전자상거래 촉진을 위한 공유키 기반 신용카드 조회 시스템

장 시 웅[†] · 신 병 철[††] · 김 양 곡[†††]

요 약

본 논문에서는 전자상거래 시스템에서의 보안 문제를 해결하기 위해 신용카드 조회 시스템을 설계하고 구현하였다. 전자 상거래시 PC에서 신용카드 조회기를 사용하면 키보드 입력 없이 신용카드 조회기에서 신용카드를 읽어 신용카드 결재를 수행한다. 새로운 신용카드 조회 시스템은 신용카드 조회기 내부의 칩에서 공유키 기반으로 신용카드 정보를 즉시 암호화하여 호스트 시스템에 보냄으로써 키보드 해킹 위험에서 안전하다는 장점이 있다. 신용카드 조회 시스템의 암호화/복호화를 위해 quotient ring에 기반한 행렬 연산을 사용하였으며, 암호화의 안전성을 위해 모든 암호 대상 데이터에 대해 서로 다른 암호 행렬을 생성하는 방법을 제시하고, 서로 다른 암호 행렬를 구성하기 위해 요구되는 암호키의 크기 및 행렬의 크기를 산출하였다. 신용카드 결재를 위하여는 소량(0.1KB)의 데이터가 요구되므로, 암호키의 크기가 128bits만 되어도 역행렬을 고려한 2×2 행렬의 경우 좋은 성능을 보이는 것으로 분석되었다. 신용카드 조회 시스템을 인증용으로 사용하기 위하여는 0.5KB 이상의 데이터가 필요하므로, 암호키의 크기가 256bits이상에서 2×2 행렬이나 3×3 행렬을 사용하면서 역행렬을 고려하는 것이 좋은 것으로 분석되었다.

A Credit Card Sensing System based on Shared Key for Promoting Electronic Commerce

Si-Woong Jang[†] · Byoung-Chul Shin[††] · Yangkok Kim[†††]

ABSTRACT

In this paper, the magnetic sensing system is designed and implemented for the safe security in internet commerce system. When the payment is required in the internet commerce system, the magnetic sensing system will get the information from a credit card without keyboard input and then encrypt and transmit the information to server. The credit card sensing system, which is proposed in this paper, is safe from keyboard hacking because it encrypts card information immediately in its internal chip and sends the information to host system. For the protection of information, the magnetic sensing system is basically based on a synchronous stream cipher cryptosystem which is related to a group of matrices. The size of matrices and the bits of keys for the best performances are determined for various cases. It is shown that for credit card payments, matrices of size 2 have good performance even at most 128bits keys with the consideration of inverse matrices. For authentication of general-purpose data, the magnetic sensing system needs more than 0.5KB data and in this case, the optimum size of matrices is 2 or 3 at more 256bits keys with consideration of inverse matrices.

키워드 : 전자지불(Electronic Payment), 신용카드(Credit Card), 인터넷(Internet)

1. Introduction

Nowadays, expansion of Internet business is rapidly activating the electronic commerce. Electronic commerce may be defined as the entire set of processes that support commercial activities on a network and involve information analysis [1]. The objectives of electronic commerce are manifold. They involve increasing the speed and efficiency of business transactions and processes, and improving customer service. Moreover, participants should be empowered to head into unexplored directions including enhanced competitiveness, job creation, and economic growth. Three distinct classes of electronic commerce applications can be distinguished : customer-to-business, business-to-business and intra-organization [2]. Our system is one of the customer-to-business applications.

Electronic commerce systems require method to pay for

※ 본 결과물은 과학기술부 · 한국과학재단에서 지정한 지역협력연구센터(RRC) 및 산업자원부 · 한국산업기술평가원에서 지정한 지역기술혁신센터(TIC)인 동의대학교 전자세라믹스센터의 지원(2003학년도 동의대학교 자체학술연구조성비 포함)을 받았습니다.

† 종신회원 : 동의대학교 컴퓨터과학과 교수
†† 정 회 원 : 동의대학교 나노공학부 교수
††† 정 회 원 : 동의대학교 수학 · 정보통계학부 교수

논문접수 : 2002년 7월 24일, 심사완료 : 2003년 6월 16일

the delivery of data, products, and services. Electronic payments include digital currencies (smart cards and electronic money), electronic checks, and credit card payments [3]. Credit card systems may be impractical because the cost of processing the transaction is greater than the actual chare [4]. However, credit card payment systems are well suited for hard good sales. Credit card payments are familiar with most persons. Credit card purchases involve the transmission of encrypted data through computer networks. This includes entering credit card information (card number and expiration date) into the system, having it encrypted by the system, and sending the information over the network.

