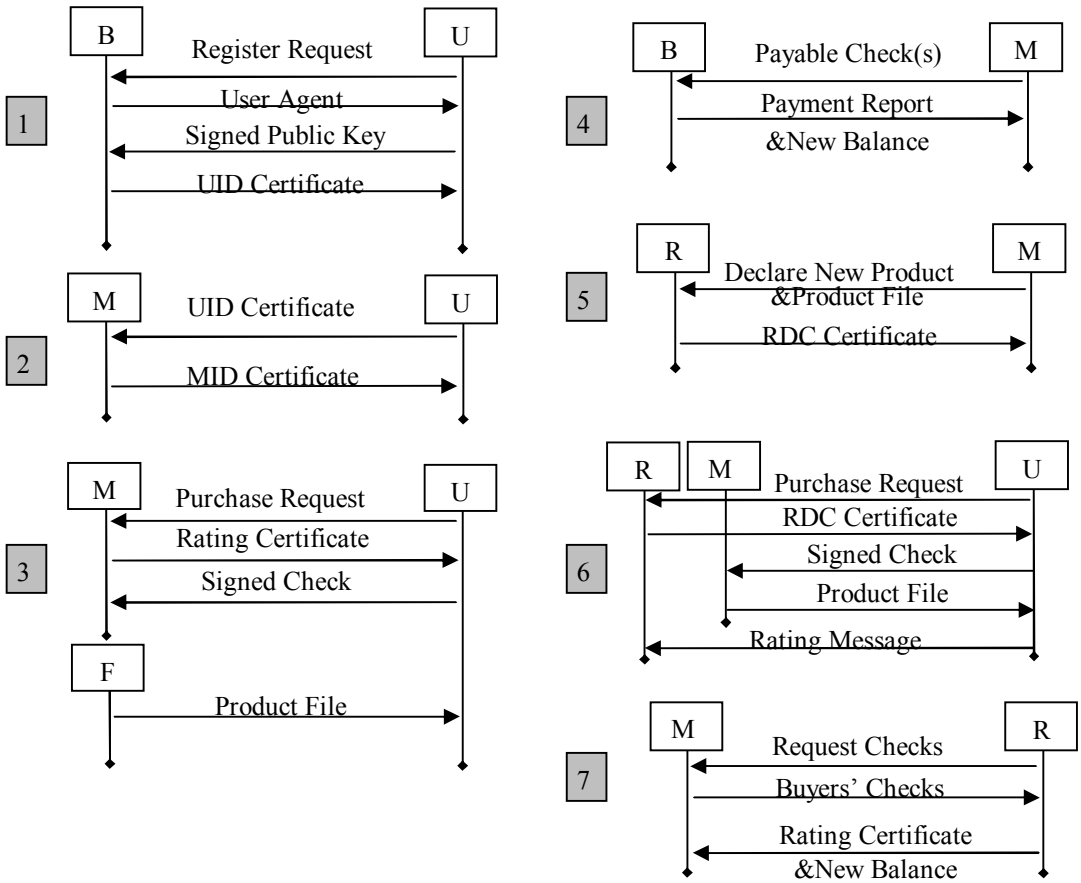

Fig.1 - Dataflow in different sessions of TMR2

If user assign a rating to this purchase (it is optional) user agent updates the max_trans_rate[m] and max_cost_rate[c] tables. User can also request to send this rating to bank, especially if he is not satisfied or conversely found the product worthwhile, in this case the rating is sent to rating agent of the bank and the rating is included in new rating certificate after the current one expires. This process is shown in figure 2.

Rating certificates have an expiration date which is based on the price of the product and its rating degree. The new rating data gathered from the customers will be included in the rating certificate at the time the merchant requests for the renewal of the rating certificate.

5.4 Cashing the checks and settlement

This process is done off-line as described in sections 2-3 and 2-4. There is just one modification here, bank do not pay the checks that their EMIC are not approved (the products that do not have Rating certificate). For more details about this case refer to section 4-1.

