Abstract—A new public-key (two-key) cipher scheme is proposed in this paper. In our scheme, keys can be easily generated. In addition, both encryption and decryption procedures are simple. To encrypt a message, the sender needs to conduct a vector product of the message being sent and the enciphering key. On the other hand, the receiver can easily decrypt it by conducting several multiplication operations and modulus operations. For security analysis, we also examine some possible attacks on the presented scheme.

Index Terms—Public keys, private keys, cryptosystems, Diophantine equation problems, integer knapsack problems, one-way functions, trapdoor one-way functions, NP-complete.

I. INTRODUCTION

In [6], Diffie and Hellman proposed their pioneering idea of public key cryptosystems. In a public key system, each user \( U \) uses the encryption algorithm \( E(PK_u, M) \) and the decryption algorithm \( D(PR_u, C) \), where \( PK_u \) is the public key, \( PR_u \) is the private key of \( U \) and \( M \) and \( C \) are the texts to be encrypted or to be decrypted, respectively. Each user publishes his encryption key by putting it on a public directory, while the decryption key is kept secret by himself. Suppose that user \( A \) wants to send a message \( M \) to user \( B \). First, \( A \) finds the public encryption key, namely \( PK_b \), for \( B \) from the public directory. Then \( A \) encrypts the message \( M \) to \( C \) by \( C = E(PK_b, M) \) and sends \( C \) to \( B \). On receiving \( C \), \( B \) can decode it by computing \( M = D(PR_b, C) \). Since \( PR_b \) is private for \( B \), no one else can perform this decryption process. Therefore, for practical purposes, the encryption and decryption algorithms \( E \) and \( D \) have to satisfy the following three requirements:

1) \( D(PR_u, E(PK_u, M)) = M \)
2) Neither of algorithms \( E \) and \( D \) needs much computing time.
3) To derive the associate \( PR_u \) from the publicly known \( PK_u \) is computationally infeasible [5].

A number of public-key cryptosystems have been proposed [1], [3], [7], [9], [17], [20]–[22], [26]. These systems can be put into two categories. One is based on hard number theoretic problems such as factoring, taking discrete logarithms, etc.; while the other is related to NP-complete problems such as \( 0/1 \) knapsack and so on. To construct cryptosystems based on these computationally hard problems, secret “trapdoor” information is added such that a one-way function is invertible. A function \( F \) is called a one-way function if and only if the computation of \( F(x) \) is easy for all \( x \) in the domain of \( F \), while it is computationally infeasible to compute the inverse \( F^{-1}(y) \) given any \( y \) in the range of \( F \), even if \( F \) is known. It is a trapdoor one-way function if the inverse becomes easy when certain additional information is given. This additional information is used as a secret decryption key.

In this paper, a new public-key cipher scheme is proposed. By the use of our scheme, the generating steps of keys are simple. Both the encryption and decryption procedures can be completed efficiently. Our cipher scheme is based upon the Diophantine equations [18]. In general, a Diophantine equation is defined as follows: We are given a polynomial equation \( f(x_1, x_2, \ldots, x_n) = 0 \) with integer coefficients and we are asked to find rational or integral solutions. Throughout this paper, we shall assume that the solutions are nonnegative. For instance, consider the following equation:

\[
3x_1 + 4x_2 + 7x_3 + 5x_4 = 78.
\]

The above equation is a Diophantine equation if we have to find a nonnegative solution for this equation. In fact, our solution is \( (x_1, x_2, x_3, x_4) = (2, 5, 1, 9) \). Another example of a Diophantine equation is

\[
3x_1^2 + 4x_1x_2x_3 + 5x_4 = 105.
\]

Diophantine equations are usually hard solve. In [14], it was proved that the problem of deciding whether there are positive integer solutions for

\[
\alpha x_1^2 + \beta x_2 - \gamma = 0,
\]

where \( \alpha, \beta \) and \( \gamma \) are positive integers, is NP-complete [4], [8]. Some specific cases of Diophantine equations and their computational complexities were studied in [24], [25].

