A Secure Cryptosystem based on Affine Transformation [†]

Mohsen Toorani [‡] Abolfazl Falahati

Abstract

In this paper, it is proved that Lin et al.'s scheme that tried to strengthen the Hill cipher against the known-plaintext attack has several security flaws and is vulnerable to the chosen-ciphertext attack. This paper also introduces a secure and efficient symmetric cryptosystem based on affine transformation. The proposed cryptosystem includes an encryption algorithm that is an improved variant of the Affine Hill cipher, and two cryptographic protocols that are introduced for the proposed cryptosystem.

1. Introduction

The Hill cipher was invented in 1929 by Lester S. Hill [1, 2]. It is a famous polygram and classical ciphering algorithm based on matrix transformation that its attributes, including its cryptanalysis are described in some cryptographic textbooks [3, 4]. Although susceptibility of the Hill cipher to cryptanalysis has rendered it unusable in practice, it still serves an important pedagogical role in both cryptology and linear algebra. The Hill cipher is a block cipher that has several advantages such as disguising letter frequencies of the plaintext, its simplicity because of using matrix multiplication and inversion for encryption and decryption, and its high speed and high throughput [5] but it is vulnerable to the known-plaintext attack [6].

Several researchers tried to improve the security of the Hill cipher. Yeh et al. [7] used two co-prime base numbers that are securely shared between the participants but their scheme is not efficient and requires many manipulations. Saeednia [8] tried to make the Hill cipher secure using some random permutations of columns and rows of the key matrix but it is proved that his cryptosystem is vulnerable to the known-plaintext attack [9], the same vulnerability of the original Hill cipher. Ismail et al. [5] tried to improve the Hill cipher's security by introduction of an initial vector that multiplies successively by some orders of the key matrix to produce the corresponding key of each block but it has several inherent security problems [10]. Lin et al. [9] claimed that taking some random numbers and using a one-way hash function thwarts the known-plaintext attack to the Hill cipher but their scheme is not so

[†] **Copyright © John Wiley & Sons, Ltd**. Reprinted from **Journal of Security and Communication Networks, Vol. 4, No. 2, pp. 207-215, Feb. 2011.** Original published version of this manuscript is available at: http://dx.doi.org/10.1002/sec.137.

[‡] Corresponding Author, ResearcherID: A-9528-2009

efficient and in this paper, we prove that it is vulnerable to the chosen-ciphertext attack due to a great security flaw in the underlying protocol of their scheme.

The main contribution of this paper is to introduce a secure cryptosystem that is a variant of the Affine Hill cipher, which overcomes all of its security drawbacks. Our proposed cryptosystem includes an encryption algorithm for which two secure cryptographic protocols are introduced. The first one is a two-pass protocol that is a variant of the Hughes key-exchange protocol [11] and includes an authentication step to thwart the man-in-the-middle attack. The second protocol is one-pass and is suitable whenever both of participants are not online. The encryption core of the proposed cryptosystem has the same structure of the Affine Hill cipher but its internal manipulations are different from the previously proposed schemes. Although HMAC is usually used for the purpose of authentication, we utilize it in the encryption core of our proposed cryptosystem and for extracting the corresponding random number of each block in a hash chain. This has been accomplished due to the inherent advantage of HMAC over ordinary hash functions, and to give more randomization to the linear structure of the Affine Hill cipher, especially when the output of HMAC can be considered as a random number.

The rest of this paper is organized as follows. Section 2 briefly introduces the Hill cipher. Section 3 is devoted to Lin et al.'s scheme [9] and its cryptanalysis. The proposed cryptosystem and its attributes, including evaluation of its computational costs is presented in Section 4. A key generation algorithm for the proposed cryptosystem is described in Section 5, and Section 6 concludes the paper.

2. The Hill Cipher

In the Hill cipher, the ciphertext is obtained from the plaintext by means of a linear transformation. The plaintext row vector **X** is encrypted as $\mathbf{Y} = \mathbf{XK} (\bmod m)$ in which **Y** is the ciphertext row vector, **K** is an $n \times n$ key matrix where $k_{ij} \in \mathrm{Z}_m$, Z_m is ring of integers modulo *m*, and *m* is a natural number that is greater than one. The encryption procedure proceeds with encoding the resulted ciphertext row vector into alphabets of the main plaintext. The ciphertext **Y** is decrypted as $\mathbf{X} = \mathbf{YK}^{-1} (\bmod m)$.

For decryption to be possible, the key matrix **K** that is securely shared between the participants should be invertible or equivalently, it should satisfy $\gcd(\det \mathbf{K} (\bmod m), m) = 1$ [6]. The value of modulus *m* in the original Hill cipher was 26 but its value can be optionally selected. Actually, many of square matrices are not invertible over Z_m. The keyspace of the Hill cipher is $GL(n, \mathrm{Z}_m)$, the group of $n \times n$ matrices that are invertible over Z_m. As it is proved in [12], when $m = \prod_i p_i^{e_i}$ is a composite modulus, we have:

$$|GL(n, Z_m)| = \prod_i \left(p_i^{(e_i - 1)n^2} \prod_{k=0}^{n-1} (p_i^n - p_i^k) \right) \tag{1}$$

and the proportion of $n \times n$ invertible matrices will be:

$$f(n, m) = \prod_i \prod_{j=1}^{n} (1 - \frac{1}{p_i^j}) \tag{2}$$

Thus, the probability of a randomly selected square matrix to be invertible is about one for any large prime modulus while it is almost zero for a composite modulus with many different prime divisors so the risk of determinant having common factors with the modulus can be reduced by taking a prime number as the modulus. Such selection also increases the keyspace of the cryptosystem since a prime modulus generates a larger keyspace than that of a composite modulus [12]. The keyspace also increases with increase in *n*, the rank of the key matrix, as it is apparent from (1). For the sake of increasing the keyspace and improving the security, we should increase the rank of the key matrix and choose a large enough prime number as the modulus but it may increase the running time and decrease the efficiency. This is a tradeoff between the security and efficiency.