The problem with credit card payment system is their non-refutability, which requires the physical presence of the credit card owner for authentication of the signature [5]. Other problems associated with such payment methods are privacy, speed of transaction, and safety [6, 7]. When a consumer buys some things over the network, he can enter other's credit card number and expiration date using keyboard. He can easily steal other's card number and expiration date by looking at the card or the card sales slips, or by hacking the network. And, if a consumer inputs his credit card number by typing keyboard on the Internet, the information of credit card can be hacked by the key board hacking, the spoofing, and etc.

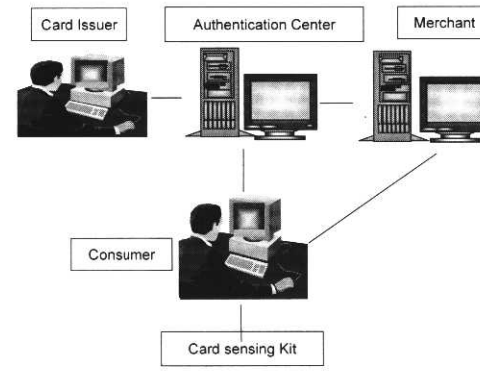
It is desirable that only card owner can pay over the network. In this research, a new card sensing system has been assembled and tested. When this card sensing system reads the card information, it encrypts the data by the keys that are received from the main server of an authentication center and generated in the kit itself. Because the inputted data by using internet card reader is encrypted in the card reader itself, the data are safe from keyboard hacking and spoofing on the Internet. In this paper, we propose a new magnetic card sensing system based on Internet card reader for the secure commerce on the Internet.

2. Magnetic card sensing system on the Internet

In this section, we describe concept of magnetic card sensing system on the internet and architecture of magnetic card sensing kit.

2.1 Concept of new magnetic card sensing system

(Figure 1) shows a conceptual approach for using credit cards to purchase goods and services on the Internet. The consumer accesses a merchant's home page and receives a display of the merchant's goods. The consumer selects desired goods and offers a credit card payment to the merchant.



(Figure 1) Credit Card payment system on the Internet

The merchant server accesses the server of an authentication center. The server generates an encryption key and sends it to the card sensing kit. The sensing kit read the consumer's credit card. And a microchip in the kit encrypts the card data by two encryption keys that are received from the authentication center and generated in the kit itself. The sensing kit sends the encrypted data to the authentication center. The authentication center decrypts the encrypted data and encrypts it again by the different key agreed between the authentication center and card issuer. The center sends the secondly encrypted data to the card issuer.

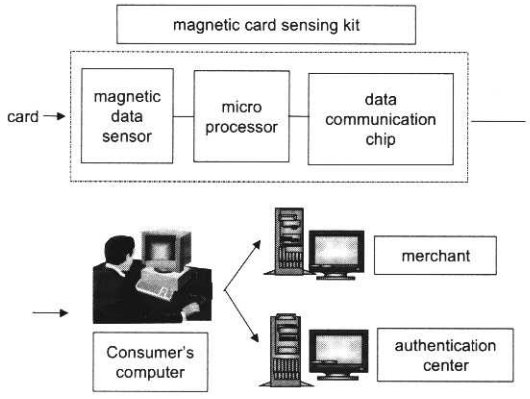
The card issuer informs the authentication center and the merchant whether to proceed with the purchase. The merchant informs the consumer whether the transaction has been completed. The merchant sends the receipt to the consumer and the merchant's bank. The consumer may receive the account updates once a month through postal mail.

2.2 Magnetic card sensing kit

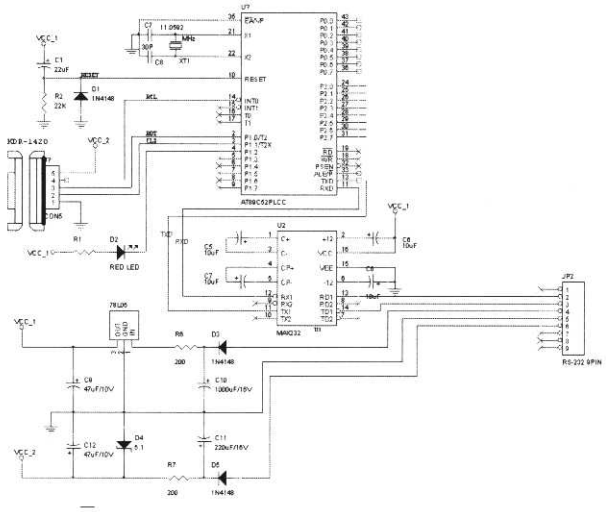
The magnetic card sensing kit is composed of a magnetic data sensor (KDR-1420, KDE Co.), a microprocessor (AT 89C52PLCC) and a data communication chip (MAX232) as shown in (Figure 2).

