# Vulnerabilities in the Adachi-Aoki-Komano-Ohta Micropayment Scheme

Minho Kim[1] and Çetin Kaya Koç[2]
*(Corresponding author: Minho Kim)*

Information Security Laboratory, School of EECS, Oregon State University[1]
Corvallis, Oregon 97331, USA (Email: mhkim@eecs.oregonstate.edu)
Information Security Research Center, Istanbul Commerce University[2]
Eminönü, Istanbul 34112, Turkey (Email: koc@cryptocode.net)

*(Received Dec. 10, 2005; revised and accepted Dec. 31, 2005)*

## Abstract

Rivest and Shamir presented two simple micropayment schemes, "PayWord" and "MicroMint," for making small purchases over the Internet [14]. Recently, Adachi et al. have pointed out that the PayWord scheme has two security problems, and proposed a new micropayment scheme to overcome these problems [1]. Nevertheless, we show that their protocol is still vulnerable to impersonation and replay attacks.

*Keywords: Electronic commerce, impersonation, micropayment, payword, replay attack*

## 1 Introduction

There exist several payment protocols for electronic commerce [3, 7, 8, 9, 11, 12, 15]. Most currently-used protocols for Internet e-commerce are based on credit card charging over Secure Sockets Layer (SSL). Such schemes require the merchant to perform an online credit check using a process hidden from the user. The cost of a check is around 10 cents, making it expensive for low-value transactions. A secure and user-friendly solution for micropayments can be built to provide a cost-efficient and effective infrastructure for creating a network of interoperable payment service providers. Such solution offers the following: 1) facilitation of independent operators of the system to interoperate in order to create a payment network, enabling the operators to reach out rapidly to a critical mass of consumers and merchants, 2) multi-channel access to different devices, 3) an open and extendible platform for developing multiple payment applications, 4) lower operational costs and automated dispute resolution. Generally, micropayment systems collect the accumulated amount of money as one regular payment either before or after the transactions. It is well suited to be used as a charging mechanism for public transportation systems, access control to sites and services, selling contents (music, video, software, etc.) subscriptions, and "pay-per-view or -click" Web services for small amounts of money called "microcents". Since it is not practical for individual users to charge small amounts of money, such as a penny or a fraction of a penny, to a major charge card, a different method of payment is needed for sites that wish to go "micro".

Although many micropayment systems already exist, none has obtained a dominant market position. Customers and providers are faced with the situation where they have to install and use multiple systems, which is not a desirable situation. Several methods of micropayment collection are being examined, many of which involve the encoding of per-fee-links inside HTML pages and an Internet wallet account where individuals would establish a cash balance with a third-party application that would monitor, collect, and distribute micropayments. A central requirement for any electronic payment system is that a compromise or failure should not have tragic consequences. For example, it should not be possible to double spend in a digital cash system, nor should the compromise of a client's authorization secret entail unlimited client liability or uncollectible transactions. Traditional payment systems are designed to prevent such failures.

NetBill [4] is an on-line transactional payment protocol with many advanced features (atomicity, group membership, etc.) that requires communication with the NetBill server for each transaction. However, this protocol also has the same flaw of double spending. Another scheme proposed in [6] is unappealing for micropayments for these same reason.

Millicent [10] and NetCents [13] are scrip-based off-line-friendly micropayment protocols. The trust model in Millicent defines three roles: vendors, customers, and brokers. Brokers act as intermediaries between vendors and customers. A customer enters into a long-term relationship to buy digital money or scrip from brokers, who are assumed to be large entities such as banks or credit card

issuers. Brokers are most trusted in this model; customers are least trusted. Since the monetary unit used in these protocols is vendor-specific, double-spending is made to be very difficult. The assumption behind both protocols is that people tend to re-use the same merchants repeatedly. If this assumption holds, the interactions between the customer and the bank are kept at a minimum. A hidden assumption is that merchants have total information over their sales, so double spending with the same merchant is detectable. However, this scheme does not realize anonymity, because the bank purchases a scrip by himself from the server on behalf of the client.