The security of the Hill cipher depends on confidentiality of the key matrix **K** and its rank *n*. When *n* is unknown and the modulus *m* is not too large, the opponent could simply try successive values of *n* until the key is found. If the guessed value of *n* was incorrect, the obtained key matrix would be disagreed with further pairs of plaintext and ciphertext. The most important security flaw of the Hill cipher is regarded to its vulnerability to the known-plaintext attack since it can be broken by taking *n* distinct pairs of plaintext and ciphertext [6, 7].

The Affine Hill cipher is an extension to the Hill cipher that mixes it with a nonlinear affine transformation [6] so the encryption expression has the form of $\mathbf{Y} = \mathbf{XK} + \mathbf{V} (\bmod m)$. In this paper, we extend this concept in the encryption core of our proposed cryptosystem.

3. Cryptanalysis of Lin et al.'s Cryptosystem

3.1. Lin et al.'s Scheme

The Lin et al.'s scheme [9] is depicted in Figure 1. Alice selects a random integer *a* in the range $0 < a < m$ where *m* can be a composite number. She concatenates *a* with other elements of **K** and computes $b = h(a \,||\, k_{11} \,||\, k_{12} \,||\, ... \,||\, k_{ij} \,||\, ... \,||\, k_{nn})$ in which $h(x)$ is a one-way hash function and || denotes the concatenation. She picks up k_{ij} that is the ij^{th} element of the key matrix **K** where $i = 1 + ([\frac{b-1}{n}] \bmod n)$ and $j = b - [\frac{b-1}{n}] \times n$. She produces the row vector $\mathbf{V} = [v_1 \quad v_2 \quad \ldots \quad v_n]$ using the recursive expression $v_t = h(v_{t-1}) \bmod m$, $t = 2,3,\ldots,n$ where $v_1 = h(k_{ij}) \bmod m$. Alice encodes the plaintext into some row vectors $\mathbf{X} = [x_1 \quad x_2 \quad \ldots \quad x_n]$ and encrypts them as $\mathbf{Y} = k_{ij}\mathbf{X}\mathbf{K} + \mathbf{V} (\bmod m)$. She finally sends the pairs of (**Y**, *a*) to Bob. On the other hand, Bob proceeds the same procedure with the clearly sent random number *a* and computes *b*, k_{ij} and **V**. He decrypts the received ciphertext as $\mathbf{X} = k_{ij}^{-1}(\mathbf{Y} - \mathbf{V})\mathbf{K}^{-1} (\bmod m)$. It should be notified that the decryption expression in [9] was incorrect so it is modified in this paper.

3.2. Vulnerability to Chosen-Ciphertext attack

The encryption expression of the Lin et al.'s scheme has the form of $\mathbf{Y}_t = C_t\mathbf{X}_t\mathbf{K} + \mathbf{V}_t (\bmod m)$ in which C_t is the corresponding k_{ij} for the t^{th} block of data. Even if the attacker knows *n* pairs of $(\mathbf{X}_t, \mathbf{Y}_t)$ in which $t = 1,\ldots,n$, the key matrix **K** and parameters C_t and $\mathbf{V}_t$ are unknown. Lin et al. claimed that the known-plaintext attack cannot be accomplished on their scheme since *n* equations cannot be used for solving an unknown $n \times n$ matrix and *2n* unknown parameters [9]. However, we show that their scheme is vulnerable to the chosen-ciphertext attack if the attacker chooses those equations that have the same unknown parameters C_t and $\mathbf{V}_t$. In this kind of attack, the cryptanalyst can choose different ciphertexts and has access to the corresponding plaintexts. He will try to deduce the key. Although this kind of attack is most relevant to public-key cryptosystems, it can also be effective against a symmetric algorithm [12].

The vulnerability of Lin et al.'s scheme is that the values of *b* and **V**, and the choice of k_{ij} depend on the value of *a* and their values do not differ for the same value of *a*. Although *a* is randomly selected, it is clearly sent over the communication link which enables the eavesdropper to easily see and use it for a chosen-ciphertext attack. The attack can take place as follows: When Alice sends the pairs of (**Y**, *a*) to Bob, Eve eavesdrops and saves them. The random number *a* will be repeated in some pairs of (**Y**, *a*) especially when dealing with a bulk of data. This will be strengthened with a small choice of the modulus. Eve chooses *n*+1 pairs of (**Y**, *a*) that have the same random number *a*. According to the chosen-ciphertext attack, she has access to the corresponding plaintext of the chosen ciphertexts. For example, she may have access to a cryptographic module that automatically performs decryption. Now, she has a set of equations as $\mathbf{Y}_t = k_{ij}\mathbf{X}_t\mathbf{K} + \mathbf{V}(\bmod m)$, $t = 1,...,n+1$ where $\mathbf{X}_t$ and $\mathbf{Y}_t$ are the known parameters. She can easily derive the key matrix **K** using any of conventional methods described in the linear algebra. Although Lin et al. used the structure of the Affine Hill cipher, the row vector **V** can be easily eliminated from the pairs encrypted with the same random number so for such pairs, their scheme acts as the simple Hill cipher.

3.3. Further defects

In addition to vulnerability of the Lin et al.'s scheme to the chosen-ciphertext attack, it involves other faults and weaknesses. For each block of plaintext, a different random number should be generated and it should be transmitted with its corresponding ciphertext. This can decrease the efficiency due to the repeated random number generations and increased bandwidth requirements. Another weakness of their scheme is that the modulus can be a composite number so the keyspace and subsequently, the notion of security will be decreased. The composite modulus may also make the decryption impossible. For decryption to be possible in their scheme, it is necessary to have $\gcd(k_{ij}, m) = 1$ and $\gcd(\det \mathbf{K}(\bmod m), m) = 1$. Such necessary conditions may fail especially when m is composite. If the modulus is a prime number p, the decryption will be failed only when k_{ij} is zero mod p. However, for a composite modulus m, the picked up k_{ij} with a fair probability has some common divisors with the modulus that makes the decryption impossible.