The magnetic data sensor (KDR-1420, KDE Co.) reads the information of the magnetic card. The microprocessor (AT89C52PLCC) encrypts the information by the keys that are received from the authentication center and generated in the microprocessor itself. After encryption, the data communication chip (MAX232) sends the encrypted data

to the authentication center. (Figure 3) shows the circuit of the magnetic card sensing kit.



(Figure 2) Block diagram of the magnetic card sensing Kit



(Figure 3) Circuit of the magnetic card sensing kit

3. Encryption and Decryption of the card sensing kit

The information on a credit card obtained by the card reader will be transmitted to the server through an insecure channel, Internet. Hence a cryptosystem for protecting the information from illegal and uninvited user should be introduced. Indeed the information, which is obtained from a credit card, can be encoded in various crypto algorithms and then can be transmitted. Basically we rely on a synchronous stream ciphers, and a keystream for a stream ciphers is generated by two factors, one from the host and the other from the card reader.

3.1 Overview of an encryption algorithm

We recall algebraic structures "group" and "ring", which are nonempty sets with one and two binary operations respectively[8, 9]. We usually take addition and multiplication, denoted by + and ×, for two binary operations for a ring. We now introduce a ring Z_n and a group $GL_{m\times m}(Z_n)$ where Z_n is a quotient ring of the integer ring Z by its ideal nZ with usual addition and multiplication and $GL_{m\times m}(Z_n)$ is a group of all invertible $m\times m$ matrices over Z_n with usual matrix product. In the sequel, we simply say the quotient ring with base n for Z_n. We note that addition and multiplication will be done under module. We now recall the formal definition of a synchronous stream ciphers[8, 9] as follows.

A Synchronous Stream Cipher is a 7-tuple (P, C, K, L, F, E, D), where the following conditions are satisfied.

① P is a finite set of possible plaintexts.
② C is a finite set of possible ciphertexts
③ K, the keyspace, is a finite set of possible keys.
④ L is a finite set called the keystream alphabet.
⑤ $F = (f_1, f_2, \cdots)$ is the keystream generator. For each i, $f_i : K \rightarrow L$.
⑥ For each z in L, there is an encryption rule e_z in E and a corresponding decryption rule d_z in D. $e_z : P \rightarrow C$ and $d_z : C \rightarrow P$ are functions such that $d_z(e_z(x)) = x$ for every plaintext x in P.

We basically use a synchronous stream cipher for encryptions and decryptions of information from a credit card. Its keystream $e_1, e_2, \cdots$ for a stream cipher consists of matrices $A_1, A_2, \cdots$ in $GL_{m\times m}(Z_n)$. The host will fix $M_1, M_2, \cdots$ in $GL_{m\times m}(Z_n)$ for a keystream. Then the card reader will partition a plaintext into several blocks and produce a keystream with suitable length by taking some out of $M_1, M_2, \cdots$ and multiplying the selected matrices in certain order. We need to consider the order of product because the product of matrices is not commutative. Finally each block of a plaintext will be encoded by a keystream independently. We note that a card reader will perform all these procedure on the basis that each block of a plaintext will be encoded with complete independence of encoding for other blocks.

For an ordered subset $S = \{M_1, M_2, \cdots, M_q\}$ of $GL_{m\times m}(Z_n)$, the lexicographic product $L(r)$ of length $r \geq 1$ of S is a finite arrangement $P_1, P_2, \cdots, P_{q^r}$ with a lexicographic order where

$P_i = M_{i(1)} M_{i(2)} \cdots M_{i(r)}$ and

$M_{i(1)}\ M_{i(2)} \cdots M_{i(r)}$ precedes $M_{j(1)}\ M_{j(2)} \cdots M_{j(r)}$ if

(i) $i(1) < j(1)$ or

(ii) $i(h) = j(h)$ for all $h \leq k \leq r$ and $i(k) < j(k)$.