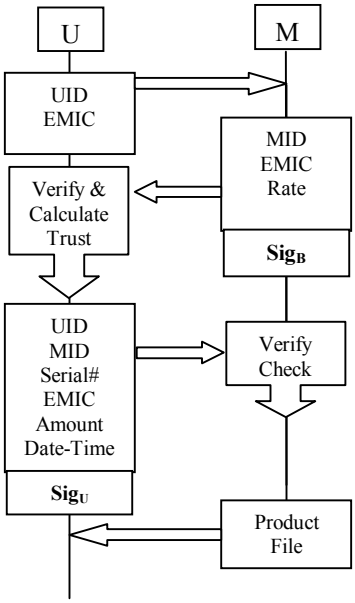

Fig. 2 - Normal Purchase

5.5 Registering new product

For each new product merchant need to acquire a rating certificate and register it in system. Refer to

section 4-1 for the process of acquiring a rating certificate. Merchant agent sends the price, category and product file to the rating agent of bank. Rating agent appends a merchant identifier (MID) to the hash of product file and produces the EMIC. Hash digest of these fields and a random number forms a code called Discount Code for Rating (DCR). Rating Discount Certificate (RDC) consists of DCR, EMIC, MID and product category signed by bank. Figure 3 shows this process.

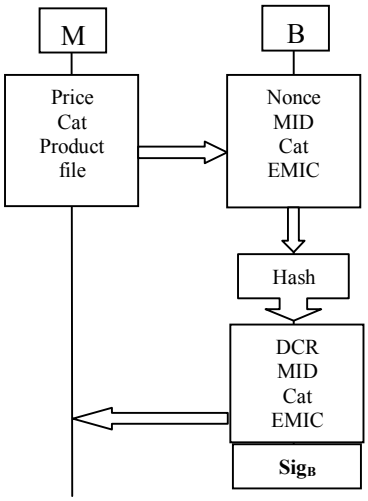

Fig. 3 - Registering new product

5.6 Buying new Products

In the purchasing process, if the seller agent sends a RDC instead of a rating certificate, the customer agent prompts the customer about this product and asks him whether to buy the product or cancel the process. If the customer is willing to buy the product the process continues as a normal purchase.

Ultimately, if the customer rates the product and is willing to attend in the rating, the customer agent signs the rating and the DCR with the customer's private key and sends it to the rating agent of the bank.

Although participating in the rating is optional but all customers are protected by the trust system. If an unrated purchased product is defective the customer can request to block the check he paid to buy it. In this case bank stores the UID of this user and the RDC. If the rating for this product failed to achieve the desired point and amount of the complaining customers grow, bank will not pay the checks paid to buy that product, so the customers are protected.

5.7 Issuing rating certificate

When the ratings for a new product achieves to the critical volume, rating agent requests buyers' checks from merchant agent, merchant agent sends all saved checks for that product. If rating is low the process ends here and the users' accounts are refunded. Otherwise rating agent separates the checks of the users who attended in the rating, signs them and uses F function to select 10 percent (discount percent) of them, and then credits their related accounts with the products price. This way 10 percent of users who attended in rating get the product for free which is statistically equal to 10 percent discount. After that rating agent applies the normal payment on all the checks and credits the merchant's account with value of payable ones subtracting the discount paid to users. Finally rating agent issues a rating certificate for this product. This certificate contains EMIC, Rating, Number of ratings, hash of product file, MID and expiration date, and is signed by bank.

6 Implementation and Results

We implemented our proposed protocol on a Java platform, to study its characteristics. One important issue is the overhead of adding the proposed trust model to the MR2 micropayment. Most of the sessions added (e.g. rating operations) happen limited times and their overhead is negligible. Some extra condition testing (e.g. as checking the EMIC to be approved before paying a check) is done off-line by bank so they are easily handled. Therefore, we analyzed the on-line normal purchase process which is the most important session in the system and happens more frequently. To evaluate the computational overhead of trust calculation, we coded a user program that generates requests and pays checks to merchant agent repeatedly and saves the results. We compared the results of running system without trust (just MR2), with the full system with trust. Table 1 shows the result of runs on an Intel Pentium4, 3.4 GHz, with 1 GB of RAM and Windows XP - Professional Edition operating system. Because our intention was to compare the computational overhead we used 1024 bit RSA keys. For real proposes, using online-offline signatures enables handling much more transaction per second.