4. The Proposed Cryptosystem

The proposed cryptosystem includes a ciphering core that is depicted in Figure 2 and has the same structure of the Affine Hill cipher. In order to give more randomization to the introduced scheme and to strengthen it against the common attacks, each block of data is encrypted using a random number. For avoiding multiple random number generations, only one random number is generated at the beginning of encryption and the corresponding random numbers of following data blocks are recursively generated using HMAC in a chain. The basic random number that is generated prior to encryption should be securely shared between the participants so it is necessary to introduce some cryptographic protocols. Figures 3 and 4 depict two cryptographic protocols for the proposed cryptosystem in which the encryption and decryption procedures should be followed from Figure 2. The first introduced protocol called “A” is a two-pass protocol while the second protocol named “B” is one-pass. Protocol “A” is actually an improved variant of the Hughes key-exchange protocol [11] that is selected because of its simplicity and our reluctance to involve any trusted third party but the Hughes protocol is vulnerable to the man-in-the-middle attack so protocol "A" modifies it by introduction of an authentication step. Protocol "B" is a new one-pass protocol that is designed for the proposed cryptosystem. As a one-pass protocol, it does not have any authentication step but it is secure, does not reveal any secret information, and is suitable for situations where both of participants are not online.

The following steps briefly describe the proposed cryptosystem using protocol "A" in which p' is a large prime number, *g* is a primitive element of the multiplicative group $Z^{*}_{p'}$, and values of p' and *g* can be publicly available to anyone. As stated in Section 2, the modulus *p* is a large prime number.

1. Alice secretly selects a random integer x in $0 < x < p' - 1$ and computes $a_0 = g^x (\bmod p')$. She encodes the plaintext message into some row vectors $\mathbf{X} = [x_1 \quad x_2 \quad \ldots \quad x_n]$. For the t^{th} block of data to be encrypted ($t = 1,2,\ldots$), she computes a_t with a recursive expression as $a_t = HMAC_{k'}(a_{t-1})$ in which $k' = (k_{11} \,\|\, k_{12} \,\|\, k_{13} \,\|\, \ldots \,\|\, k_{nn} \,\|\, a_{t-1}) \bmod 2^q$ is the required *q*-bits key of the deployed HMAC that is simply generated by taking *q*-bits from least significant bits of $(k_{11} \,\|\, k_{12} \,\|\, k_{13} \,\|\, \ldots \,\|\, k_{nn} \,\|\, a_{t-1})$. If a_t is invertible mod *p*, i.e. $a_t \not\equiv 0 (\bmod p)$, she puts $v_0 = a_t (\bmod p)$. Otherwise, she puts $v_0 = 1$. She produces the row vector $\mathbf{V} = [v_1 \quad v_2 \quad \ldots \quad v_n]$ with the recursive expression as $v_i = k_{ij} + \widetilde{v}_{i-1} a_t (\bmod p)$ for $i = 1,\ldots,n$ and $j = (v_{i-1} \bmod n) + 1$, in which $\widetilde{v}_{i-1}$ is defined as $\widetilde{v}_{i-1} = 2^{\lceil \gamma/2 \rceil} + (v_{i-1} \bmod 2^{\lceil \gamma/2 \rceil})$ where $\gamma = \lfloor \log_2 v_{i-1} \rfloor + 1$ denotes the bit-length of v_{i-1}, $\lfloor . \rfloor$ denotes the floor, and $\lceil . \rceil$ indicates the ceiling. She then encrypts all the plaintext vectors as $\mathbf{Y} = v_0 \mathbf{X}\mathbf{K} + \mathbf{V} (\bmod p)$. She repeats the procedure until all blocks of plaintext become encrypted.
2. Bob secretly selects a random integer y in $0 < y < p' - 1$ and computes $c = g^y (\bmod p')$ and $d = HMAC_{k''}(c)$ in which $k'' = (k_{11} \,\|\, k_{12} \,\|\, k_{13} \,\|\, \ldots \,\|\, k_{nn}) \bmod 2^q$. He then sends the pair (*c, d*) to Alice.
3. Alice receives the pair (*c, d*) and uses *c* to compute *d* as $d = HMAC_{k''}(c)$ for verifying the received value of *d*. This authenticates Bob and ensures Alice that Bob is the other party since it is assumed that only Bob knows the elements of the key matrix. She sends

Bob all the ciphertext vectors **Y**, produced in step 1 together with number $e = c^{x} (\bmod p')$.

4. Bob computes $u = y^{-1} (\bmod p')$ and uses it for retrieving a_0 as $a_0 = e^{u} (\bmod p')$. He uses a_0 for decrypting the ciphertext as $\mathbf{X} = v_0^{-1} (\mathbf{Y} - \mathbf{V}) \mathbf{K}^{-1} (\bmod p)$, as it is depicted in Figure 2.

Description of protocol "B" is similar to that of protocol "A" and can be easily followed from Figure 4.

4.1. Properties of the Proposed Scheme

The proposed cryptosystem neutralizes all the security drawbacks of the Hill cipher. It thwarts the known-plaintext attack with the same reasoning that was stated for the Lin et al.'s scheme. Choosing a large prime number *p* as the modulus has extremely enhanced the keyspace so the brute-force or equivalently, the ciphertext-only attack does not have any benefit for the attacker. The random number after a secure transmission is recursively encoded with HMAC so it differs for each block of plaintext. Moreover, the key of HMAC is another random number that is varying for each block of plaintext. The chosen-ciphertext and chosen-plaintext attacks are also thwarted since the random number a_0 that its knowledge is necessary to accomplish such attacks, is exchanged via a secure protocol. For the proposed cryptosystem, Eve cannot use the pairs (**Y**, *a*) for performing the chosen-ciphertext attack as it was the case for the Lin et al.'s scheme. Furthermore, protocol "A" includes an authentication step and prevents the man-in-the-middle attack so Mallory cannot impersonate Alice or Bob since he does not have the secret key matrix.