A famous Diophantine equation problem is Hilbert’s tenth problem [11], which is defined as follows: Given a system of polynomials \( P_i(x_1, x_2, \ldots, x_n), 1 \leq i \leq m \), with integer coefficients, determine whether it has a nonnegative integer solution or not. In [15] and [23], it was shown that the Hilbert problem is undecidable for polynomials with degree 4. It was shown in [16] that the Hilbert problem is undecidable for polynomials with 13 variables. Gurari and Ibarra [10] also proved that several Diophantine equations are in NP-complete class.
Finally, we sketch the organization of this paper as follows. Underlying mathematics is described in Section II. The generation of the system, encryption and decryption algorithms, will appear in Section III. Section IV investigates the security of our cipher scheme. We also show that in order to break our system, one has to solve some specific Diophantine equations. Finally, conclusions are made in Section V.

II. THE UNDERLYING MATHEMATICS

In this section, we describe the mathematics on which the new cryptosystem is based. Let \( w \) be some positive integer and the domain \( D \) be a set of positive integers in the range of \([0, w]\). Let \( w = 2^k - 1 \), where \( b \) is some positive integer. Assume that a sending message \( M \) with length \( nb \) bits is broken up into \( n \) pieces of submessages, namely \( m_1, m_2, \cdots, m_n \). Each submessage is of length \( b \) bits. In other words, we can represent each submessage by a decimal number \( m_i \), and \( m_i \) in \( D \).

Suppose that \( n \) pairs of integers \( (q_i, k_i) \), \( (q_2, k_2) \), \( \cdots \), and \( (q_n, k_n) \) are chosen such that the following conditions hold:

1. \( q_i \)‘s are pairwise relatively prime; i.e., \( (q_i, q_j) = 1 \) for \( i \neq j \).
2. \( k_i > w \) for \( i = 1, 2, \cdots, n \).
3. \( q_i > k_i w (q_i \mod k_i) \), and \( q_i \mod k_i \neq 0 \) for \( i = 1, 2, \cdots, n \).

These \( n \) integer pairs \( (q_i, k_i) \)’s will be kept secret and used to encrypt messages. For convenience, we name the above three conditions the DK-conditions since they will be used as deciphering keys. Note that for the generating of pairwise relatively primes, one can consult [2]. Furthermore, the following numbers are computed. First, compute \( R_i = q_i \mod k_i \) and compute \( P_i \)’s such that two conditions are satisfied: 1) \( P_i \mod q_i = R_i \), and 2) \( P_i \mod q_i = 0 \) if \( i \neq j \). Since \( q_i \)’s are pairwise relatively primes, one solution for \( P_i \)’s satisfying the above two conditions is that \( P_i = Q_i b_i \) with

\[
Q_i = \prod_{j=1 \atop j \neq i}^{n} q_j
\]

and \( b_i \) is chosen such that \( Q_i b_i \mod q_i = R_i \). Since \( Q_i \) and \( q_i \) are relatively prime, \( b_i \)’s can be found by using the extended Euclid’s algorithm [5]. Note that the average number of divisions performed by the extended Euclid’s algorithm for finding \( b_i \) is approximately \( 0.843 \cdot \ln(q_i) + 1.47 \) [13].

Secondly, compute \( N_i = [q_i/(k_i R_i)] \) for \( i = 1, 2, \cdots, n \).

Finally, compute

\[
s_i = P_i N_i \mod Q_i, \quad \text{where} \quad Q = \prod_{i=1}^{n} q_i
\]

(1)

That is, we have a vector \( S = (s_1, s_2, \cdots, s_n) \) with each component computed as above.

After this, \( S \) can be used as the enciphering key for encrypting messages. By conducting a vector product between \( M = (m_1, m_2, \cdots, m_n) \) and \( S = (s_1, s_2, \cdots, s_n) \); i.e.,

\[
C = E(S, M) = M \ast S = \sum_{i=1}^{n} m_i s_i
\]

(2)

a message \( M \) is transformed to its ciphertext \( C \), where \( \ast \) denotes the vector product operation. Conversely, the \( i \)th component \( m_i \) in \( M \) can be recovered by the following operation:

\[
m_i = D((q_i, k_i), C) = \{ [k_i C/q_i] \mod k_i \} \quad \text{for} \quad i = 1, 2, \cdots, n.
\]

(3)

Theorem 2.1 shows that (3) is the inverse function of (2). The following lemmas are helpful in the proof of the theorem.