The WebMoney [16] schemes realize the anonymity of the customers and the divisibility of the coins. However, it has a flaw. The bank can deceive the client by rewriting the sum of cash. And it has some inconveniences. It needs the high commission fee from the bank to the server, and the client inputs 16 decimal digits prepaid number for every purchase. Moreover, it still has an on-line check for detecting double spending.

Digital cash-based systems [2, 5] provide attractive features such as anonymity, inherent off-line operation, identity revealing on double spending and the Chaum's blind signatures. However, [2] does not directly address the issue of double spending, and [5] does not fulfill the divisibility of coins and needs an on-line check for detecting double spending that increases the computation cost for every purchase. The same drawback is apparent in the micropayment protocols, such as PayWord [14]. While the double-spending possibility is an inherent property of all such systems, none of the above protocols employ any kind of risk management scheme to address it.

Rivest and Shamir introduced two simple micropayment schemes, "PayWord" and "MicroMint" [14]. Their goal was to minimize the number of public-key operations required per payment using hash operations whenever possible. Generally, there are two positions of the bank with regard to the certificate. In Position 1, the bank takes full responsibility for the certificate and compensates all payments created by the customer's purchases. In Position 2, the bank does not redeem payments exceeding a limit set for the customer and shares the loss with the shop if trouble occurs.

Recently, Adachi et al. have pointed out that the PayWord scheme has two security problems [1]. A malicious customer can incure damages to the bank by purchasing in excess of the customer's credit. In general, a bank guarantees the customer's credit by issuing a certificate. The shop accepts the customer and initiates the transaction after checking the validity of the certificate that is kept by the customer. Because the certificate is not amended by the shop, malicious customers can use the same certificate at another shop and exceed their true credit level. In the PayWord scheme, the bank could reduce its risk by adopting Position 2 rather than Position 1. However, the bank can damage the shop in Position 2 by impersonating an imaginary customer and making the shop share the loss with the bank. Adachi et al. introduced two attacks: 1) customer certificate abuse attack and 2) bank falsification attack. They then proposed a new micropayment scheme that demands one on-line communication connection between the bank and the shop at the beginning of each transaction between the customer and the shop. Unfortunately, we find that their protocol still suffers from impersonation and replay attacks.

# 2 Review of the Adachi et al's Scheme

This scheme was described in [1]. We first introduce the notation used to describe the protocols then briefly show their scheme.

## 2.1 Notations

- $I_B/I_C/I_S$ denote bank $B$/ customer $C$/ shop $S's$ ID.
- $PK_B/PK_C$ denote the public keys of $B$ and $C$.
- $SK_B/SK_C$ denote the secret keys of $B$ and $C$.
- $\{M\}_{SK_B}/\{M\}_{SK_C}$ denote a message $M$ with its digital signature that was generated by $B/C's$ secret key.
- $E$ denotes the expiration date of $M$.
- $I$ denotes any additional information.
- $r$ denotes a random nonce that was selected by $S$.
- $h(\cdot)$ denotes a one-way and collision-resistant hash function and $h(m_1, m_2)$ means the hash of the concatenation of the message $m_1$ and $m_2$.
- The expression $A \longrightarrow B \ : \ X$ means $A$ sends the message $X$ to $B$.

## 2.2 The Adachi et al's Scheme

In the PayWord Scheme [14], the bank notices that a customer certificate abuse attack has occurred when it gets the hash coins with certificates from each shop used by the client. At this time, the shop has already sent goods or provided services to the client since it trusted the certificate. The bank has to cover the loss of the shop. To solve this problem, the bank withdraws the money corresponding to the length of the hash chain from the client's account and pools it in advance when the client starts a transaction with the shop. Moreover, the bank guarantees the validity of the customer's public key and can impersonate an imaginary customer as an avenue of an attack. To avoid this possibility, the validity of this key is guaranteed not by the bank but by the CA. In the Adachi et al's scheme, instead of the shop, the bank verifies the commitment $M$ and guarantees its validity for the client's certificate. The shop can then check the veracity of the commitment and the client's credit at the

same time. Hence, their scheme can reduce the verification cost of the commitment without degrading security. The procedures of the Adachi et al's scheme are listed as follows (Figure 1):

P1. $C$ requests $B$ to establish a bank account.