Utilizing HMAC in the proposed cryptosystem is to take advantage of the key matrix that is secretly shared between the involved participants and due to the advantages of HMAC over ordinary hash functions [13]. HMAC treats the hash function as a black box so the kind of its embedded hash function can be changed when necessary. HMAC executes in approximately the same time as its embedded hash function for long messages [14] but it provides further security [13, 15].

The introduced expression for generating the elements of vector **V** as $v_i = k_{ij} + \tilde{v}_{i-1} a_t (\bmod p)$ and defining $\tilde{v}_{i-1}$ as $\tilde{v}_{i-1} = 2^{\lceil \gamma/2 \rceil} + (v_{i-1} \bmod 2^{\lceil \gamma/2 \rceil})$ takes advantages of ideas behind the MQV key-exchange protocols [16]. $\tilde{v}_{i-1}$ is simply computed by taking the least significant half in binary representation of v_{i-1} and its definition in this way will decrease the computational costs and consequently, increases the efficiency [16].

The security of protocol “A” depends on the computational intractability of the Discrete Logarithm Problem (DLP) so certain considerations should be taken into account to assure its computational intractability. The security of exchanging the random number a_0 depends on difficulty of factoring numbers with the same size as p'. To thwart known attacks, the prime modulus p' should have at least 300 digits, and $p'-1$ should have at least one large prime factor [6]. p' should also be a safe prime, i.e. it should be selected in a way that $(p'-1)/2$ becomes a prime number too [17]. Theoretically, the parameter g that is used for generating a_0 as $a_0 = g^x (\mathrm{mod}\, p')$ should be a primitive element of the multiplicative group $Z^*_{p'}$, i.e. the powers of g should generate all the distinct integers from 1 to $p'-1$ in some order [13]. However, it actually does not have to be a primitive element. It just has to generate a large subgroup of the multiplicative group $Z^*_{p'}$ [11]. Random number generation is also an important issue for which certain considerations should be taken into account [18].

4.2. Computational Costs

In this section, the time complexity of the proposed scheme is evaluated. To have a fair comparison between the proposed cryptosystem and other schemes, only the computational costs of the ciphering core is considered and we neglect the required computations of the protocols. We also neglect the required computations for computing the inverse key matrix that is used for the decryption since it can be assumed that the key matrix and its inverse have been securely shared between the participants. Let T_{Enc} and T_{Dec} denote the running time for encryption and decryption of each block of data respectively. By considering the above-mentioned eliminations, we have:

$$T_{Enc} \cong (n^2 + 2n)T_{Mul} + (n^2 + n + 1)T_{Add} + T_{HMAC} \qquad (3)$$

$$T_{Dec} \cong (n^2 + 2n)T_{Mul} + (n^2 + n + 1)T_{Add} + T_{HMAC} + T_{Inv} \qquad (4)$$

in which T_{HMAC} is the running time for the HMAC calculations, and T_{Mul}, T_{Add} and T_{Inv} are the time needed for the modular multiplication, addition and inverse calculations respectively. T_{HMAC} depends on the kind of embedded hash function that is used within HMAC. Total number of operations for the HMAC-SHA1 calculation is as:

$$T_{HMAC-SHA1} = 32 + 1110 \times (2 + n_k) \qquad (5)$$

while for the case of HMAC-MD5, it is:

$$T_{HMAC-MD5} = 32 + 744 \times (2 + n_k) \qquad (6)$$

in which $n_k = \dfrac{N+k}{512}$ is the number of input blocks to the embedded hash function where N is the bit-length of the total message and k is the bit-length of extra-appended inner form of the key [19]. Each of T_{Mul}, T_{Add} and T_{Inv} requires different number of operations. Let $\zeta = \lfloor \log_2 p \rfloor + 1$ denotes the bit-length of modulus *p*. Using the conventional methods, we have:

$$T_{Add} = O(\zeta) \tag{7}$$

$$T_{Mul} = O(\zeta^2) \tag{8}$$

$$T_{Inv} = O(\zeta^3) \tag{9}$$

in which the big-*O* notation denotes the order of magnitude of the complexity [20]. There are many fast algorithms for the computations [18] but we consider the time complexity of the conventional methods since it corresponds with the worst situation. The computational costs of the proposed scheme for encrypting and decrypting each block of data can be simply estimated by substituting expressions (5-9) into (3) and (4). The running time for encryption and decryption of each block of data explicitly depends on ζ and *n*. Table 1 gives a comparison between the required number of operations for encrypting/decrypting each block of data in the proposed scheme and those of the other schemes. The required number of operations for encrypting each block of data using different schemes and for different rank values of the key matrix are depicted in Figure 5, where the plaintext and modulus are fixed and the deployed hash function is SHA-1. Regardless of the security advantages of the proposed scheme over the previously proposed schemes, Figure 5 explicitly exhibits its computational efficiency.

