An Adaptive Agent Approach to Micropayment System

Surachai Chaiyarangkitrat and Yongyuth Permpoontanalarp
Logic and Security Laboratory
Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi
91 Pracha-Uthit Road, Bangkok 10140, Thailand
Tel. (662) 470-9087 Fax. : +66-2-872-5050
E-Mail: Surachai.c@cdg.co.th, yongyuth@cpe.eng.kmutt.ac.th

Abstract: Micropayment is an electronic payment system for small value transaction. It needs to use a little amount of resources, such as communication and computation due to its small value. In other words, the processing cost for the micropayment must be less than the value of the payment. Several kinds of transactions are suitable for micropayment, eg. the purchasing of train tickets or digital newspapers. Since micropayment systems are designed for small-amount payment, the key factor for any micropayment system design is believed to be the minimization of resource consumption without compromising the standard security. In this paper, we propose an adaptive agent approach to credit-based micropayment system, which employs the concept of dynamic balancing between the resource consumption and the risk in the system. As a result of the dynamic balancing, our system not only solves the problem of global overspending but also uses fewer amount of resources than existing approaches. Our approach limits the amount of money spent by untrusted customers to all merchants. Thus, our approach provides a boundary of the global overspending. In addition, for trusted customers, our approach requires less scale of communication for verifying authorizations than all existing approaches.

1. Introduction

Micropayment is an electronic payment system for small value transaction. It needs to use a little amount of resources, such as communication and computation due to its small value. In other words, the processing cost for the micropayment must be less than the value of the payment. Such processing cost includes communication cost, in particular, that for merchants to verify the authorization of any payment made by customers. Several kinds of transactions are suitable for micropayment, eg. the purchasing of train tickets or digital newspapers.

There are two main approaches to micropayment, namely debit and credit systems. For debit-based micropayment, the existing approaches [3,4] suffers from the following problems. Either its electronic money can be used at particular vendors only, or it requires a great amount of communications. Even though an agent-based approach to debit micropayment has been proposed [6] to solve these problems, it suffers from a problem, the potential lost of electronic money. However, this problem is caused mainly by the nature of debit systems, that is, electronic money is pre-paid.

A number of approaches [1,2] has been proposed for credit-based micropayment. In such systems, a customer is authorized a specific amount of credits that the customer can spend at a vendor for a certain period of times. If the total amount of money spent at a vendor during the period of times is not more than the credit limit, then there is no need for verifying the authorization with brokers, thus reducing communication costs. Such kind of authorization verification is called *offline* since it does not use any communication at all. Even though these systems do not suffer from the lost of electronic money due to the nature of credit systems, they suffers from the global overspending, i.e. the overspending that may occur collectively at all vendors. In other words, the global overspending occurs when a customer spends an arbitrary amount of money at an arbitrary amount of vendors even though the customer does not overspend the credit at each of those vendors.

Since micropayment systems are designed for small payment, it is believed that the main design factor for any micropayment system is to minimize the resource consumption as much as possible while keeping the security at some acceptable level.

We argue in this paper that the more suitable design factor should be the *dynamic balance between the resource consumption and the risk in the system*, in particular banks' risk. In particular, if the risk is high, then the system may require more communication in order to reduce such risk. In converse, if the risk is low, the need for resource consumption is also low. Such adjustment for the resource consumption is carried out dynamically as the system proceeds.

In this paper, we propose an adaptive agent approach to credit-based micropayment system, which employs the concept of dynamic balancing between the resource consumption and the risk in the system. Our system not only solves the problem of the global overspending but also uses fewer amounts of resources than the existing approaches as a result of the dynamic balancing. In particular, for untrusted customers, our approach limits the amount of money that the untrusted customers can spend at all vendors. Moreover, for trusted customers, our approach requires less number of communications for verifying authorizations than the existing approaches.

2. Background

There are two widely known approaches to credit-based miropayment system, namely Minipay and Blaze's approach.

