

Design of a Robust Cryptosystem Algorithm for Non-Invertible Matrices Based on Hill Cipher

Rushdi A. Hamamreh, Mousa Farajallah

Computer Engineering Department, Faculty Of Engineering, Al-Quds University

Summary

Information security is an important issue. Hill Cipher is one of the most famous symmetric cryptosystem that can be used to protect information from unauthorized access. Hill cipher is a polygraph substitution cipher based on linear algebra. Invented by Lester S. Hill in 1929, it was the first polygraph cipher which was practical to operate on more than three symbols at once, Hill Cipher has many advantages in data encryption. First, it is resistant to the frequency letter analysis. It's also very simple since it uses matrix multiplication. Finally, it has high speed and high throughput [1,13,14], However, noninvertible key matrix over Z_m is the main disadvantage of Hill Cipher, because few of the matrices have inverses over Z_m . This means that the encrypted text can't be decrypted [2]. This paper suggests a new technique in Hill Cipher algorithm to overcome its major problem-noninvertible key matrix. also our paper suggest enhancement the security of Hill Cipher against known plaintext attack because all steps in Hill Cipher depend on linear algebra calculation This is possible by using public key idea and key generating depending on various options and without linear algebra steps. Therefore, it will be difficult for the attacker to get the key. Finally, our paper will present a proper solution for the problem of sending secret key (K_S) for the first time. Our idea is using a public key public (PU) that doesn't sent but generated in the encryption and decryption sides.

Key words:

Plaintext, Ciphertext, Symmetric Encryption, Hill Cipher, known-plaintext attack, security.

1. Introduction

This paper try to enhance the security of Hill Cipher which has drawback from known-plaintext attack, when pair of plaintext and ciphertext known, since Hill Cipher depend on matrix multiplication and so when plaintext and ciphertext know the key matrix easy to calculate [3], to overcome this problem we need to use key for every encrypted block but this approach not secure since the key send every block of data, and the key has size quadratic refer to the size of the plaintext, so our approach is to generate new key not every message but every block of

plaintext, with steps has many possibilities of generation and have nonlinear algebra steps to make reverse of steps difficult, the first key used send by encrypted data using public RSA key then no key needed to send.

Also Hill Cipher become handicapped when the determinant of the key matrix not prime relative to the modular value that use, mean that more than half of the matrix not included in the Hill Cipher algorithm that means less variety of options for key matrix then less secure and more restricted modeling, using our new technique we can use any matrix for key included noninvertible matrix, Our paper organized as follows.

To use Hill Cipher we need secret key to be shared by the sender and the recipient, the problem is how to deliver the key for the first time of using, one of the options is used public key to send the secrete key, but still a problem of deliver the public key exist, our technique is used Diffie-Hellman key exchange not to exchange the secrete key but to generate the public key that used to send secrete key.

Section 2 differentiate symmetric and asymmetric cryptosystems, section3 describe the Hill Cipher algorithm and give example of its disadvantages, section 4 describe our method to overcame noninvertible key matrix problem, section 5 describe how sending secrete key using public key and introduce new procedure to overcome sending secrete key for the first time, section 6 will introduce our method for enhanced Hill Cipher security, the conclusion are in section 7.

2. Symmetric and Asymmetric Cryptosystems

The cryptosystems are divided into two parts first is the symmetric or private cryptosystem and the other type is the asymmetric or public key cryptosystem[12].

In symmetric cryptosystem the same key used in the sender and the recipient, which means the key used for encryption algorithm are also used for decryption algorithm [4].

Symmetric ciphers use substitution or transposition techniques or mixed of the two techniques, substitution map each plaintext elements into ciphertext elements, while transposition transpose the positions of plaintext elements [5].

Symmetric encryption algorithm not consuming too much computing power, so faster and simpler than asymmetric

encryption algorithm, also key generation more easy in symmetric encryption, but the problem is that key distribution in symmetric encryption, also the key are share for sender and recipient[11].

In asymmetric encryption the opposite of previous points, that means asymmetric encryption slower, more complicated, but not need to share the same key for sender and recipient, but more secure.

The decision to use symmetric or asymmetric encryption depends on the nature of use, it's desirable to use different encryption and decryption keys in some cases of cryptosystems, and also a combination of both is being used. The asymmetric keys are used for authentication and

after this have been successfully done, one or more symmetric keys are generated and exchanged using the asymmetric encryption [4-7].