The total processing time for enciphering /deciphering the whole blocks of plaintext/ciphertext can be simply estimated by multiplying the running time of each block of data by the total number of blocks. The total number of blocks depends on the length of input data and the rank of the key matrix since the plaintext is divided into blocks of *n* letters. If the total plaintext has a length of *L* letters that is not a multiple of *n*, it should be padded until it becomes a multiple of *n* so the number of data blocks is $\left\lceil \dfrac{L}{n} \right\rceil$. For a fixed data length, increasing *n* will decrease the number of data blocks and vice versa. The running time for encrypting the whole plaintexts is:

$$T_{Total\_Enc} \cong \left\lceil \frac{L}{n} \right\rceil \left((n^2 + 2n) T_{Mul} + (n^2 + n + 1) T_{Add} + T_{HMAC} \right) \tag{10}$$

While the running time for decrypting the whole ciphertexts will be:

$$T_{Total\_Dec} \cong \left\lceil \frac{L}{n} \right\rceil \left((n^2 + 2n)T_{Mul} + (n^2 + n + 1)T_{Add} + T_{HMAC} + T_{Inv} \right) \qquad (11)$$

The computational costs of the proposed scheme for encrypting/decrypting all blocks of data is simply estimated by substituting expressions (5-9) into (10) and (11). Figure 6 depicts the effects of rank value of the key matrix on the total number of operations for encipherment of the whole plaintext that is obtained using (10) for $L = 400$ and $p = 29$. The size effects of the modulus p on the total number of operations for encipherment of the whole plaintext is also depicted in Figure 7 that is obtained using (10) for $L = 400$ and $n = 9$. It is noteworthy that the waves in Figure 6 are according to the introduced ceiling function in (10) while the steps in Figure 7 are due to logarithmic relationship between the modulus p and its bit-length ζ. The computational costs of the decryption can be easily evaluated in the same manner.

5. Key Generation

The key matrix should be randomly generated. It should be nonsingular to be invertible in the Galois field $GF(p)$ and it should satisfy the condition $\gcd(\det \mathbf{K} \ (\bmod\, p), p) = 1$ as it was stated in Section 2. Taking a matrix inversion over a Galois field may be a tedious task. The square matrix **R** is assumed as an inverse of the square matrix **K** if and only if $\mathbf{K.R} (\bmod\, p) = \mathbf{R.K} (\bmod\, p) = \mathbf{I}$ where **I** is an $n \times n$ identity matrix. The inverse key matrix can be calculated using the Gaussian elimination method [3]. The total number of operations for the Gauss-Jordan matrix inversion algorithm is of $O(n^3)$ [21]. However, a fast algorithm is presented in [7] that generates an $n \times n$ pseudo-random square matrix and its inverse over the Galois field $GF(p)$ with some simple and fast manipulations. First, **K** and **R** matrices are initialized with the identity matrix **I**. The algorithm then proceeds with the following loop:

For $q = 1 : n$ do

$$\mathbf{K} = \mathbf{K.H}_q (\bmod\, p) \ , \ \mathbf{R} = \mathbf{G}_q \mathbf{.R} (\bmod\, p) \qquad (12)$$

in which $\mathbf{H}_q$ is the q^{th} row elementary matrix and $\mathbf{G}_q$ is its corresponding inverse row elementary matrix [7]. One can use the former for encryption and the latter for decryption or vice versa.

6. Conclusions

In this paper, it is proved that Lin et al.'s scheme [9] that tried to strengthen the Hill cipher against the known-plaintext attack includes some flaws and is vulnerable to the chosen-ciphertext attack. A symmetric cryptosystem that is actually a variant of the Affine Hill cipher is also introduced. The introduced cryptosystem includes a ciphering core that has an outer structure similar to the Affine Hill cipher but its inner manipulations are different. Each block of data is actually encrypted using a different random number that is generated via employment of a chained HMAC. Two cryptographic protocols are also introduced. The first one is a two-pass protocol for which we have modified the Hughes key-exchange protocol, and includes an authentication step to thwart the man-in-the-middle attack. The other protocol is one-pass and is suitable when both of participants are not online. The proposed cryptosystem and its underlying protocols thwart the known-plaintext, chosen-ciphertext, chosen-plaintext, and man-in-the-middle attacks. The keyspace has been greatly enhanced since the modulus is a prime number. The ciphertext-only attack is also thwarted due to the increased keyspace of the cryptosystem.

7. References

[1] Hill LS. Cryptography in an Algebraic Alphabet. *American Mathematical Monthly* 1929; **36**: 306-312.

[2] Hill LS. Concerning Certain Linear Transformation Apparatus of Cryptography. *American Mathematical Monthly* 1931; **38**: 135-154.

[3] Konheim AG. *Computer Security and Cryptography*. John Wiley & Sons, 2007.

[4] Koblitz N. *A course in Number theory and Cryptography*. Springer-Verlag: New York, 1987; 64-74.

[5] Ismail IA, Amin M, Diab H. How to repair the Hill cipher. *Journal of Zhejiang University-Science A* 2006, **7**: 2022-2030.

[6] Stinson DR. *Cryptography Theory and Practice*. Chapman & Hall/CRC, 2006.

[7] Yeh YS, Wu TC, Chang CC, Yang WC. A New Cryptosystem Using Matrix Transformation. *25th IEEE International Carnahan Conference on Security Technology* 1991: 131-138.

[8] Saeednia S. How to Make the Hill Cipher Secure. *Cryptologia Journal* 2000; **24**: 353-360.

[9] Lin CH, Lee CY, Lee CY. Comments on Saeednia's improved scheme for the Hill cipher. *Journal of the Chinese institute of engineers* 2004; **27**: 743-746.

[10] Li C, Zhang D, Chen G. Cryptanalysis of an image encryption scheme based on the Hill cipher. *Journal of Zhejiang University - Science A* 2008; **9**: 1118-1123.

[11] Schneier B. *Applied cryptography: Protocols, Algorithms, and Source Code in C*. Second edition, John Wiley & Sons, 1996.

[12] Overbey J, Traves W, Wojdylo J. On the Keyspace of the Hill Cipher. *Cryptologia Journal* 2005; **29**: 59–72.

[13] Stallings W. *Cryptography and Network Security Principles and Practices*. Prentice Hall, 2006.

[14] Wang MY, Su CP, Huang CT, Wu CW. An HMAC Processor with Integrated SHA-1 and MD5 Algorithms. *IEEE Asia and South Pacific Design Automation Conference (ASP-DAC)* Jan. 2004; 456-458.

[15] Bellare M. New Proofs for NMAC and HMAC: Security without Collision-Resistance. *Advances in Cryptology – CRYPTO'06 (Lecture Notes in Computer Science)* 2006; **4117**: 602-619.