If we give an order $L(i) < L(j)$ for $i < j$, then we can produce an n elements ordered set L_n for all n. Indeed the set L_n consists of the first n elements of a sequence $L(1)$, $L(2)$, ….

We now summarize an encryption algorithm in the card reader as follows.

Step 1 : Get the number b, m and n for the base of a quotient ring, the size of the matrices and the number of matrices from the host respectively.

Step 2 : Order the set S of n matrices from the server.

Step 3 : Get a k elements ordered set L by the above method.

Step 4 : Generate a permutation $f(i)$ on L.

Step 5 : Get a keystream $e_i = f(i)$ for $i = 1, 2, \cdots, k$.

We need to recall that an $m \times m$ matrix A lies in $GL_{m \times m}(Z_b)$ if and only if it is invertible. Equivalently $\det(A)$, the determinant of a matrix A, is relatively prime to b. As we see the above algorithm, the card reader get b, m and n from the host. Then a card reader will partition a plaintext into several blocks of the same length m. If the length of a plaintext is not the multiple of m, we can add additional digit numbers (Step 1). For each block, a card reader will produce an encoding key by using n invertible matrices from the host. We note that the number k in Step 3 will be determined according to the length of a plaintext. A card reader generates a keystream by using the information from the host and the random number generated by itself. For this purpose, the card reader first orders n invertible matrices from the host numbers, and then it will select a certain number of matrices out of n matrices (Step 2, 3). By taking the products of selected matrices with a fixed order (Step 4), it will produce an encoding matrix e_1 for the first block (Step 5). Then the matrix e_1 will encode the first block of a plaintext. Similarly the next block will be encoded by a key e_2. We continue this procedure until we finish all the blocks of a plaintext.

3.2 Organization of encryption keys on the host system

We now introduce a method for the packing of encryption keys. For our simplicity we restrict our attention to the size of encryption keys by 128bits ~ 1024 bits. It can be naturally extended to the general case. The ingredients for encryption keys basically consist of three components, the base b for a quotient ring, m for the size of matrix and n for the number of matrices. Encryption keys are operated as shared keys between a card reader and a host system.

b	m	n	The components of matrices

(Figure 4) Format of an encryption key

We need to make the base of a quotient ring bigger than 91 because it contains all ASCII values for the alphabets and numbers. Hence we allocate 8bits for the value b and take an interval [91, 255] for a domain for b. The number of operations related to matrices is sharply increasing according to increase of the size of the matrices. Hence we only consider the size of matrix up to 4 and allocate 8bits for the size of a matrix. Similarly the number of matrices will be bounded by 128 and 8bits will be for the number of matrices. We need 24bits for b, n and m. Moreover we insist that the bits for components of an encryption matrix be bigger than or equal to 4 in order to have various values in some extents. Then if the total bits for a keystream is T_b, then the bits T_m for components of an encryption matrix satisfies the following condition :

$$T_m = \left\lfloor \frac{T_b - 24}{m^2 n} \right\rfloor \geq 4 \qquad (1)$$

When we allocate 128~1024bits for an encryption key, the condition equation (1) gives the cases of the organization for encryption keys shown in <Table 1>. As we see in <Table 1>, we can produce 6 2×2 matrices, 3 3×3 matrices and 1 4×4 matrix by 128bits for encryption keys.

<Table 1> Organization of Encryption matrices

Size of Keys	Minimum number of matrices			Maximum number of matrices		
	m	n	T_b	m	n	T_b
128bits	2	1	28	2	6	4
	3	1	12	3	3	4
	4	1	7	4	1	7
256bits	2	1	60	2	15	4
	3	1	26	3	6	4
	4	1	15	4	3	5
512bits	2	1	124	2	31	4
	3	1	55	3	13	4
	4	1	31	4	7	4
1024bits	2	1	252	2	63	4
	3	1	112	3	28	4
	4	1	63	4	15	4

For an example, we let the base for a quotient ring 102, the size of the matrices $m=2$, the number of matrices $n=4$ and the bits of a keystream 128. We suppose that a plaintext is 1 2 3 4 5 6 7 8 1 2 3 4. Then we need to consider 6 blocks, (1 2), (3 4), (5 6), (7 8), (1 2), (3 4) and we allocate 9bits for each matrix by the inequality equation (1). Thus all components of a matrix have values between 0 and 127. Let

$$A = \begin{pmatrix} 34 & 100 \\ 5 & 16 \end{pmatrix},\ B = \begin{pmatrix} 79 & 108 \\ 125 & 13 \end{pmatrix},$$

$$C = \begin{pmatrix} 88 & 78 \\ 77 & 11 \end{pmatrix} \text{ and } D = \begin{pmatrix} 101 & 74 \\ 14 & 18 \end{pmatrix}.$$