To evaluate the overall performance of the system, we simulated a case in which 100 users send 100, 1000 and 10000 checks with random value between 1 to 20 cents (same random sequence was used for all runs) and $s=0.001$. The results in Table 2 show that our modification (fixed payment value) balances the payments made by each party if users

spend an adequate amount of checks (in our configuration 1000 checks) and is fair. Also we checked the penalty system with a user that randomly double spends a check and it was detected in all the tests.

.

Table 1 – Processing overhead of the trust model

	With Trust			Without Trust			Overhead	
	Number Of Trans.	Proc. Time (ms)	Payable checks	Number Of Trans.	Proc. Time (ms)	Payable checks	Added Time (ms)	% added
Run1	20000	195485	21	20000	188125	22	7360	0.039123
Run2	20000	192547	18	20000	187985	23	4562	0.024268
Run3	20000	193563	22	20000	185672	19	7891	0.0425
Run4	40000	400891	42	40000	384922	39	15969	0.041486
Total	100000	982486	103	100000	946704	103	42719	0.03771

Table 2 - Value of payments by different parties

Number of Checks	Actual Value	Paid to Merchant	Collected from User
10000	10506280	10462000	10418230
1000	1050628	1001000	936614
100	105192	99000	37052

7 Conclusion and Summery

Many micropayment schemes have been proposed but most of the micropayment implementations have been unsuccessful. An important barrier for expansion of micropayment is the mental cost for users to trust an unknown merchant and pay for his product or service. In this paper, we proposed a trust model from user point of view and combined it with MR2 micropayment scheme and called the new scheme TMR2. This trust model is supported by micropayment provider and assures the users that they will not be charged for in case the product is not satisfactory or it is corrupt.

We implemented TMR2 and our results shows that the computational overhead of this trust model is less than 4 percent which is acceptable considering the benefits of the trust model for users and for the promotion of micropayment. Since our trust model adds a signature validation, and MR2 needs a signature calculation, 4 percent overhead can be verified comparing to the results of [26].

To the best of our knowledge, there is no research on integrating the concept of user trust to micropayment systems other than our work in this paper.

References:

[1] Micali, S., Rivest, R. L., "Micropayments revisited", B. Preneel, editor, Proceedings of Cryptography Track at RSA Conference 2002, pp.149–263, 2002.

[2] Jarecki, S., Odlyzko, A., "An efficient micropayment system based on probabilistic polling", Proceedings of the First International Conference on Financial Cryptography, LNCS Vol. 1318, pp. 173-192, 1997.

[3] Kniberg H., What makes a micropayment solution succeed, Master Thesis, Kungliga Tekniska Högskolan, Stockholm, November 2002.

[4] Odlyzko, A., "The case against micropayment systems", Proceedings of the 7th Int. Conf. on Financial Cryptography, Wright, R.N. ed., LNCS Vol. 2742, Springer, 2003.

[5] Nielsen, Jakob, "The Case for Micropayments", http://www.useit.com/alertbox/, January 25, 1998.

[6] Szabo, Nick, "The Mental Accounting Barrier to Micropayments", http://szabo.best.vwh.net/index.html.

[7] Van Someren, N., Odlyzko, A. M., Rivest, R. L., Jones, T., Goldie-Scot, D., "Does Anyone Really Need MicroPayments?" Proceedings of the 7th Financial Cryptography Conference (FC 2003), pp. 69-76, 2003.

[8] Puhrerfellner, Michael, An implementation of the Millicent micro-payment protocol and its application in a pay-per-view business model, Master Thesis, Technische Universität Wien, Wien, Austria, December 2000.

[9] Mahony, Donald O., Peirce, Michael, Tewari Hitesh., *Electronic Payment Systems for E-Commerce 2nd Ed*. Artech House, 2003.

[10] Coggins, Shai, "BitPass : Micropayment System", http://www.justmakemoneyonline.com

[11] Rivest, R. L., Shamir A., "PayWord and MicroMint: Two Simple Micropayment Schemes", Proceedings of 1996 International Workshop on Security Protocols, (ed. Mark Lomas), LNCS Vol. 1189, pp. 69-87, Springer, 1997.