[16] Law L, Menezes A, Qu M, Solinas J, Vanstone S. An efficient Protocol for Authenticated Key Agreement. *Journal of Designs, Codes and Cryptography* 2003; **28**: 119-134.

[17] Pohlig SC, Hellman ME. An Improved Algorithm for Computing Logarithms in GF(p) and Its Cryptographic Significance. *IEEE Transactions on Information Theory* 1978; **24**: 106–111.

[18] Knuth DE. *The Art of Computer Programming*. Addison-Wesley: Massachusetts, 1981; 2: 1-33.

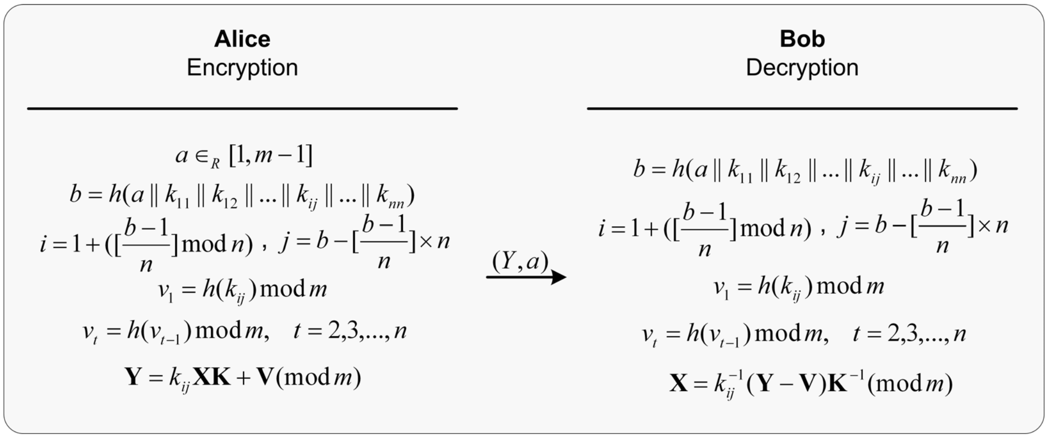
[19] Elkeelany O, Matalgah MM, Sheikh KP, Thaker M, Chaudhry G, Medhi D, Qaddour J. Performance Analysis of IPSec Protocol: Encryption and Authentication. *IEEE International Conference on Communications* 2002; **2**: 1164-1168.

[20] Rosen KH. *Elementary Number Theory and Its Applications.* Addison-Wesley, Massachusetts, Second edition, 1988.

[21] Matos GM, Neto HC. Memory Optimized Architecture for Efficient Gauss-Jordan Matrix Inversion. *3rd IEEE Southern Conference on Programmable Logic (SPL'07)* Feb. 2007; 33-38.

Table 1. Computational costs of different schemes for encryption/decryption of each block of data

Different Schemes	Operation	T_{Mul}	T_{Add}	T_{Inv}	T_{Hash}
Original Hill Cipher	Encryption	n^2	$n^2 - n$	-	-
	Decryption	n^2	$n^2 - n$	-	-
Affine Hill Cipher	Encryption	n^2	n^2	-	-
	Decryption	n^2	n^2	-	-
Lin et al.'s Scheme [9]	Encryption	$n^2 + n + 3$	$n^2 + 4$	-	$n + 1$
	Decryption	$n^2 + n + 3$	$n^2 + 4$	1	$n + 1$
Our Proposed Scheme	Encryption	$n^2 + 2n$	$n^2 + n + 1$	-	1
	Decryption	$n^2 + 2n$	$n^2 + n + 1$	1	1



Alice
Encryption

$$a \in_R [1, m-1]$$
$$b = h(a \| k_{11} \| k_{12} \| ... \| k_{ij} \| ... \| k_{nn})$$
$$i = 1 + ([\frac{b-1}{n}] \bmod n)\ ,\ j = b - [\frac{b-1}{n}] \times n$$
$$v_1 = h(k_{ij}) \bmod m$$
$$v_t = h(v_{t-1}) \bmod m, \quad t = 2,3,...,n$$
$$\mathbf{Y} = k_{ij}\mathbf{X}\mathbf{K} + \mathbf{V} (\bmod m)$$

$$\xrightarrow{(Y,a)}$$

Bob
Decryption

$$b = h(a \| k_{11} \| k_{12} \| ... \| k_{ij} \| ... \| k_{nn})$$
$$i = 1 + ([\frac{b-1}{n}] \bmod n)\ ,\ j = b - [\frac{b-1}{n}] \times n$$
$$v_1 = h(k_{ij}) \bmod m$$
$$v_t = h(v_{t-1}) \bmod m, \quad t = 2,3,...,n$$
$$\mathbf{X} = k_{ij}^{-1}(\mathbf{Y} - \mathbf{V})\mathbf{K}^{-1} (\bmod m)$$

Fig. 1. The corrected Lin *et al.*'s scheme [9]

Encryption

$$k' = (k_{11} \| k_{12} \| k_{13} \| \ldots \| k_{nn} \| a_{t-1}) \bmod 2^q$$

$$a_t = HMAC_{k'}(a_{t-1})$$

if $a_t \underset{p}{\not\equiv} 0$: $v_0 = a_t \pmod p$

if $a_t \underset{p}{\equiv} 0$: $v_0 = 1$

$$\begin{cases} j = (v_{i-1} \bmod n) + 1 \\ \tilde{v}_{i-1} = 2^{\lceil \gamma/2 \rceil} + (v_{i-1} \bmod 2^{\lceil \gamma/2 \rceil}) \\ v_i = k_{ij} + \tilde{v}_{i-1} a_t \pmod p \end{cases} \quad , \quad i = 1,\ldots,n$$