P2. $B$ makes $C's$ bank account and replies to $C$.

P3. $C \longrightarrow S : M = \{I_S, w_0, n, E\}_{SK_C}$

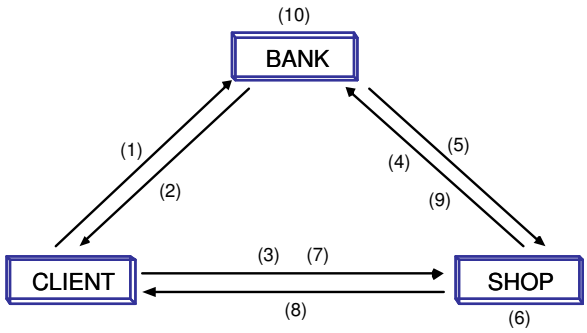


Figure 1: The Adachi et al's scheme

$C$ produces the value $w_n$ at random and computes a hash chain, $w_n \xrightarrow{h} w_{n-1} \xrightarrow{h} \cdots w_1 \xrightarrow{h} w_0$. $C$ can select the length $n$ of the hash chain. Next, $C$ calculates $M$ and sends it to $S$.

P4. $S \longrightarrow B : I_C, M$, and $r$

$S$ sends $I_C, M$, and $r$ to $B$ and requests a check for the validity of $C's$ credit.

P5. $B \longrightarrow S : C_C = \{I_C, M, Yes, r, I\}_{SK_B}$

$B$ checks $M$ and $C's$ credit. If they are valid, $B$ withdraws the money corresponding to the hash length $n$ from $C's$ account and keeps it as a pooled value in its database(DB). Next, $B$ computes a certificate $C_C$ and sends it to $S$.

P6. $S$ checks $C_C$ using $PK_B$ to confirm $C's$ credit and $M's$ correctness.

P7. $C \longrightarrow S : h(w_i, i)$

$C$ sends an order and the hash value $h(w_i, i)$ for the payment to $S$, where $i$ is the index of the hash value.

P8. $S \longrightarrow C :$ Goods or services

$S$ verifies $w_{i-1} = h(w_i)$ and confirms the validity of the payment. If $S$ verifies all transactions, he sends goods or provides services required to $C$.

P9. $S \longrightarrow B : h(w_i, i)$

$S$ sends the hash value $h(w_i, i)$ for the payment to $B$. In this step, $S$ may send only the latest coin $h(w_k, k)$ received from $C$ for the payment.

P10. $B$ verifies $C_C$ and $h(w_k, k)$ received from $S$ using the information stored in $B's$ DB. If they are valid, then $B$ puts money into $S's$ bank account from the pooled value of $C$ in $B's$ DB. Next, $B$ updates and stores only the latest coin in $B's$ DB.

# 3 Our Attacks

In Adachi et al's proposed scheme, we suppose that the adversary $A$ monitors their communications. Our attack is briefly summarized below:

A1. $C \longrightarrow A : M = \{I_S, w_0, n, E\}_{SK_C}$

$A$ intercepts the message $M = \{I_S, w_0, n, E\}_{SK_C}$ in step P3. Next, $A$ decrypts $M$ with $PK_C$ and obtains $I_S, w_0, n$, and $E$.

A2. $A \longrightarrow B : I_C, M$, and $r_A$

$A$ pretends to be $S$ and sends ID of $C$, $M$, and his own random number $r_A$ to $B$. Next, he requests a check for the validity of $C's$ credit.

A3. $B \longrightarrow A : C_C = \{I_C, M, Yes, r_A, I\}_{SK_B}$

$B$ checks $M$ and $C's$ credit. If they are valid, $B$ withdraws the money corresponding to the hash length $n$ from $C's$ account and keeps it as a pooled value in its database(DB). Next, $B$ computes a certificate $C_C$ and sends it to $A$. $A$ decrypts $C_C$ with $PK_B$ and obtains $I_C, M, Yes, r_A$, and $I$.