Then we have lexicographic products

$$L(1) = \{A, B, C, D\}$$
$$= \left\{\begin{pmatrix} 34 & 100 \\ 5 & 16 \end{pmatrix}, \begin{pmatrix} 79 & 108 \\ 125 & 13 \end{pmatrix}, \begin{pmatrix} 88 & 78 \\ 77 & 11 \end{pmatrix}, \begin{pmatrix} 101 & 74 \\ 14 & 18 \end{pmatrix}\right\},$$

$$L(2) = \{AA, AB, AC, \cdots, DD\}$$
$$= \left\{\begin{pmatrix} 24 & 2 \\ 46 & 42 \end{pmatrix}, \begin{pmatrix} 90 & 76 \\ 49 & 34 \end{pmatrix}, \begin{pmatrix} 84 & 80 \\ 40 & 56 \end{pmatrix}, \cdots, \begin{pmatrix} 17 & 34 \\ 34 & 34 \end{pmatrix}\right\}\cdots,$$

and so a lexicographic ordering

$$\begin{pmatrix} 34 & 100 \\ 5 & 16 \end{pmatrix}, \begin{pmatrix} 79 & 108 \\ 23 & 13 \end{pmatrix}, \begin{pmatrix} 88 & 78 \\ 77 & 11 \end{pmatrix}, \begin{pmatrix} 101 & 74 \\ 14 & 18 \end{pmatrix}, \begin{pmatrix} 24 & 2 \\ 46 & 42 \end{pmatrix}\cdots$$

Hence A, B, C, D, AA, AB will be 6 matrices for a keystream. We need a permutation of $\{A, B, C, D, AA, AB\}$. Let AA, B, C, AB, D, A be a new ordering. Then AA, B, C, AB, D, A will compose a keystream and so $e_1 = AA$. The first matrix AA will be selected for a key of the first block (1, 2) and so by simple matrix multiplications :

$$AA\begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 24 & 2 \\ 46 & 42 \end{pmatrix}\begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 28 \\ 28 \end{pmatrix}.$$

This means that the first block (1, 2) will be encoded to (28, 28). By the same way we can encode the second block by B. We continue this procedure until we finish all blocks of a plaintext. Then the host easily will decode a ciphertext and get a plaintext by the reverse process.

3.3 Complexity of Matrix Products

When we use only one matrix to produce 1000 matrices, the maximum length of matrix products is 1000. However if we take two matrices, the maximum length of matrix products is 500. In general as the number of matrices is increasing, then the maximum length of matrix products is sharply decreasing. We note that all matrices obtained by matrix products may not be different each other. Hence the matrices that compose a keystream need not be different. Suppose that L is the length of a plaintext, m is the size of matrices and n is the number of matrices. Then we need L/m elements for a keystream. Since the number of matrices with product length l is n^l, the maximum length of matrix products for a keystream without the consideration of inverse matrices is the least integer k such that

$$\frac{L}{m} \le n + n^2 + n^3 + \cdots + n^k.$$

Similarly when we consider inverse matrices, the maximum length of matrix products for a keystream is the least integer k such that