$$\mathbf{Y} = v_0 \mathbf{X}\mathbf{K} + \mathbf{V} \pmod p$$

Decryption

$$k' = (k_{11} \| k_{12} \| k_{13} \| \ldots \| k_{nn} \| a_{t-1}) \bmod 2^q$$

$$a_t = HMAC_{k'}(a_{t-1})$$

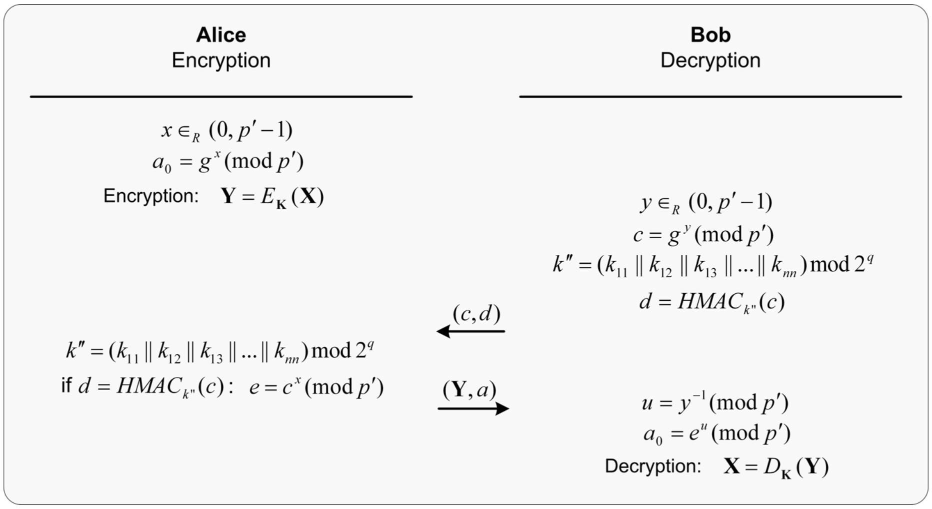
if $a_t \underset{p}{\not\equiv} 0$: $v_0 = a_t \pmod p$

if $a_t \underset{p}{\equiv} 0$: $v_0 = 1$

$$\begin{cases} j = (v_{i-1} \bmod n) + 1 \\ \tilde{v}_{i-1} = 2^{\lceil \gamma/2 \rceil} + (v_{i-1} \bmod 2^{\lceil \gamma/2 \rceil}) \\ v_i = k_{ij} + \tilde{v}_{i-1} a_t \pmod p \end{cases} \quad , \quad i = 1,\ldots,n$$

$$\mathbf{X} = v_0^{-1}(\mathbf{Y} - \mathbf{V})\mathbf{K}^{-1} \pmod p$$

Fig. 2. Encryption core of the proposed cryptosystem



Alice
Encryption

$x \in_R (0, p'-1)$

$a_0 = g^x \pmod{p'}$

Encryption: $\mathbf{Y} = E_{\mathbf{K}}(\mathbf{X})$

Bob
Decryption

$y \in_R (0, p'-1)$

$c = g^y \pmod{p'}$

$k'' = (k_{11} \| k_{12} \| k_{13} \| \ldots \| k_{nn}) \bmod 2^q$

$d = HMAC_{k''}(c)$

$\xleftarrow{(c,d)}$

$k'' = (k_{11} \| k_{12} \| k_{13} \| \ldots \| k_{nn}) \bmod 2^q$

if $d = HMAC_{k''}(c)$: $e = c^x \pmod{p'}$

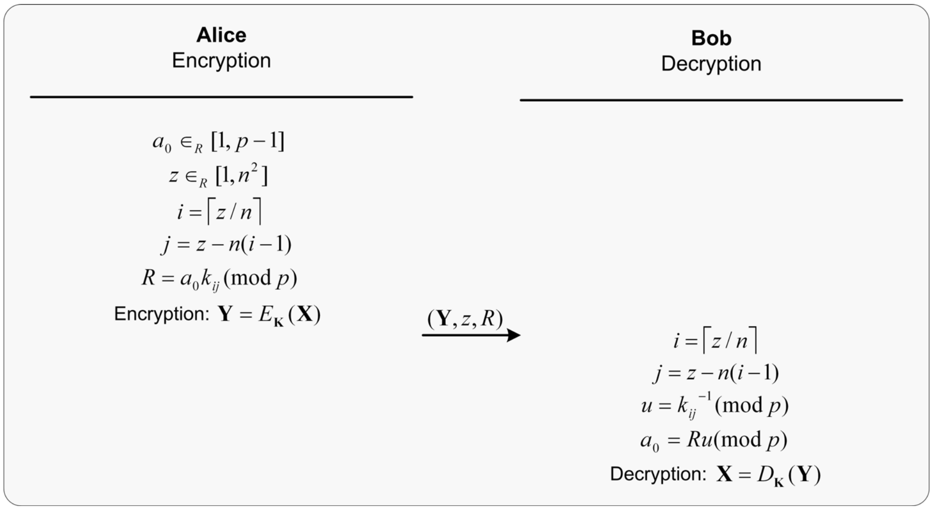
$\xrightarrow{(\mathbf{Y}, a)}$

$u = y^{-1} \pmod{p'}$

$a_0 = e^u \pmod{p'}$

Decryption: $\mathbf{X} = D_{\mathbf{K}}(\mathbf{Y})$

Fig. 3. A two-pass protocol for the proposed cryptosystem (Protocol 'A')



Alice
Encryption

$a_0 \in_R [1, p-1]$

$z \in_R [1, n^2]$

$i = \lceil z/n \rceil$

$j = z - n(i-1)$

$R = a_0 k_{ij} \pmod p$

Encryption: $\mathbf{Y} = E_{\mathbf{K}}(\mathbf{X})$

$\xrightarrow{(\mathbf{Y}, z, R)}$

Bob
Decryption

$i = \lceil z/n \rceil$

$j = z - n(i-1)$

$u = k_{ij}^{-1} \pmod p$

$a_0 = Ru \pmod p$

Decryption: $\mathbf{X} = D_{\mathbf{K}}(\mathbf{Y})$

Fig. 4. A one-pass protocol for the proposed cryptosystem (Protocol 'B')