[12] Lin, I., Hwang, M., Chang, C., "The General Pay-Word: A Micro-payment Scheme Based on n-dimension One-way Hash Chain", *Springer Designs, Codes and Cryptography*, Vol. 36, pp. 53–67, Springer, 2005.

[13] Lipton, R., Ostrovsky, R., "Micro-payments via efficient coin-flipping", Proceedings of the second Financial Cryptography Conference FC'98, February, 1998.

[14] Rivest, R. L., "Electronic lottery tickets as micropayments", Proceedings of the first Financial Cryptography Conference FC'97, Anguilla, British West Indies, February, 1997.

[15] Rivest, R. L., "Peppercoin micropayments", Proceedings of Financial Cryptography Conference FC'04, LNCS Vol. 3110, pp. 2-8, February, 2004.

[16] Vishnumurthy, V., Chandrakumar, S., Sirer, E.G., "KARMA: a secure economic framework for peer-to-peer resource sharing", Proceedings of the Workshop on the Economics of P2P Systems, Berkeley, USA, 2003.

[17] Garcia, F.D., Hoepman, H., "Off-line Karma: A Decentralized Currency for Static Peer-to-peer and Grid Networks", Proceedings of International Conference on Applied Cryptography and Network Security, LNCS Vol. 3531, pp. 364-377, Springer, 2005.

[18] Yang, B., Garcia-Molina, H., "PPay: Micropayments for Peer-to-Peer Systems", Proceedings of the 10th ACM conference on Computer and communications security, pp. 300-310, ACM Press, 2003.

[19] Jia, Z., Tiange, S., Tiange, H., Yiqi, D., "A New Micro-payment Protocol Based on P2P Networks", Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE'05), 2005.

[20] Hamad, A., Kouta, M., Afify, Y. M., "Evaluation of Probabilistic Payment Systems", Proceedings of the 8th IEEE International Conference on E-Commerce Technology and the 3rd IEEE International Conference on Enterprise Computing, E-Commerce, and E-Services (CEC/EEE'06), 2006.

[21] Prhonyi, R., Pras, A., Nieuwenhuis, L.J.M., "Second generation micropayment systems: lessons learned", Proceedings of the Fifth IFIP Conference on e-Commerce, e-Business and e-Government (IEEE 2005), Poland, October, 2005.

[22] Foley, S. N., Quillinan, T. B., "Using trust management to support micropayments", Annual Conference on Information Technology and Telecommunications, Waterford, Ireland, October, 2002.

[23] Sabater, Jordi, Sierra, Carles, "Review on Computational Trust and Reputation Models", Artificial Intelligence Review (2005) 24: pages 33–60, 2005.

[24] Ruohomaa, S., Kutvonen, L., "Trust Management Survey", Proceedings of 3rd International Conference on Trust Management (iTrust 2005), pp.77-92, 2005.

[25] Ioannidis, J., Keromytis, A.D., "Ditributed Trust", In Practical Handbook of Internet Computing, Munindar Singh (editor), pp. 47/1 - 47/16. CRC Press, 2004.

[26] Naor, D., Shenhav, A., Wool, A., "One-Time Signatures Revisited: Have They Become Practical?" Technical Report 2005/442, Cryptology ePrint Archive, 2005. http://eprint.iacr.org/2005/442/

[27] Fabio Roberto Pillatt , Francilene Procopio Garcia, A model for the trust handling on e-business transactions, *WSEAS Transactions on Systems*, Vol.2, Issue 1, January 2003, pp. 113-118.

[28] Muhammad Hanif Durad, Yuanda Cao, Zhu Liehuang, A Novel Evidential Trust Evaluation Algorithm, *WSEAS Transactions on Computers* , Vol.5, Issue 6, June 2006, pp.1254-1261.

[29] Geoff Skinner and Mirka Miller, Managing Privacy, Trust, Security, and Context Relationships Using Weighted Graph Representations, *WSEAS Transactions on Information Science & Applications*, Vol.3, Issue 2, February 2006, pp. 283-290.