Military Technical College
Kobry El-kobbah,
Cairo, Egypt

5th International Conference
on Electrical Engineering
ICEENG 2006

Mobile Agent as a digital coin For clone detection and double spending prevention

Mostafa Salama*
Mohamed Kouta**, Mohamed Abu-Rizka**

* M. Sc. Engineer, Arab Academy for Science and Technology, Heliopolis, Cairo, Egypt.
**Professor of computer sciences, Arab Academy for Science and Technology, Heliopolis, Cairo.

ABSTRACT

Mobile agent and digital coin represent two growing technologies in E-Commerce systems. However, mobile agent systems suffer problems of cloning agents and the inability to detect the user who cloned the agent. While digital coin suffers a problem of spending the same coin more than once i.e. double spending. Merging these two technologies into one scheme may solve such arising problems. Lam-Wei's Scheme [1] proof that the double spending detect-and-accuse cloning algorithm in e-cash can be used to detect-and accuse cloning offenders. The presented paper follows an opposite approach to this scheme as it implements a digital coin as mobile agent using some cryptographic concepts. This proposed scheme implements an E-cash system that prevents the double spending problem.

1. Introduction

Digital coin, in other words e-cash, is a cryptographic technique that was first presented by Chaum [2]. It is the electronic counterpart to the physical cash (paper bills and coins) in real life. The life cycle of the e-cash has four main phases which are setup of the bank and users, withdrawal of the coin from the bank, payment between two different users and deposit of paid coins to the bank. One of the security threats on e-cash system is spending the same digital coin more than once (double spending). Nearly all schemes that implement e-cash do prevent double-spending rather than it checks for duplicates after the agent is returned back to the bank (deposit), and in this case (and this case only) the identity of the coin's owner can be detected, as the anonymity of the coin's owner is guaranteed by the system. The anonymity of the user is a key parameter in E-cash system.

On the other hand mobile agent technology offers a new computing paradigm in which an autonomous program which is working on behalf of its owner. The life cycle of the Agent, according to Foundation for Intelligent Physical Agent (FIPA), passes through different states which are initiated as; active, suspended, waiting, deleted and transition state. The agent can suspend its execution on a host computer then it migrates itself to another agent-enabled on the network, and finally it resumes the execution on the new host, according to [11]. One of the security threats on mobile agent systems is the unauthorized mobile agent cloning. Some systems like [12, 13] addressed this problem as an open issue that is difficult to solve. Baek [10] proposed a clone detection system in which a central site oversees migration. Lam-Wei [1] proposed another e-cash based system for mobile agent to detect mobile agent clone detection.

However the security threats on both technologies are still the deployment bottleneck. Both technologies play an important role in E-commerce and especially in E-payment system. E-cash is used in electronic transactions as medium of electronic payment [2], [3,4]. Other systems and Mobile agent are used for purchasing and payment in the electronic transaction [5,6]. These two systems are integrated by Lam-Wei to serve the requirement of clone detection.

In this paper, both systems are integrated to solve the problem of double spending in order to prevent double spending during payment and not during the deposit of the coin. In section 2, the four phases of an E-cash system are introduced. Then in the next section the previous schemes for implementing the E-cash are discussed. In section 4, an overview of the proposed scheme is presented. The proposed scheme is discussed in details in sections 5, 6, 7 and 8. The conclusion showing the presented scheme as a new approach towards an ideal E-cash system is given in section 9.

2. E-Cash system

General Scheme of an e-cash system consists of 4 phases; First phase is Setup which is the preparation of the bank and users for the later protocol of communications and the keys to be used. The second phase is the withdrawal phase, where the user receives the coin, that is signed by the bank, generally the user creates the coin and blinds it, then send it to the bank blinded to be signed, the bank replies by the results to the user, who receives the coin blinded and signed then the user finally un-blinds the coin. Third phase is the payment phase, where the user sends the coin to merchant. Usually the merchant sends a challenge to the user and waits to check the response of the user to be sure that the coin is of a correct format, and that the coin includes the user Id inside to be sure his identity will be revealed if he double spends the coin and to be sure that he is the owner of the coin. The final phase is the deposit of the coin, the merchant sends the coin back to the bank, where the bank checks that coin is of correct structure and that the responses of the agent are correct.