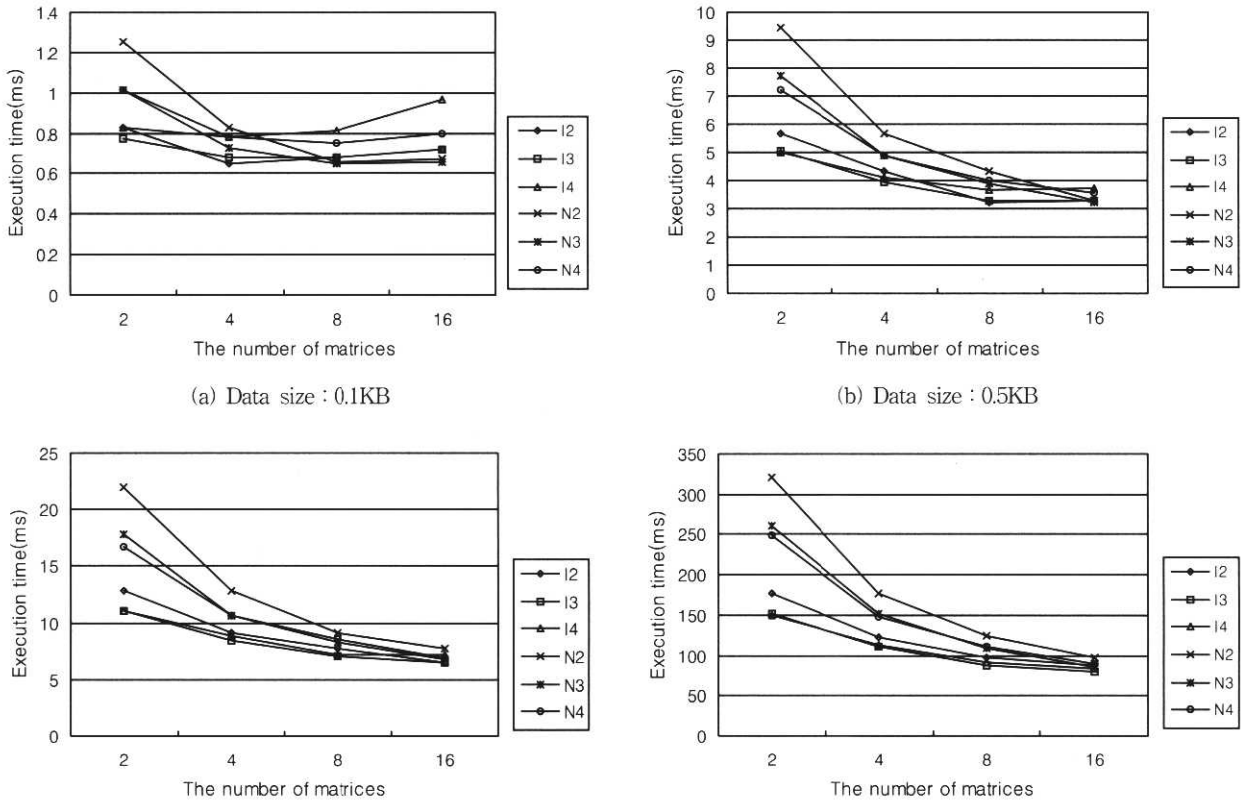
$$\frac{L}{m} \le 2n + (2n)^2 + \cdots + (2n)^k.$$

4. Results and Discussions

In this section we analyze the execution time of the encryption and decryption algorithm which based on the operations of matrices over a quotient ring in terms of the size of a plaintext, the number of matrices and the size of matrices. From experimental results, we conclude the size of matrices for the best efficiency for the encryption and decryption of a plaintext according to various cases.

4.1 The comparison of the execution times

The graphs in (Figure 5) show the comparisons of the total times required for the encryptions and decryptions in PC. As mentioned above, we produce all the matrices which will compose a keystream. Then we get an ordered set with a lexicographic order. Finally we need a permutation of the ordered set for a keystream. It is clear that the length of matrix products for a keystream is inverse proportional to the number of matrices. As the size of a plaintext is increasing, the maximum length of matrix products has more effect to the total execution time for the encoding and decoding time as we see in (Figure 5). When we consider inverse matrices in the production of a keystream, the maximum length reduces to the half of the one without the consideration of inverse matrices. That means, the total time for the consideration of inverse matrices relatively



(a) Data size : 0.1KB

(b) Data size : 0.5KB

(c) Data size : 1KB

(d) Data size : 10KB

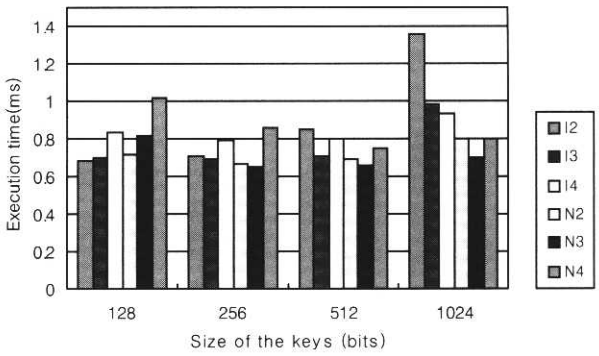
(Figure 5) Execution time according to data size and the number of matrices