A4. $C \longrightarrow A : h(w_i, i)$

$C$ sends an order and the hash value $h(w_i, i)$ for the payment to $S$. $A$ intercepts this hash value $h(w_i, i)$.

A5. $A \longrightarrow B : h(w_i, i)$

$A$ sends the hash value $h(w_i, i)$ for the payment to $B$ and ask to put the money $A's$ account. In this step, $A$ may send only the latest coin $h(w_k, k)$ received from $C$ for the payment.

A6. $B$ verifies $C_C$ and $h(w_k, k)$ received from $A$ using the information stored in $B's$ DB. If they are valid, then $B$ puts money into $A's$ bank account from the pooled value of $C$ in $B's$ DB. Next, $B$ updates and stores only the latest coin in $B's$ DB.

**Replay attack** is the offensive action that impersonates or deceives another legitimate participant through the reuse of information obtained in a protocol. It indicates an attempt by an unauthorized third party to record the exchanged messages.

**Impersonation attack** deceives the identity of one of the legitimate parties. The attacker inserts or changes a message and claims it coming from a real sender.

When $A$ intercepts the message in step P3, obtains $M = \{I_S, w_0, n, E\}_{SK_C}$ and decrypts with $PK_C$, he can get $I_S, w_0, n$ and $E$. Next, $A$ pretends to be $S$ and replays $I_C$, $M$, and his own random number $r_A$ to $B$ in step A2. $B$ checks $M$ and $C's$ credit. Since they are valid, $B$ withdraws the money corresponding to the hash length $n$ from $C's$ account and keeps it as a pooled value in his database. Next, $B$ computes a certificate $C_C$ and sends it to $A$ in step A3. $A$ decrypts $C_C$ with $PK_B$ and obtains $I_C, M, Yes, r_A$, and $I$. Here, $A$ has more information from $I$. $C$ sends an order and the hash value $h(w_i, i)$ for the payment to $S$. $A$ intercepts this hash value $h(w_i, i)$. $A$ can replay again by sending the hash value $h(w_i, i)$ for the payment to $B$. In this step, $A$ may send only the latest coin $h(w_k, k)$ received from $C$ for the payment. After $B$ verifies $C_C$ and $h(w_k, k)$, $B$ puts money into $A's$ bank account from the pooled value of $C$ in $B's$ DB. In addition, in order to dig a pit for the third party, $A$ would instead impersonate the third party and cause the bank to deposit money into the third party's account. This is achieved by sending information of which account the bank need to deposit to in step A6.

Even though $A$ did not know anything before step P7, given that he can obtain the hash value $h(w_i, i)$ in step P7, he can attack successfully. By sending $h(w_i, i)$ and asking to put the money into $A's$ account, $B$ will verify $C_C$ and $h(w_k, k)$. If they are valid, $B$ puts money into $A's$ bank account from the pooled value of $C$ in $B's$ DB.

After step A3, $A$ obtains any additional information from $I$. In addition, since he gets information $h(w_i, i)$ in step A4, he can compute all hash values. The attacker can continue to impersonate the customer and the shop. So he can attack again until consume the whole money which he can spend before the end of expiration date $E$.

Thus, $A$ can attack successfully by replaying the original messages and modifying them to impersonate another party. Therefore, their proposed scheme is still vulnerable against replay and impersonation attacks.

# 4 Conclusions

In this paper, we have shown that the solutions to the security problems found in Rivest and Shamir's PayWord scheme proposed in [1] are still vulnerable to impersonation and replay attacks.

# References


[1] N. Adachi, S. Aoki, Y. Komano, and K. Ohta, "Solutions to security problems of Rivest and Shamir's PayWord scheme," *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, vol. E88-A, no. 1, pp. 195-202, Jan. 2005.

[2] M. Bellare, J. Garay, C. Jutla, and M. Yung, "VarietyCash: a multi-purpose electronic payment system," in *Proceedings of the Third USENIX Workshop on Electronic Commerce*, pp. 9-24, USENIX, Sep. 1998.