Lemma 2.1.
Let \( a \) and \( b \) be some positive integers where \( b > a \). Then for all \( x \), \( \frac{a[x/b]}{x} < \frac{a}{x(b-a)} \).

Proof: Let \( x = y^2 \) for some integer \( c \). Then \( \frac{y}{b} < c < \left( \frac{x}{b} + 1 \right) \). We have

\[
ac < (ax/b + a).
\]

(4)

On the other hand, if \( x \geq \frac{ab}{b-a} \), then \( (b-a)x \geq ab \); that is,

\[
(ax/b + a) \leq x.
\]

(5)

Combining (4) and (5), we have that \( a[x/b] < x \) if \( x \geq \frac{ab}{b-a} \).

Lemma 2.2.
Let \( R_i = q_i \mod k_i \), then \( k_i R_i m_i [q_i/(k_i R_i)] \mod k_i q_i = k_i R_i m_i [q_i/(k_i R_i)] \).

Proof: Let \( a = R_i m_i, b = k_i R_i, \) and \( x = q_i \). Since \( q_i > k_i w \), we know that \( q_i > k_i R_i w \).

That is, \( x \geq \frac{ab}{b-a} \) is satisfied. By applying Lemma 2.1, it can be seen that \( m_i [q_i/(k_i R_i)] < q_i \).

Therefore, \( k_i R_i m_i [q_i/(k_i R_i)] \mod k_i q_i = k_i R_i m_i [q_i/(k_i R_i)] \).

Lemma 2.5.
Let \( m_i \), \( k_i \), \( s_i \), and \( q_i \)’s be chosen such that the DK-conditions are satisfied. Let \( R_i = q_i \mod k_i \). Then

\[
k_i R_i m_i [q_i/(k_i R_i)] = m_i.
\]

Proof: Let \( \delta \) be \( [k_i R_i m_i [q_i/(k_i R_i)]]/q_i \). It can be easily seen that the following two inequalities hold:

\[
\delta < \lfloor k_i R_i m_i [q_i/(k_i R_i)] + 1/q_i \rfloor
\]

(6)

and

\[
\delta \geq \lfloor k_i R_i m_i [q_i/(k_i R_i)] \rfloor.
\]

(7)

Furthermore, the right-hand side of (7) is identical to \( m_i \) and that of (6) is \( \lfloor m_i + k_i R_i m_i/q_i \rfloor \). On the other hand, since \( m_i \) is an integer and \( k_i R_i m_i/q_i < 1 \), the right-hand side in (6) becomes \( \lfloor m_i + k_i R_i m_i/q_i \rfloor = m_i \). Combining these two inequalities, we obtain that \( m_i \leq \delta < m_i \). Finally, we have \( \delta = m_i \), since \( \delta \) is an integer.

Theorem 2.1.
Let \( (q_1, k_1), (q_2, k_2), \cdots, \) and \( (q_n, k_n) \) be \( n \) pairs of positive integers satisfying the DK-conditions. Let the vector \( S \) be computed by applying (1). Then (3) is the inverse function of (2). That is, a message encrypted by (2) can be decrypted by (3).

Proof: Let us prove the theorem by the following two steps. First, from (1), define \( \bar{s}_i = P_i N_i \); we have a vector \( \bar{S} = (\bar{s}_1, \bar{s}_2, \cdots, \bar{s}_n) \); i.e., \( s_i = \bar{s}_i \mod Q_i \) for \( i = 1, 2, \cdots, n \).

Let \( C' = M \ast \bar{S} = \sum_{i=1}^{n} m_i \bar{s}_i = \sum_{i=1}^{n} m_i P_i N_i \). Since \( P_i \)'s satisfy the following two conditions, 1) \( P_i \mod q_i = q_i \mod k_i = R_i \); and 2) \( P_i \mod q_i = 0 \) if \( i \neq j \), \( k_i C' \mod
afterward, the integer is described. First, an informal description is given. Then algorithms for constructing the cryptosystem, encrypting messages, and decrypting messages, respectively, are presented.