3. Previous Schemes implementing E-cash requirements

The challenges that face E-cash systems are anonymity, offline, transferability, divisibility, overspending prevention, blackmailing and money laundry prevention. Most of the E-cash schemes starting from Chaum's system [2] that implements offline and anonymity Requirements. Overspending prevention Requirement is not implemented in most E-cash systems, except in brand’s system [3], as they only detect the user who makes the spending of the same coin more than once during the deposit of the duplicate coins. Brands scheme implements an observer (counter) in wallet (the electronic card), such that in withdrawal this observer incremented by the value withdrawn and in payment of a coin of a certain value, this counter is decremented by this value, So that the user can not pay an extra amount more than he withdrawn from the bank. Key parameter of Brand system is that the prevention of double spending depends on the observer that is inserted by the bank inside the smart card, and so it is a hardware dependent system rather than cryptograph and software dependence. Blackmailing and money laundry prevention is implemented using coin-owner tracing mechanism that is implemented using trustees (third part trusted servers), this mechanism is used in [8] and [9]. Owner tracing protocol traces the identity of the owner of a specific coin while Coin tracing protocol traces the coin(s) originated from a withdrawal. The ability to do this protocol is used by trustees only under the permissions from the authorities. Divisibility requirement is implemented by systems like [14] and [15] that uses divisible cash in order to solve the problem of coins of values larger than paying value i.e. problem of change. Transferability requirement is implemented only in a transferable cash scheme approach that appears in [16] that declares that size of coin increases with a number of bits after each payment. This happens in order to allow the bank to identify the people who use the coin more than once after a chain of payments. This approach is confirmed in [17] that if transferability is going to be used, then the size of the coin must increase with each payment and then this approach is used by [1] to prevent the cloning of agent.

4. Overview of the proposed scheme

As a modification of the previous scheme, the coin will be represented as a mobile agent. So, this agent will be named as "coin agent". The environment of this agent will consists of three types of hosts, the trustee which is the creator and activator of the coin agent, the bank who adds the signature to the coin agent, and finally the user and the merchant who are considered as the users of the coin. This agent will contain fields and functions; one of the fields that differentiate between the considered coin agents is the serial number, since the instance of the coin agent could be considered as an electronic file, so we could generate a hash code for such instance using one way hash function. The usage of this hash code will be useful for applying bank signature on and also to proof that no one change in the structure or functionality of the agent and it is still activated and working probably during its life cycle.

In withdrawal of this coin, the coin agent will be created and activated by the trustee, and hash code will be formed for this coin agent, then the agent will migrate carrying its hash code and leaving its serial number behind at the trustee. The bank will sign the hash code carried by the coin agent then it migrates back to have its serial number. So there will be no need for blinding the coin during withdrawal such that the trusted third party will show the information of who owns what coin, and this is according to the permission of government if any violation happens.

In Payment, before the coin agent migrates to the merchant it checks that there is no other copy from its instance on the user's device. The check on the structure of the coin using challenge and response technique between the user and the merchant will not be used. The hash code which is signed by the user's private key will decrypted by user's public key at the merchant , so when coin reaches the merchant he will be sure that who sends him the coin now is really the owner of the coin. When the coin agent reaches the merchant, the merchant will check that this coin agent is signed by the bank, he decrypts the signed hash code retrieved by the coin agent using the bank's public key, then he compares it by the hash code resulted by applying the hash function on the instance of the incoming coin agent.

5. Agent Structure

The agent structure, according to the previously proposed scheme, can be described by the following components and procedures:

The Agent Fields:

Serial Number: Serial number of the agent that is represented as the coin's serial number.
Coin value: it can be determined by a certain global code k.
UserId: The id of the user who makes the withdrawal, it may contain information of the owner of the coin, in order to migrate to him after withdrawal is done.
AgentHashCode: the code that is resulted after the instance of the agent is hashed, the hashing is done when the agent has the default (zero) values of the AgentHashCode, counter, let's call the coin agent in this format the empty-coin agent instance.
CreationDate: The date of creation of the agent, it is set when agent is activated.
Counter: a number that is incremented every second, start incrementing with every clock of the framework it contains.
Activating code: code resulted from applying a one way hash function on a secret code. This secret code is known only by the merchant.

The Agent Methods:

Activating method: this method is used to activate the agent. It runs only if the entity which activates the agent has the private activating code of the agent. Note that this private code is not saved inside agent, but it contains a code that is resulted after applying a one-way function on this secret code. Firstly the entity, that runs this agent, supplies the agent by this secret code. Then, the agent applies its one-way hash function on this secret code and compares it with its activation code. If they are matched, the activation process will be completed.