[3] O. Chun, M. K. Lars, and B. Jonathan, "A formal service specication for the Internet open trading protocol," in *Applications and Theory of Petri Nets 2002: 23rd International Conference, ICATPN 2002*, LNCS 2360, pp. 352-373, Springer-Verlag, 2002.

[4] B. Cox, D. Tygar, and M. Sirbu, "NetBill security and transaction protocol," in *Proceedings of the First USENIX Workshop on Electronic commerce*, pp. 77-88, USENIX, July 1995.

[5] J. B. Friis, "Digicash implementation," http://www.appliedcrypto.com, 2003.

[6] E. Foo and C. Boyd, "A payment scheme using vouchers," in *Proceedings of the Second International Conference on Financial Cryptography*, LNCS 1465, pp. 103-121, Springer-Verlag, 1998.

[7] Y. Kawatsura, *Secure Electronic Transaction (SET) Supplement for the v1.0 Internet Open Trading Protocol (IOTP)*, RFC 3538, June 2003.

[8] V. F. Kleist, "A Transaction cost model of electronic trust: Transactional return, incentives for network security and optimal risk in the digital economy," *Electronic Commerce Research*, vol. 4, no. 1-2, pp. 41-57, 2004.

[9] I. W. Lee, H. J. Park, and S. H. Kim, "Development of electronic commerce micropayment system with a pay-later payment method," *Communication Systems and Applications(CSA 2004)*, ACTA Press, pp. 161-165, 2004.

[10] M. S. Manasse, "The millicent protocols for electronic commerce," in *Proceedings of the First USENIX Workshop on Electronic Commerce*, pp. 117-123, USENIX, July 1995.

[11] B. Meng and H. Zhang, "An electronic commerce system prototype and its implementations," in *Proceedings of the 2005 The Fifth International Conference on Computer and Information Technology (CIT05)*, pp. 966-970, IEEE Computer Society Press, Sep. 2005.

[12] V. Patil and R. K. Shyamasundar, "An efficient, secure and delegable micro-payment system," in *International Conference on e-Technology, e-Commerce and e-Service (EEE- 04)*, pp. 394-404, IEEE Computer Society Press, Mar. 2004.

[13] T. Poutanen, H. Hinton, and M. Stumm, "NetCents: A lightweight protocol for secure micropayments," in *Proceedings of the Third USENIX Workshop on Electronic Commerce*, pp. 25-36, USENIX, Sep. 1998.

[14] R. L. Rivest and A. Shamir, "Payword and micromint : Two simple micropayment schemes," *Fourth Cambridge Workshop on Security Protocols*, Springer-Verlag, pp. 69-87, Apr. 1996.

[15] M. Strobel, "Design of roles and protocols for electronic negotiations," *Electronic Commerce Research*, vol. 1, no. 3, pp. 335-353, 2001.

[16] WebMoney Corporation, WebMoney for Internet shopping, http://www.webmoney.com, 2005.

**Minho Kim** is a Ph.D. student in the Department of Electrical Engineering and Computer Science at Oregon State University. He received B.S. degree in Computer Science from Korea Air Force Academy and M.S. degree in Computer Science from Yonsei University, Seoul, South Korea, in 1993 and in 1998. He has also worked as an assistant professor of Computer Science at Korea Air Force Academy. His research interests are in cryptography, computer and network security, and wireless communications.

**Çetin Kaya Koç** is currently a professor of Electrical Engineering and Computer Science at Oregon State University. He received his Ph.D. degree from University of California, Santa Barbara. Dr. Koç's research interests are in cryptographic engineering, algorithms and architectures for cryptography, computer arithmetic and finite fields, parallel algebraic computation, and network security. He has founded the Workshop on Cryptographic Hardware and Embedded Systems (CHES), and has been an Associate Editor of IEEE Transactions on Computers and IEEE Transactions on Mobile Computing. Dr. Koç has also been working as a consulting engineer with research and development interests in cryptographic engineering and embedded systems for several companies including Intel, RSA Security, and Samsung Electronics. Dr. Koç is currently on leave from Oregon State University, working at Information Security Research Center of Istanbul Commerce University in Istanbul, Turkey.