short compared to the total time without the consideration of inverse matrices when other factors are fixed. However it takes time to calculate inverses of matrices. When the size of data is less than 0.1KB and the number of matrices is 4 on condition that inverse matrices are considered, it has the best performance in execution time(see (Figure 5) (a)). When the number of matrices is 8, it has worse performance in execution than when the number of matrices is less than 8. This comes from the reason : when the size of a plaintext is small, the increase of the number of matrices accompanies with the increase of the load for the calculation of inverse matrices. We note that the calculation for an inverse matrix of size 3 takes relatively longer time than the one of a matrix of size 2 but the size of data for encryption is reduced by 33%. However for a matrix of size 4 even if the operation for the calculation of the inverse matrix is sharply increasing, the size of data for encryption is reduced by merely 25%. From these information we can conclude 3 as the best size of matrices as shown in (Figure 5). When we have the size of a plaintext bigger than 1KB, the increase of the matrix numbers comes with the improvement of performance.

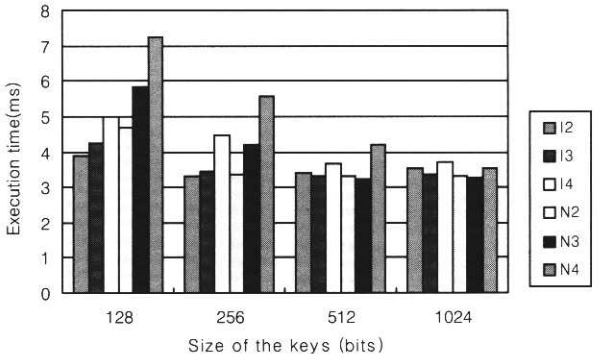
4.2 Optimum matrix size according to the size of a keystream

As we see in (Figure 6), it has best performance for matrices of size 2 in case that the size of keys is in 128~256bits on condition that inverse matrices are considered. That is due to facts that if matrix size is small, the number of matrices selectable is large with the small size of encryption key.

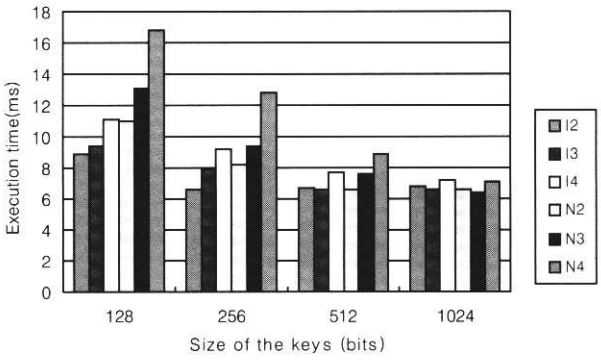
However when the size of a plaintext is small and the bits for keys is bigger than 256, we can select 3 as the size of matrices for the most effective execution. In this case we do not consider the inverse of matrices because of the overhead for the calculation of inverse matrices. This means that when the size of data for encryptions and decryptions is small, we had better not to consider inverse matrices for a keystream. Since a credit card has relatively small amounts of information, we can take 2 as the size of matrices with the consideration of inverse matrices provided that we allocate 128bits for a keystream. However the size of matrices need to be 3 without the consideration of inverse matrices when we allocate 256~1024bits for a keystream.