First, each user picks \( n \) pairs of parameters \((q_i, k_i), (q_2, k_2), \ldots, (q_n, k_n)\) such that the DK-conditions are satisfied. Afterward,

\[
Q = \prod_{i=1}^{n} q_i
\]

and \( N_i = [q_i/(k_i, q_i, \text{mod} k_i)] \) are computed, and \( b_i \)'s are integers chosen such that \( Q b_i \) is \( q_i \) \( \text{mod} k_i \), for \( i = 1, 2, \ldots, n \). Let \( P_i = Q b_i \) and \( s_i = P_i N_i \) \( \text{mod} k_i \), for \( i = 1, 2, \ldots, n \), where

\[
Q = \prod_{i=1}^{n} q_i.
\]

Therefore, a vector \( S = (s_1, s_2, \ldots, s_n) \) is obtained. Then the \( n \)-tuple \( S \) of integers is published and used as the public key of the cryptosystem for encoding messages.

The chosen parameters \((q_1, k_1), (q_2, k_2), \ldots, (q_n, k_n)\) are kept and used as the private key to decipher messages received. Specifically, let user \( A \) be the sender and user \( B \) be the receiver, and let \( A \) be sending a message represented by

\[
M = (m_1, m_2, \ldots, m_n),
\]

where \( m_i \) is a \( b \)-bits submessage represented by a decimal number in the range of \([0, 2^b - 1]\). Then \((m_1, m_2, \ldots, m_n)\) is encrypted by (2) into an integer \( C \). Afterward, the integer \( C \) is sent to user \( B \) as the ciphertext of the original message \( M \). On the receiving of integer \( C, \) user \( B \) is able to convert \( C \) into \((m_1, m_2, \ldots, m_n)\) by applying (3).
(s_1, s_2, s_3) = 3233622 and sends the integer C to B instead of sending the original message M.

When B receives the integer C, he can reveal the original message M by applying (3) on the received integer C. He will obtain

\[ m_1 = \left[ \frac{k_1 C}{q_1} \right] \mod k_1 = \left[ \frac{6 \times 3233622}{104} \right] \mod 6 = 19401732/104 \mod 6 = 186555 \mod 6 = 3, \]
\[ m_2 = \left[ \frac{k_2 C}{q_2} \right] \mod k_2 = \left[ \frac{8 \times 3233622}{147} \right] \mod 8 = [25868976/147] \mod 8 = 175979 \mod 8 = 3, \]
\[ m_3 = \left[ \frac{k_3 C}{q_3} \right] \mod k_3 = \left[ \frac{7 \times 3233622}{121} \right] \mod 7 = [22635354/121] \mod 7 = 187069 \mod 7 = 1. \]

That is, \((m_1, m_2, m_3) = (3, 3, 1)\), or the corresponding binary strings \((111101)\), is obtained. By concatenating the three submessages together, the original message \(M = (111101)\) is thus revealed.

IV. SECURITY OF THE CRYPTOSYSTEM

In this section, we investigate the security of the proposed method. Since there exists no technique to prove that a given encryption scheme is absolutely secure, the only approach available for us is to see whether anyone can think of a way to break it [21]. In the following, we examine some possible attacks on the cryptosystem from the viewpoint of a cryptanalyst. Two possibilities are considered. First, the cryptanalyst tries to decipher an intercepted ciphertext. Second, the cryptanalyst does not decipher a ciphertext directly, but tries to determine the secret decryption key. With this key, he will have the same capability as the legitimate message receiver for deciphering messages.