2.1 Minipay

In the Minipay system [2], a customer will be given the certificate based on daily basis, which includes customer's public key and customer's daily credit limitation. The customer can make a payment to the merchant without the

need for the merchant to verify the payment unless customer's daily credit is exceeded. Thus, the authorization checking can be performed by merchants and it can be done without any contact to the banks. This approach requires the use of asymmetric cryptography.

2.2 Blaze's system

Blaze's approach [1] is a credit-based system which is similar to Minipay in that in both systems customers are given certificates for customers' credit information. This certificate will be verified for its authorization to perform offline payment without the need to contact any third party (eg. Bank).

However, certificates in Blaze's system contain more information than those in Minipay. In particular, in Blaze's system, customers' certificate contains information about circumstances under which transactions can be authorized in an offline fashion. For example, some transactions (alcohol, tobacco, pornography, binding contracts, etc.) might be restricted to adults. Thus, it is a simple matter to encode a requirement for an "adulthood" credential in the merchant selling conditions.

The risk management is introduced and is used to determine transaction restriction which will be enclosed in customers' certificates. Examples of the transaction restriction are the amount of goods that can be purchased, and the period of times that any purchase can be made. Thus, the risk analysis in Blaze's approach is for limiting customers' purchase behavior, and for preventing attackers from using other users' certificates for their payment.

3. Adaptive Agent Based Approach to Miropayment System

3.1 Overview

There are two kinds of agents in our system, *customer agents* and *vendor agents*. Both agents operate on behalf of brokers. The tasks of customer agents are to check for the availability of customer's credentials (active credit) and to communicate with brokers for additional credential request. The task of vendor agents is to verify the authorizations of any payment made by a customer. Vendor agents would make their decision to perform either offline or online authorization checking, based on the customer's trustworthiness that is obtained by performing the risk analysis on the customer's behavior.

In our system, the electronic money is called *active credit* issued by brokers at the time that the customer starts a customer agent. This active credit cannot be used after the agent terminates itself (or shutdown). Note that the termination of an agent by itself is different from the temporary termination of an agent requested by the customer. To distinguish the former from the latter, we use *shutdown* to refer to the former. In case of the agent termination, an agent would update the customer's remaining active credit to its broker before it would actually stop working. Customer agents would make their decision to either proceed or shutdown, based on the customer's trustworthiness.

When the customer wants to make a payment, the agent will check for the availability of the customer's active credits. If the active credit is not enough for the payment, the customer agent will request for some additional active credits from the broker. If the request is refused, then the customer cannot make the payment.

Thus, the only way that the global overspending problem may occur is for a customer to terminate the agent by *intentionally forcing* the customer agent to fail. In such situation, the broker would not know the customer's actual remaining active credits. Then, when a new customer agent is (re)started, it will start working with the customer's incorrect remaining active credits that are more than the customer's actual remaining credits.

To deal with the global overspending, we employ the risk analysis on customer behavior to determine the values of two controlling parameters. The first is, called I, the maximum number of times that an agent may terminate by failure. This kind of agent termination may be caused only by either the accidental failure of the customer's system or the customer's intended action of failure. In order to limit the amount of global overspending, the value of the second parameter, called AAC, that is the amount of active credits that would be issued to a customer on a request, is determined. Then, the limit of global overspending is set to $AAC * I$.

There is another parameter that is used for dealing with offline authorization checking, and it is called Preauthorized Active Credit (PAC). PAC is the maximum amount of money that a customer can spend at a vendor without requiring the vendor to verify the authorization of the payment with brokers. Thus, vendor agents would use PAC to decide for either offline or online authorization checking.

Using the risk analysis on customer behavior, our agent system is *adaptive* in that it will adjust dynamically the three controlling parameters to balance between the risks in the system and the resource consumption in the system.