6. Proposed Mobile agent Modifications as a digital coin scheme:

The setup and the procedure of this proposed scheme can be summarized as follows:

***Setup*:**

a. Bank sets his public **(e_k, N)** and private **(d_k, N)** keys for each coin value, (e.g. a coin of a value k), where **N** is some composite whose factorization is known only to the bank..
b. User sets the public and private keys of the user.
c. Bank generates a global one way hash function **H** that gets instance of the intelligent agent as an input and produces a hash code.
d. The Government announced a trustee (Third party) which is responsible for the creation of agents and save IDs of the user and the coins they withdraw from the bank.

***Withdrawal*:**

a. User sends a request to withdraw a coin of k value from Third Trustee party.

b. The trustee creates an agent with a unique serial number; it activates the agent, and calculates a hash function on an activated instance of the agent in the form of an empty-coin agent instance.

c. The active agent gets the hash code **C** from the trustee and gets the user's ID. The agent then retrieves a blinding factor **r** from the trustee in order the get the hash code blinded., such that the blinded hash code C' defined as follows

$$\mathbf{C'=r^{ek}.C \bmod N}$$

d. On the other hand the active agent leaves both its serial number and the blinding factor at trustee. Then it migrates to the bank.

e. The coin agent gives the bank (another agent at the bank) the blinded hash code it gets from the trustee and gives the bank too the user ID of the user who withdraws the coin.

f. The bank signs the blinded hash code by the private key corresponding to the k value.

$$\mathbf{Sign(C') = (r^{ek}.C \bmod N)^{dk} = r.C^{dk} \bmod N}$$

g. Then the bank sends Sign(C') back to the agent and decrements the user account by k value then the agent migrates to the trustee.

h. The agent uses the blinding factor at the trustee to un-blind the signed hash code so that the hash code will be retrieved with the banks signature sign (C)

$$\mathbf{Sign(C) = (Sign(C'))/r = C^{dk} \bmod N}$$

i. Finally the agent migrates to the user, where the AgentHashCode is equal to **Sign(C)**.
[Bank encrypts the hash code by the key, public key, of the user so no one can use the agent except the one who has the private key of the user. This caution is required as the agent may migrate through the net where there is a possibility that someone tries to access the coin]

j. The coin agent located at the user now has a signed hash code **Sign(C)**, signed by the bank's private key.

k. The agent suspends its processing at the user until a need for payment appears. During waiting for payment, the counter of the agent will still count every second, note that counter could be of a bigger period (i.e. count every hour) according to the design and processing power.

***Payment*:**

a. Agent wakes up to start the payment process, Agent checks that there is no other copy of itself or the AgentHashCode at the user device.

b. Then the agent encrypts the AgentHashCode it contains by the public key of the merchant and migrates to merchant.

$$\mathbf{AgentHashCode = Sign(C)^{eM} \bmod N}$$

c. When the coin agent reaches merchant, it sends to the merchant (agent at the merchant) the AgentHashCode, the counter and UserId. Then the coin agent removes the AgentHashCode, the counter values, i.e. set these fields to the default values, then hashes the instance of the agent (while it is in the form of empty-coin agent instance) to get $\mathbf{C_1}$.

d. The agent at the merchant un-sign the AgentHashCode from the agent by decrypting it by its own private key

un-signed the AgentHashCode = (AgentHashCode)^{dM} = Sign(C)

e. Then the merchant decrypts the result by the public key of the coin of **k** value of the bank to get $\mathbf{C_2}$ then checks that these two results $\mathbf{C_1}$ and $\mathbf{C_2}$ that they are the same.