3. Hill Cipher

The Hill cipher is a famous symmetric cryptosystem from the early days, which was invented by Lester S. Hill (1929; 1931), Hill cipher requires inverse of the key matrix while decryption. In fact that not all the matrices have an inverse and therefore they will not be eligible as key matrices in the Hill cipher scheme. Furthermore, due to its linear nature, the basic Hill cipher succumbs to known-plaintext attacks [4][9].

In Hill Cipher algorithm to encrypt plaintext block of size n , we need key matrix ($K_{n \times n}$) with entries are between $(0, p-1)$ included, but the determinant must be relatively prime to p , each entry in the plaintext block is between $(0, p-1)$ included each block of plaintext is then an n -dimensional vector x . We encrypt vector x simply to produce the ciphertext vector c using the following linear algebra equation.

$$c = k \times x \text{ mod } p.$$

To decrypt ciphertext vector c we need first to find the inverse matrix k^{-1} to k , where that matrix must be invertible over \mathbb{Z}_p , then can calculate x from the mathematical model.

$$x = k^{-1} \times c \text{ mod } p.$$

Using cryptosystem notation.

For encryption

$$C = E_k(P) \text{ mod } p = k \times P \text{ mod } p$$

For decryption

$$P = D_k(C) \text{ mod } p = k^{-1} \times C \text{ mod } p$$

To understand the Hill Cipher and its disadvantage of known-plaintext attack and not invertible key matrix let us consider the following simple example.

Example:

Let $p = 26$ and the plaintext message vector $x = \begin{bmatrix} 1 \\ 17 \end{bmatrix}$ the key matrix determinant must be prime relative to 26, let $k = \begin{bmatrix} 1 & 2 \\ 2 & 19 \end{bmatrix}$ the determinant of k is 15 and this number is prime relative to 26, and we can calculate the inverse of k that is

$$k^{-1} = \begin{bmatrix} 3 & 12 \\ 12 & 7 \end{bmatrix}$$

then the ciphertext is

$$c = k \times x \text{ mod } p$$

By substitute.

$$c = \begin{bmatrix} 1 & 2 \\ 2 & 19 \end{bmatrix} \times \begin{bmatrix} 1 \\ 17 \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} 35 \\ 325 \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} 9 \\ 13 \end{bmatrix}$$

In the recipient side the plaintext can be decrypted as following

$$x = \begin{bmatrix} 3 & 12 \\ 12 & 7 \end{bmatrix} \times \begin{bmatrix} 9 \\ 13 \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} 183 \\ 199 \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} 1 \\ 17 \end{bmatrix}.$$

Due to Hill Cipher linear nature, the basic Hill cipher easy broken by known-plaintext attacks to understand that let us consider the following example.

Example:

$$\text{Let } x_1 = \begin{bmatrix} 5 \\ 12 \end{bmatrix}, y_1 = \begin{bmatrix} 15 \\ 13 \end{bmatrix}, x_2 = \begin{bmatrix} 3 \\ 21 \end{bmatrix}, y_2 = \begin{bmatrix} 14 \\ 25 \end{bmatrix}$$

Where x_1 and x_2 the plaintext vectors and y_1, y_2 the ciphertext vectors.

Then we can write the following key equations:

$$\begin{bmatrix} x_{11} & x_{21} \\ x_{12} & x_{22} \end{bmatrix} \times \begin{bmatrix} k_{11} & k_{12} \\ k_{21} & k_{22} \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} y_{11} & y_{21} \\ y_{12} & y_{22} \end{bmatrix}$$

$$\begin{bmatrix} 5 & 3 \\ 12 & 21 \end{bmatrix} \times \begin{bmatrix} k_1 & k_2 \\ k_3 & k_4 \end{bmatrix} \text{ mod } 26 = \begin{bmatrix} 15 & 14 \\ 13 & 25 \end{bmatrix}$$

Then $k = \begin{bmatrix} 1 & 3 \\ 3 & 2 \end{bmatrix}$, here the key calculated from known-plaintext.

Since the inverse of the matrix used for encrypting the plaintext does not always exist, the encrypted text cannot be decrypted, also any ciphertext vector can be generated from two different plaintext vectors, so the question which of the two plaintext from generate the ciphertext, to understand this problem let us consider the following example.