3.2 The Detail Protocol Description

Agent Starting Protocol: When a customer wants to start the payment, he/she needs to activate broker function at the customer site by (re)starting an agent. The customer can restart the agent by sending an agent restart request to the broker. If the number of times that the agent has been shut down exceeds I, then the request for restarting the agent will be refused. Otherwise, the request is granted.

Agent Starting Response: If the request is granted, then the broker will do the following.

B→C: E(Agent, BC_Key)s

Where Agent = (Agent Program, PAC), I = Maximum Allowed Agent Fail Count, PAC = Preauthorized credit, and BC_Key is the share key between broker and customer.

If the request is refused, the customer agent will be shutdown permanently since the system is considered to be in high-risk level. The customer needs to contact the broker in face to reactivate the customer agent.

Active Credit Requesting Protocol: If the customer still has credit in the account, more credit can be granted.

1. C→B: (Customer ID, A request for Active Credit)

2. B→C: E (Active Credit, BC_Key)

Where Active Credit= (Active Credit ID, customer ID, value, expire, h(A_Key//Active Credit ID//customer ID//value//expire)), Active Credit ID = (Broker ID, Agent ID, Running Number), and A_Key is the key that is shared between customer agent, vendor agent, and broker.

Otherwise, the request will be rejected.

Purchasing Request Protocol: When a customer is interested in a product, the customer will request for the product information (e.g. price, product ID), in order to correctly issue the payment requisition.

1. C→M: E (Purchase-requisition, Customer ID, SK)

2. M→C: E (Purchase-response, SK)

Where Purchase-requisition = (Product ID, quantity of Product) and Purchase-response = (Ordering ID, price) and SK stands for Session Key which refers to the key used between the customer and the merchant.

Payment Protocol: In this protocol, the customer agent will check for the availability of the Active Credit.

1. C→Customer Agent: (Purchase-response, Payment Request)

If the remaining active credit is enough for a payment, the customer agent will create an e-coin and deduct the remaining active credit accordingly.

2. C→M (via CA): E ((payment-request, e-coin), SK), E (SK, A_Key)

Where e-coin = (PAC, e-coin ID, merchant ID, value, expire, h(A_Key//Preauthorized-Credit//e-coin ID//merchant-ID//value//expire)), and E-coin ID = (customer ID, Running number)

If the Active credit is not enough for a payment, the customer agent will automatically request for additional active credits.

Payment Response Protocol: In this protocol, merchant will check for the remaining of credit that can accept from the customer. If more credit can be accepted, the payment request will be granted.

1. M→C: payment-response

Otherwise, merchant will need to ask for authorization from broker before grant the payment request.

2. M→B: (Value of PAC, Customer ID, Merchant ID, request for updating customer account)

Also, broker need to check for the account of that customer. If the account is still in good status, broker will authorize for that payment.

3. B→M: (New PAC, Customer ID, Merchant ID)

Otherwise, payment is rejected.

Customer Agent Termination Protocol: When the customer wants to terminate the agent, one more communication with the broker is needed in order to update the remaining active credit to the broker.

1. C→B: Request to shutdown the Agent, Customer ID, Active Credit

2. B→Customer Agent: Order to terminate the agent

where "Order to terminate the agent" means an order from the Broker to shutdown Customer's agent.

3. Customer Agent→B: Termination response

After the "*Order to terminate the agent*" is sent from the broker, the customer agent will be temporarily off. If the customer wants to make a payment with any merchant again, the customer needs to go back to first and second step to reactivate the customer agent and ask for the new active credit.

3.3 Balancing the resource consumption and the risk in the system

We propose a system which determine dynamically the three parameters: Allowed Agent Fail Count (*I*), Allowed Active Credit (*AAC*) and Preauthorized Credit (*PAC*) by using customer trustworthiness. Such dynamic determination of the parameters achieves the effect of the dynamic balance between resource consumption and the risks in the system.

The three parameters will be adjusted according to a parameter called *Trust/Risk Ratio* determined by the broker based on the customer profiles and the likelihood of customer fraud, which can be detected by both *customer agent* and *merchant agent.*

The basic of risk-management policies can be described as the relationship of the three parameters with the customer trustworthiness.