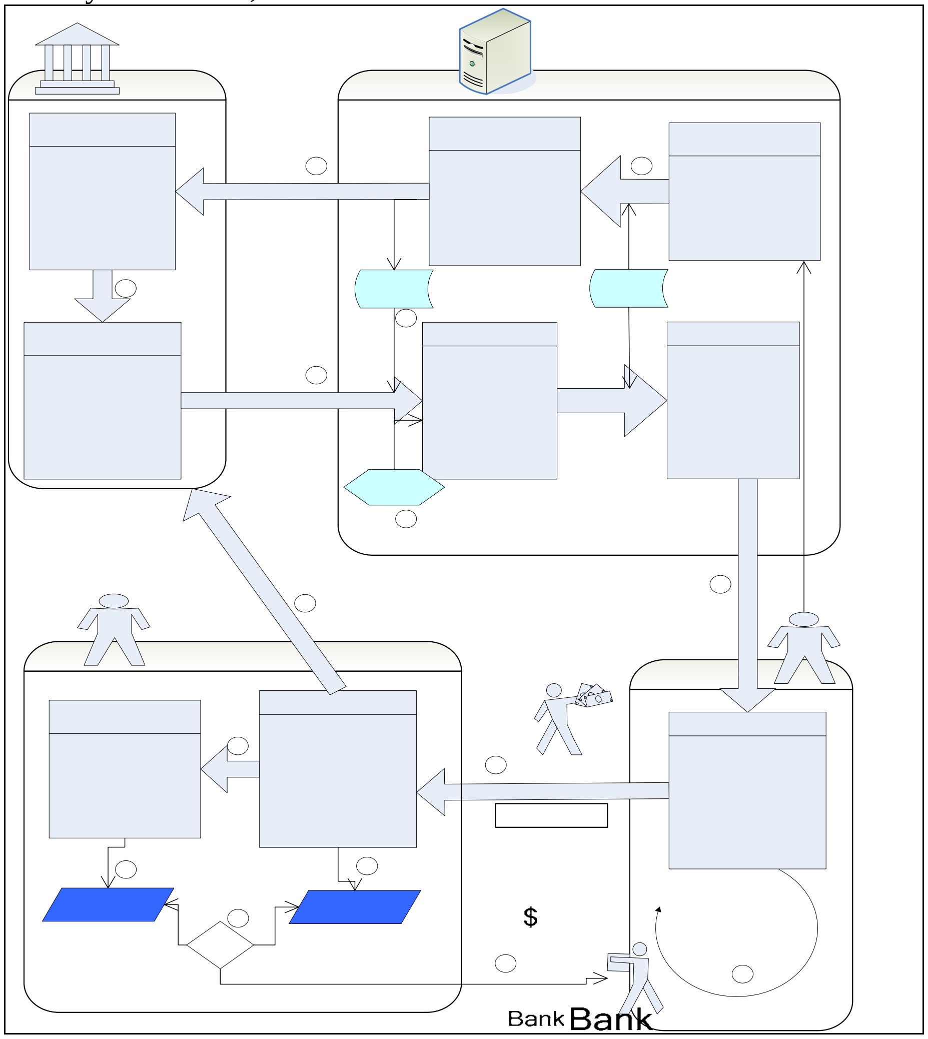
f. The agent at the merchant checks that [**counter** + **CreationDate**] date is the same as the current date with allowing small variance (error).

g. Coin agent then retrieves the AgentHashCode, the counter and merchant, then the Coin agent suspends for later payment or deposit.

***Deposit*:**

a. Agent wakeup to start the deposit process, the agent remove the Identity of the user from contents then migrates from the merchant or user to the bank. The bank checks the signature of the hash code using the same steps (b, d and e) in the payment process.

b. If all checks are done, the bank increments the account of the merchant or the user by the k value, value of the coin.



coin agent

Serial Number [Default

Coin value (k)

Creation Date

Activating code

2

Migration

Figure 1: Agent life cycle vs. Digital coin Scheme

7. Proposed Caution Actions

- In withdrawal, to guarantee that the bank can not try to hack the agent to know its serial number, the agent removes the serial number before migrating to the bank and after signing the hash code, the agent returns back to retrieve its code, the agent then migrates to the user.
- In case the coin agent in the payment process is bigger than the required value, the coin agent or another agent (e.g. agent that is responsible for searching for change) can migrate to search over the internet for another user or a merchant who has a change. Then repeat the process of migration and checking from original user to this new user who have change. While the change coin also processes the migration steps (payment) from that new user to the original user who wants the change. We could name this new agent as the change request agent.

8. System Attacks

- The attacks that could face this system is that the user can make a copy (snapshot) of all the memory of his device, and makes a payment then past the old snapshot to act as if he never did this payment, this problem is solved partially by the counter field in the agent, because if he did so, then after the snapshot is pasted, the (counter + CreationDate) will vary from the current date at the merchant during payment.
- The above attack may be not yet prevented as the user could change the counter inside object in the memory. Then the system could prevent this attack by being partially dependent on some hardware as a secured smart card. This smart card (electronic wallet) could contain private processing and memory unit as proposed in [7]. It will be used for saving the serial numbers of the withdrawn coins and for deleting that of the paid coins, however, both saving and deletion should be through the card itself, i.e. the user has no accessibility on such processing. So, on payment, the coin agent checks that its serial number is saved inside the private area of the card, and then the card deletes this serial after payment.