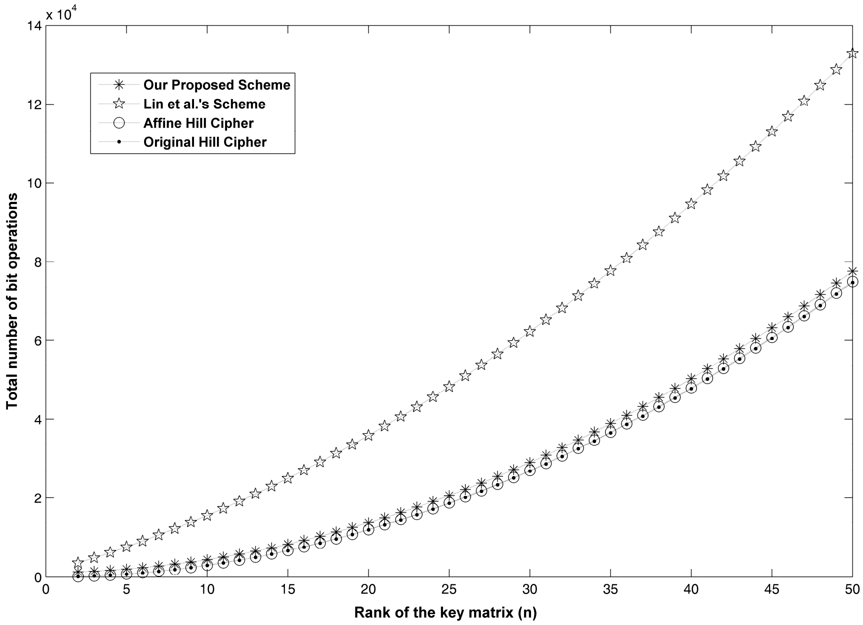


Fig. 5. Required number of operations for encrypting each block of data for different rank values when the modulus *p* is fixed (*p*=29)

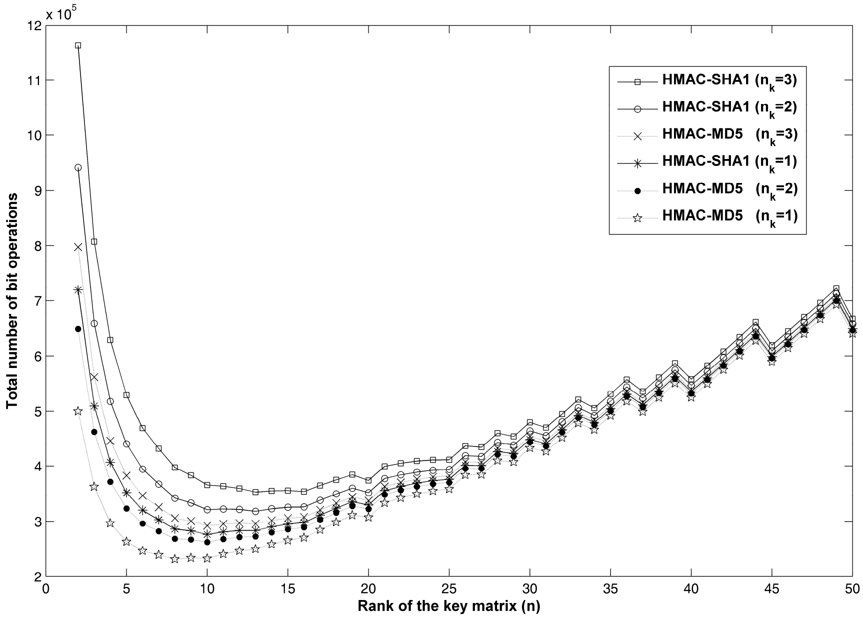


Fig. 6. Required number of operations for encrypting a plaintext of *L*=400 letters for different rank values when the modulus *p* is fixed (*p*=29)

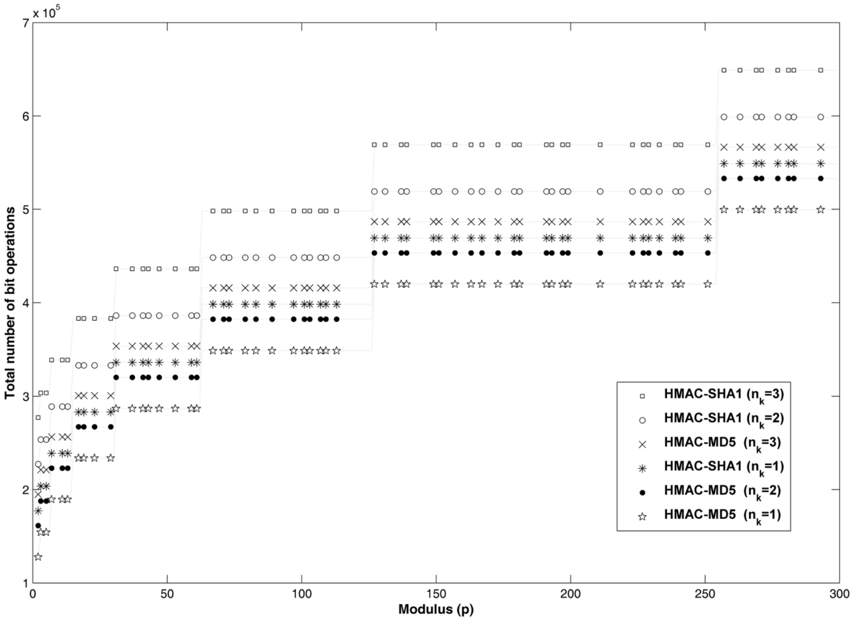


Fig. 7. Required number of operations for encrypting a plaintext of *L*=400 letters for different modulo *p* when the rank value *n* is fixed (*n*=9)