A. Brute Force for Deciphering the Ciphertext

With the publicly known encryption key \(S\) and the intercepted ciphertext \(C\), a cryptanalyst may try to decode the Step 1 in Algorithm 3.3 without knowing the private key \(PR_b\) of the legitimate receiver. To decrypt the ciphertext in this case, he has to solve the following problem. For convenience, we call it the linear Diophantine equation problem. Let \(S = \{s_i : i = 1, 2, \cdots, n\}\) be a set of given positive integers and \(C\) be a positive integer. The linear Diophantine equation problem is to determine a sequence of nonnegative integers, \(M = (m_1, m_2, \cdots, m_n)\), such that

\[ \sum_{i=1}^{n} m_i s_i = C. \]

We shall prove that the linear Diophantine equation problem is NP-complete. It can be reduced from the integer knapsack problem, which has been proved to be in the class of NP-completeness [8]. For better understanding, we present the integer knapsack problem briefly here.

**Integer Knapsack Problem** [8]: Given an \(n\)-tuple \(S = (s_1, s_2, \cdots, s_n)\), and two positive integers \(e\) and \(f\), determine whether there is a sequence of nonnegative integers, \(M = (m_1, m_2, \cdots, m_n)\), such that

\[ \sum_{i=1}^{n} m_i s_i \leq f \]

and such that

\[ \sum_{i=1}^{n} m_i s_i \geq e? \]

**Theorem 4.1**: The linear Diophantine equation problem is NP-complete.

**Proof**: Suppose that there exists an algorithm, called procedure \(IK(S, C)\), with inputs \(S\) and \(C\), and output "yes" or "no," which can solve the linear Diophantine equation problem in polynomial time. By applying procedure \(IK(S, C)\), we can also solve the integer knapsack problem in polynomial time.

Procedure \(IK(S, e, f)\) is as follows:

**procedure** \(IK(S, e, f)\)

**begin**

**boolean**: flag

**for** \(I = e\) **to** \(f\) **do**

**if** \(X(S, I) = \text{"yes"} \) **then** flag = true

**endfor**

**if** flag = true **then** print ('there exists a solution')

**else** print ('there is no solution')

**end**

**endprocedure**

Therefore, the integer knapsack problem is reduced to the linear Diophantine equation problem with the reduction process done in polynomial time. Finally, using the fact that the linear Diophantine equation problem is in NP and the fact that the integer knapsack problem is NP-complete, we have that the linear Diophantine equation problem is NP-complete.

B. Brute Force to Reconstruct the Secret Decryption Key

On the other hand, a cryptanalyst may not be interested in deciphering the intercepted ciphertext. He may try to reveal the decryption key that is kept private by the receiver. Knowing this secret key, he will be able to decipher any message sent to the receiver as he wants. Nevertheless, how can he determine the private decryption key? That is, how can he reconstruct the secret key by knowing the public key? Specifically, he has to solve the following problem: Given \(n\) integers \(s_1, s_2, \cdots, s_n\), find the corresponding \(n\) pairs \((q_1, k_1), (q_2, k_2), \cdots, (q_n, k_n)\).

We assume that the key generating procedure is known to him. From Step 2 to Step 4 in Algorithm 3.1, since \(s_i = P_i N_i\)
mod \( Q \) and \( P_i \mod q_i = R_i \), he can deduce that \( s_i \equiv R_i N_i \) (mod \( q_i \)) for \( i = 1, 2, \ldots, n \). In other words, the following equations are obtained:

\[
s_i \mod q_i = R_i N_i = R_i [q_i/(k_i R_i)]
\]

where

\[
R_i = q_i \mod k_i, \quad 1 \leq i \leq n.
\]

Equation (8) can be rewritten as

\[
s_i = q_i x_i + R_i [q_i/(k_i R_i)], \text{ for some } x_i, \quad 1 \leq i \leq n.
\]

Let \( v_i = [q_i/(k_i R_i)] \). Then \( v_i - 1 < q_i/(k_i R_i) \leq v_i \) and \( k_i R_i (v_i - 1) < q_i \leq k_i R_i v_i \). We have

\[
q_i = k_i R_i (v_i - 1) + y_i, \text{ with } 1 \leq y_i \leq k_i R_i, \quad 1 \leq i \leq n.
\]