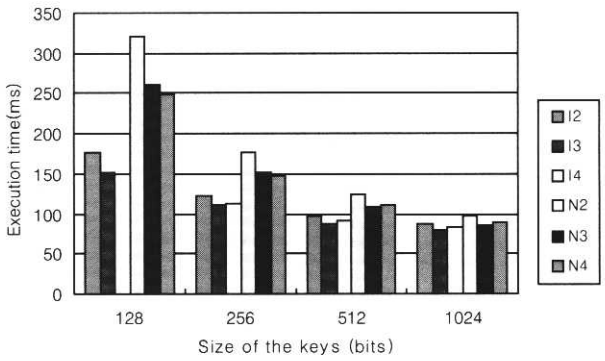
(a) Data size : 0.1KB



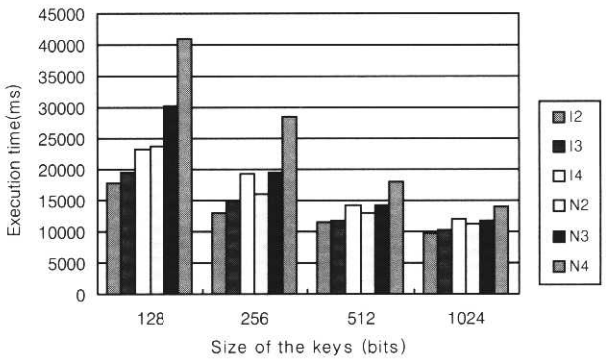
(b) Data size : 0.5KB



(c) Data size : 1KB



(d) Data size : 10KB



(e) Data size : 1MB

(Figure 6) Execution times according to the sizes of data and keys

For authentication of general-purpose data, we can take 2 or 3 as the size of matrices with the consideration of inverse matrices since the data more than 0.5KB is required. For data of more than 0.5KB, it has a good performance in execution time when we allocate at least 256bits for a keystream.

5. Conclusions

In this paper, we designed and implemented magnetic sensing system for the safe security in internet commerce system. When the payment is required in the internet commerce system, the magnetic sensing system will get the information from a credit card without keyboard input and then encode and transmit the information to server. Magnetic sensing system is based on a ring of matrices whose components lie in a quotient ring of the integer ring for its encryption and decryption keys. The given plaintext will be divided into several blocks according to the size of information on a credit card and then each block is encoded by its own encryption key.

Finally, it has a good performance at matrices of size 2

and 3 in most cases, while it has a poor performance at matrices of size 4. That is due to the sharply increased initial overhead for calculating inverse matrices in case of matrices of size 4. Since magnetic card sensing kit has small memory, it is desirable to encrypt and decrypt using matrices of size 2 with small overhead in matrix operations.

References

[1] Nabil R. Adam, Oktay Dogramaci, Aryya Gangopadhyah and Yelena Yesha, *Electronic Commerce*, Prentice Hall, pp.15-25, 1999.

[2] Ravi Kalakota and Andrew B. Whinston, *Frontiers of Electronic Commerce*, Addison-Wesley, pp.217-219, 1996.

[3] Simon Garfinkel and Gene Spafford, *Web Security & Commerce*, O'Reilly & Associates, Inc., pp.313-319, 1997.

[4] Andrew Dahl and Leslie Lesnick, *Internet Commerce*, New Riders Publishing, pp.70-81, 1996.

[5] Vijay Ahuja, *Secure Commerce on the Internet*, AP Professional, pp.170-174, 1997.

[6] 박현동, 이은성, 송상헌, 강신각, 박적수, 류재철, "안전한 인터넷 전자지불 프로토콜의 설계 및 구현", 정보처리논문지, 제6권 제8호, pp.2145-2156, 1999.

[7] 박현동, 강신각, 박성열, 류재철, "PGP를 이용한 WWW 기반에서의 전자지불 프로토콜 개발", 정보처리논문지, 제4권 제4호, pp.1046-1058, 1997.

[8] Douglas R. Stinson, *Cryptography : Theory and Practice*, CRC Press, pp.20-24, 1995.

[9] Schneier Bruce, *Applied cryptography : Protocols, Algorithms and Source code in C*, pp.197-206, 1996.

장 시 웅

e-mail : swjang@dongeui.ac.kr

1984년 부산대학교 계산통계학과(학사)

1993년 부산대학교 대학원 전자계산학과 (석사)

1996년 부산대학교 대학원 전자계산학과 (박사)

1986년~1993년 대우통신 종합연구소 주임연구원

1996년~현재 동의대학교 컴퓨터과학과 부교수

관심분야 : 전자지불, 전자상거래, 데이터베이스

신 병 철

e-mail : shinbc@dongeui.ac.kr

1984년 연세대학교 세라믹공학과(학사)

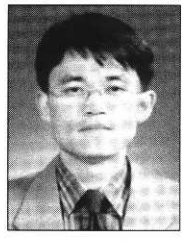
1986년 한국과학기술원 재료공학과(석사)

1988년 한국과학기술원 재료공학과(박사)

1988년~1998년 포철 RIST 신소재연구부문 책임연구원

1996년~현재 동의대학교 나노공학부 부교수

관심분야 : 전자세라믹스, 정보보안



김 양 곡

e-mail : ykkim@dongeui.ac.kr

1985년 부산대학교 수학과(학사)

1987년 부산대학교 수학과(석사)

1993년 캐나다 알버타대학교 수학과(박사)

1996년~현재 동의대학교 수학·정보통계학부 부교수

관심분야 : 순서를 가진 대수적 구조, 암호학