3.3.1 Relationship of PAC to each type of customers

$$\text{Trusted customer: } (\Sigma PAC_i/n) - PAC_n < 0$$

$$\text{Moderate Customer: } (\Sigma PAC_i/n) - PAC_n = 0$$

$$\text{Untrusted Customer: } (\Sigma PAC_i/n) - PAC_n > 0$$

$$0 < PAC_{min} < PAC_i < PAC_{max} = \text{Total Credits allowed}$$

Where PAC_i = i-th value of PAC, PAC_n = Last value of PAC and PAC_{min} = Minimum Value of Certificate in order to satisfy the Minimum Cost Effectiveness (pre calculated by Broker).

3.3.2 Relationship of AAC to each type of customers

$$\text{Trusted customer: } (\Sigma AAC_i/n) - AAC_n < 0$$

$$\text{Moderate Customer: } (\Sigma AAC_i/n) - AAC_n = 0$$

$$\text{Not trusted Customer: } (\Sigma AAC_i/n) - AAC_n > 0$$

$$0 < AAC_{min} < AAC_i < AAC_{max} = \text{Maximum account value}$$

Where AAC_i= i-th value of AAC, AAC_n = Last value of AAC, and AAC_{min}= Minimum Value of Allowed Active Credit in order to satisfy the Minimum Cost Effectiveness (pre calculated by Broker).

3.3.3 Relationship of I to each type of customers

$$\text{Trusted customer: } I = I_{max}$$

$$\text{Moderate Customer : } 1 < I < I_{max}$$

$$\text{Not trusted Customer: } I = I_{min} = 1$$

$$0 < I < I_{max}$$

$$\text{Global over spending limitation} = I_{max} * AAC$$

Where I_n = Last value of I, I_{max} = Maximum value of I which is predefined by broker to limit the maximum suffering from each customer (calculated from Imax = Global over spending limitation/AAC)

Global overspending limitation for a customer is predefined from the customer's financial status (e.g. salary).

The value of each parameter may be different among each customer. Moreover, such parameters can be controlled by using the weight on each parameter. Parameters assigned higher weight will be more significant than those with lower weight. We can define the value of each parameter for individual customer by using the *Trust/risk ratio* as follows:

$(W_I * I' + W_{AAC} * AAC' + W_{PAC} * PAC') = Trust/Risk\ Ratio'$

Where Trust/Risk Ratio' = percentage of Trustworthiness of each customer which is pre calculated by broker,

W_{PAC} *= Weight of PAC to evaluate the significance of PAC,*

W_{AAC} *= Weight of AAC to evaluate the significance of AAC,*

W_I *= Weight of I to evaluate the significance of I,*

I' *= Percentage of I, calculated from:*

$$I' = (I * I_{min}/I_{max}) * 100\%,$$

AAC' *= Percentage of AAC, calculated from:*

$$AAC' = (AAC * AAC_{min}/AAC_{max}) * 100\%, \text{ and}$$

PAC' *= Percentage of PAC, calculated from:*

$$PAC' = (PAC * PAC_{min}/PAC_{max}) * 100\%$$

We have randomly generated a purchasing behavior of a customer and applied this concept to it. Based on the assumption that this purchasing behavior derives from a trusted customer, the graph of relation between communication rate and number of purchasing transaction is show in figure 1. The graph indicates that the trusted customer requires lower communication when number of transaction is increasing.

We also do the comparison of our Adaptive agent with the static system as shown in figure 2.