Substituting (10) into (9), we obtain the following equations

\[
k_i R_i (v_i - 1) x_i + y_i x_i + R_i v_i - s_i = 0\]

with

\[
1 \leq y_i \leq k_i R_i, \quad 1 \leq i \leq n.
\]

Equation (11) is a system of \( n \) Diophantine equations with degree 4 and has variables \( k_i, R_i, v_i, x_i, \) and \( y_i \), for \( 1 \leq i \leq n \). Our job of breaking the cipher system consists of the following steps:

**Step 1.** Find \( k_i, R_i, v_i, x_i, \) and \( y_i \) satisfying (11), for \( 1 \leq i \leq n \).

**Step 2.** Calculate \( q_i \) by using (10).

**Step 3.** Check whether \( q_i \)'s are relatively prime. If they are not, go to Step 1. Otherwise, we have found at least one possible solution in the form of \( ((q_1, k_1), (q_2, k_2), \ldots, (q_n, k_n)) \).

**Step 4.** Randomly generate a message \( M = (m_1, m_2, \ldots, m_n) \). Encrypt \( M \) by the Step 4 in Algorithm 3.2 into an integer \( C \).

**Step 5.** Decrypt \( C \) into \( M' \) by Step 1 in Algorithm 3.3 using the \( n \) pairs \( ((q_1, k_1), (q_2, k_2), \ldots, (q_n, k_n)) \) obtained.

**Step 6.** If \( M' = M \) generated in Step 4 are equal, stop; otherwise go to Step 1 again.

Up to now, there seems to be no easy way of executing Step 1 (solving a Diophantine equation with degree 4). Even if we succeed, there is no guarantee that the \( q_i \)'s found by us are relatively prime to one another. Therefore, it seems difficult to break our system in this way.

**C. Attack Due to the Greatest Common Divisor of \( s_i \)'s**

Another ciphertext attack is to observe the greatest common divisor of \( s_i \)'s. On intercepting the ciphertext \( C \) and the publicly known \( s_1, s_2, \ldots, s_n \), the cryptanalyst hopes to decrypt \( C \) into \( M \) as in the Step 1 of Algorithm 3.3. Since the cryptanalyst has no legitimate \( (q_i, k_i) \)'s, \( m_i \) may be obtained by the following exhaustive searching steps.

**Step 1.** Compute \( t_i \) for \( i = 1, 2, \ldots, n \), as follows

\[
t_i = \frac{\gcd(s_1, s_2, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n)}{\gcd(s_1, s_2, \ldots, s_{i-1}, s_i, s_{i+1}, \ldots, s_n)}
\]

where \( \gcd \) denotes the greatest common divisor.

**Step 2.** Compute \( r_{ij} = j - s_i \mod t_i, \) for \( j = 1, 2, \ldots, w, \) and \( i = 1, 2, \ldots, n \), where \( w = 2^n - 1 \) if each submessage is of length \( b \) bits.

**Step 3.** Compute \( b_i = C \mod t_i \), for \( i = 1, 2, \ldots, n \).

**Step 4.** Search \( b_i \) for \( i = 1, 2, \ldots, n \), from the set \( \{r_{i1}, r_{i2}, \ldots, r_{iw}\} \). If \( b_i = r_{ik} \), then \( m_i = k \).

From the above procedure, \( m_i \) seems to be deducible from \( C \) and \( (s_1, s_2, \ldots, s_n) \). However, if we decompose the message into submessages of length 100 bits each; i.e., \( b = 100 \), then \( w = 2^{100} - 1 \). This number has magnitude of value about \( 10^{19} \). If we use a computer that can test \( 10^5 \) numbers per second. It requires about \( 2.7 \times 10^{10} \) years to complete the search for each \( b_i \). The Step 4 of exhaustive searching in the above algorithm will be extremely impossible.

**V. CONCLUSION AND DISCUSSION**

A new public-key cryptosystem is investigated in this paper. The motivation of this attempt is trying to use real numbers for its dense property. However, if real numbers are used as keys, several disturbing problems, such as representation and precision will be encountered. With the help of integer functions, the possibility of using an integer as a key is increased significantly. That is, for a cryptanalyst who tries to break the cipher, he has to conduct an exhaustive search on a long list of integer numbers.