9. Conclusion

As shown in previous systems of E-cash, there isn't any technique that implements an ideal e-cash system and fulfills all requirements together in the same E-cash Scheme. So, it is proposed here an E-cash scheme that uses the capability of Mobile agent technology and the newly appeared cryptographic concepts in implementing most of the E-cash requirements. The proposed system solves also the problem of Lam-Wei's system proposed in [1] where there is no tie between the coin and the mobile agent. In our system the missing tie is considered as the hash code that moves with the agent where this hash code is signed by the bank's private key, so this tie can not be forged. No need for Divisibility of the coin, as the coin agent system will be supported by the ability to retrieve charges through using a net change request agent to find coins with the lower value. It implements also anonymity and offline withdrawal and payment requirement of E-cash. Blackmailing and money laundry prevention is also implemented as it uses a third trusted party. Finally, this scheme forms a resistance towards double spending processes as it makes use of the capabilities of mobile agent technology. So, the presented E-cash technique could be a new approach towards an ideal E-cash system that can be implemented in the future.

10. References

[1] *T. C. Lam and V. K. Wei, " Mobile agent clone detection using general transferable e-cash", International Conference on Information Security (InforSecu '02).*

[2] *D. Chaum, A. Fiat, and M. Naor, "Untraceable electronic cash." In Advances in Cryptology –CRYPTO' 88 (1988) 319 – 327.*

[3] *S. Brands, "Untraceable Off-line Cash in Wallet with Observers." in Advances in Cryptology-CRYPTO' 93 (1993) 302-318.*

[4] *N. Ferguson, "Single Term Off-line Coins." In Advances in Cryptology - EUROCRYPT' 93 (1993) 318-328.*

[5] *An Agent-Based Secure Internet Payment System for Mobile Computing: Romão and Miguel Mira da Silva, Departamento de Matemática, Universidade de Évora 7000 ÉVORA – PORTUGAL {artur,mms}@dmat.uevora.pt}*

[6] *A. Romo, M. Mira da Silva. An Agent-Based Secure Internet Payment System for Mobile Computing. Proceedings of the International Conference on Electronic Commerce'98, Hamburg, Germany (1998)*

[7] *T. O. Lee, Y. L. Yip, C. M. Tsang, K. W. Ng. An Agent-Based Micropayment System for E-Commerce. E-Commerce Agents, Pages: 247 - 263, (2001).*

[8] *E. Brickell, P. Gemmell, D. Kravitz, "Trustee-based Tracing Extensions to Anonymous Cash and the Making of Anonymous Change," Proc. 6th Annual ACMSIAM Symposium on Discrete Algorithms (SODA), 1995, pp. 457-466.*

[9] *Y. Frankel, Y. Tsiounis, and M. Yung. "Indirect Disclosure Proof ": Achieving Effcient Fair Off-Line E-Cash. In Asiacrypt '96, LNCS 1163, pages 286--300. Springer-Verlag, 1996.*

[10] *J. Baek, "A design of a protocol for detecting a mobile agent clone and its correctness proof using Coloured Petri Nets." Technical Report TR-DIC-CSL-1998-002, Info. & Comm., K-JIST, (1998).*

[11] *W. Jansen and T. Karygainnis, "NIST Special Publication 800-19 -- Mobile Agent Security." (1999).*

[12] *W. M. Farmer, J. D. Guttman, and V. Swarup, "Security for mobile agents: Issues and requirements." Proc. of the 19th National Information Systems Security Conf. (1996) 591 – 597.*

[13] *W. A. Jansen, "Countermeasures for Mobile Agent Security." (1999)*

[14] *Okamoto, T., and Ohta, K., "Universal Electronic Cash", Proceedings of Crypto 91, pp. 324-337 (1992)*

[15] *T. Nakanishi and Y. Sugiyama. Unlinkable divisible electronic cash. In Information Security Workshop -- ISW 2000, volume 1975 of Lecture Notes in Computer Science, pages 121--134. Springer Verlag, 2000.*

[16] *Hans van Antwerpen. Electronic cash. Master's thesis, CWI, 1990.*

[17] *D. M. Chess, "Security Issues in Mobile Code Systems." In Mobile Agents and Security. LNCS 1419 (1998) 1-14.*