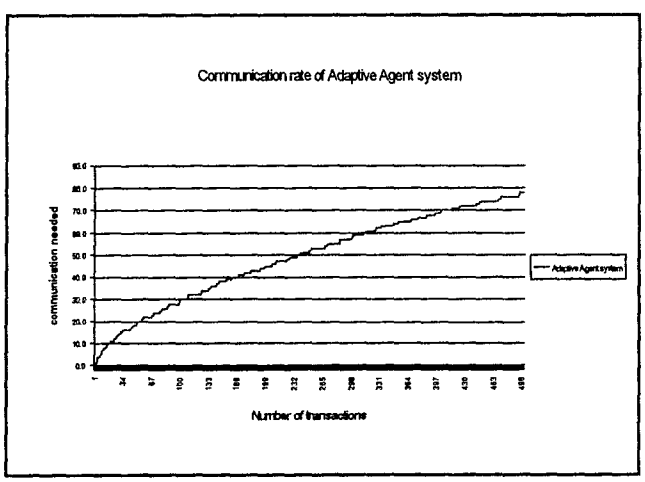

Figure 1: Communication Rate of Adaptive Agent System

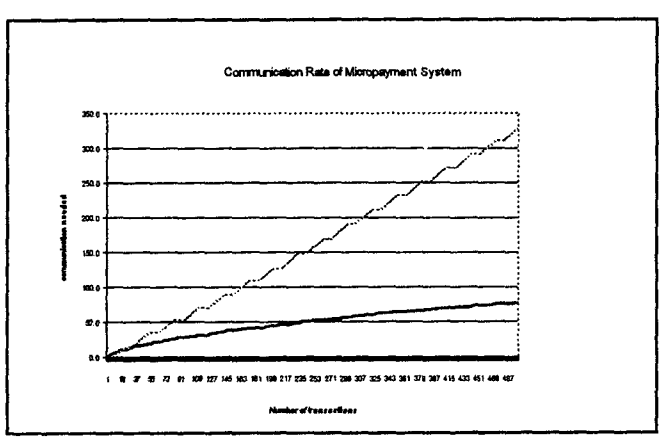

Figure 2: Communication Rate of Adaptive Agent System and Static system

4. Conclusion

We have proposed an adaptive agent-based credit micropayment system. Our system offers a solution to the agent state losing in the previous version of Agent-based debit Micropayment system [6]. Also, our system provides a solution for the global overspending from which all existing credit-based micropayment systems suffer.

Furthermore, in this paper we argue that the main design factor for credit-based micropayment system should be the dynamic balance between the resource consumption and the risk in the system. Due to the dynamic adjustment for such balance, our system offers services to customers more efficiently and securely than the existing systems, which are static.

Currently, we are developing a software prototype of our agent system. As a future work, we will apply our system to real-world case studies.

Acknowledgements

The second author would like to acknowledge support from National Research Council of Thailand under the research project "Problems and Approaches of Secure Electronic Commerce for Thailand".

References

[1] Matt Blaze, John Ioannidis, and Angelos D.Keromytis, "Offline Micropayments without Trusted Hardware," *In Proc. of the International Conference on Financial Cryptography, Springer Verlag, 2001.*

[2] A. Herzberg and Yochai Hilik, 1997, "MiniPay: Charging per Click on the Web," *In Proceedings of Sixth International World Wide Web Conference.*

[3] Manasse, M.S., 1995, "The Millicent protocols for electronic commerce," *In Proceedings of the First USENIX Workshop on Electronic Commerce, USENIX,* July 1995.

[4] Tomi Poutanen, Heather Hinton, Michael Stumm, 1998, "NetCents: A Lightweight Protocol for Secure Micropayments," *Proceeding of the 3rd USENIX Workshop on Electronic Commerce,* Auguest 31-September 3, 1998, Boston, Massachusetts, USA.

[5] Asokan, N., Jason, P., Steiner, M. and Waidner, M., 2000, "State of the art in electronic payment systems," *Advances in Computers, Edited by Zelkowitz, M.V., London, Academic Press,* pp. 425-449

[6] Urailuck Yuwathiticharoenwong and Yongyuth Permpoontanalarp, 2001, "An Agent-Based Approach tc Micropayement system," *In Proceedings of Internationa Conference on Information Technology,* Thummasa University Press.