Further, we would make some discussion on the parameters used in the presented cipher scheme. By using a concept similar to that of block cipher [5], a sending message of length \( nb \) bits will be broken into \( n \) pieces of submessages with each \( b \) bits long. The time complexity needed to compute \( q_i \)'s will be proportional to \( n^2 \) as \( n \) increases [2]. When \( q_i \)'s are determined, \( k_i \)'s can be chosen from 2) and 3) in the DK-conditions. Thus the time required to choose \( k_i \)'s is proportional to \( n \). Further, the time needed to find \( b_i \)'s grows at the rate of \( n(\log n) \) when \( q_i \)'s and \( k_i \)'s are determined.

From Section IV, we know that the execution time required, for a cryptanalyst to solve the corresponding problems, increases when \( n \) increases. Theoretically, the security of the presented scheme will be increased as \( n \) is large. For instance, when \( n = 100 \) and \( b = 100 \), it will be rather difficult to solve the problems presented in Section IV. Further, let us estimate how large the \( C \) value is. We consider that the number of bits needed to store the product of the first \( n \) prime numbers is proportional to \( n(\log n) \). Then the number of bits required to represent \( s_i \) is proportional to \( n(\log n) \). In other words, the number of bits to represent a \( C \) value is proportional to \( b + n(\log n) + (\log n) \), where \( b \) is the number of bits in each submessage. Since a sending message is of length \( bn \) bits. We conclude that the ciphertext expansion rate of the presented scheme is \( O(\log n) \).

Finally, we would like to point out that the advantage of the presented scheme is that the encryption and decryption steps
are relatively easy. For encryption, it requires $n$ multiplication operations and $n$ addition operations. For decryption, $n$ multiplication operations and $n$ modulus operations are needed. Thus, from the viewpoint of computation time, our algorithm is rather efficient.

ACKNOWLEDGMENT

The authors thank the referees for many helpful suggestions and comments.

REFERENCES

R. C. T. Lee (A’74-M’75-SM’86-F’89) received the B.S. degree in electrical engineering from the National Taiwan University in 1961 and the M.S. and Ph.D. degrees from the University of Berkeley, in 1963 and 1967, respectively, all in electrical engineering and computer science.

Dr. Lee worked for National Cash Register, Hawthorn, California, the National Institutes of Health, Bethesda, MD, and the Naval Research Laboratory, Washington, DC before joining the National Tsing Hua University in 1975. At the National Tsing Hua University, he has been department chairman for the Applied Mathematics and the Computer Science and Electrical Engineering departments, Dean of Engineering, Provost, and Acting President of National Tsing Hua University. His present job is President of Providence University.

Dr. Lee has published nearly fifty journal papers on various subjects in computer science, including mechanical theorem proving, pattern recognition and clustering analysis, database design, and sequential and parallel algorithm design. He was a co-author of the book, Symbolic Logic and Mechanical Theorem Proving (Academic Press), which has been translated into Japanese, Italian, and Russian. This book has been so popular that Academic Press selected it as one of four Computer Science Classics. His article on clustering analysis “Clustering Analysis and its Applications” appeared in Advances in Information System Science (J. T. Tou Ed., Plenum Press), and he also has a chapter on compiler writing in Handbook of Software Engineering (C. R. Vich and C. V. Ramamoorthy Eds., Van Norstrand Reinhold). He was recently invited to write an article on parallel computing that appeared in Advances in Parallel Computing (D. J. Evans, Ed., JAI Press). His book on algorithms will be published by Prentice Hall International. Dr. Lee has organized more than twenty international conferences. He is now an Editor or Associate Editor of the following journals: International Journal of Pattern Recognition and Machine Intelligence, Annals of Mathematics and Artificial Intelligence, IEEE Transactions on Knowledge and Data Engineering, International Journal of Foundations of Computer Science, Computers and Operations Research, Journal of Parallel Algorithms and Applications and International Journal of Computational Geometry and Applications, Journal of Parallel Algorithms and Applications, and Information Science Journal. He is presently a reviewer for Mathematical Reviews.