Example:

To understand how the key will not be invertible over $p=26$ let $x_1 = \begin{bmatrix} 1 \\ 5 \end{bmatrix}, x_2 = \begin{bmatrix} 1 \\ 18 \end{bmatrix}, k = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix}$ then the two plaintext vectors are transformed into the same ciphertext vector $y = \begin{bmatrix} 11 \\ 23 \end{bmatrix}$.

This happened since the key matrix used has determinant of -2 and the $\text{mod}(-2, 26)=24$ has common factor with 26 that means the determinant of key matrix not prime relatively to 26.

4. Proposed method for noninvertible Key

The new method depend on the idea that on encryption side each plaintext char convert into two ciphertext chars, and in the decryption side each tow ciphertext chars convert into one plaintext char, in this way we can use any key matrix (invertible or not), also the key generation that always difficult when key not invertible is solved.

Encryption Steps:

- 1- Convert the plaintext char into numerical numbers.
- 2- Check if the determinant matrix is zero then add identity matrix else does nothing.
- 3- Calculate the column vector $c = k \times x$.
- 4- Calculate $c_1 = \text{fix}\left(\frac{c}{p}\right), c_2 = \text{mod}(c, p)$.
- 5- Convert the numerical numbers (c_1, c_2) into chars.

Decryption Steps:

- 1- Convert the two sequence of ciphertext into numerical numbers (y_1, y_2) .
- 2- Check if the determinant matrix is zero then add identity matrix else does nothing.
- 3- Calculate the column vector $p = \text{inv}(k) \times ((y_1 * 26) + y_2)$.
- 4- Convert the numerical numbers p into char.

5. Sending secret key securely

Assume we has sender X and recipient Y, if we use public key (PU_Y) and identifier (ID_Y) and sending it to the sender and the sender generate the first secret key (K_S) of Hill Cipher algorithm and encrypted K_S using PU_Y , then Y decrypted the message to recover K_S using private key (PR_Y), then the first secret key of Hill Cipher algorithm arrive in secure way this neglect the need of third party center to distribute the Hill Cipher secret key, and more secure of third party.

The problem if the attacker knows of this procedure and also knows the time of sending (PU_Y, ID_Y), let assume that the third party Z have PU_Z , also assume Z has ability to intercepts the first message from Y to X, then Z replace this message with (PU_Z, ID_Y), and send it to X, after X encrypts the K_S using PU_Z and sending it, again Z intercepts the replay message and decrypts the K_S using PR_Z and store K_S , then use PU_Y to encrypts K_S and sending it to Y, then Z know every message after this schema this protocol shown in figure 1.

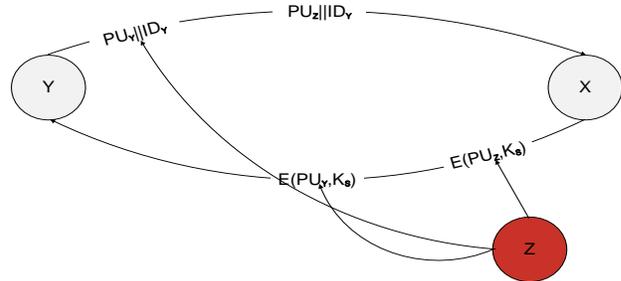


Figure1: Broken Public Key Encryption to Establish Secret Key protocol

To enhance the previous procedure, we distribute the secrete key using previous procedure and with confidentiality and Authentication to understand this procedure consider figure2.

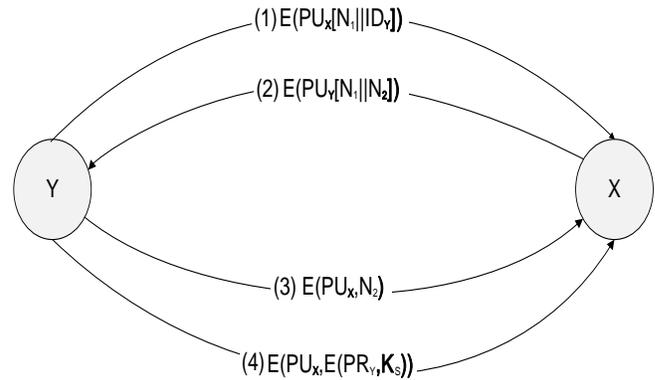


Figure2: Public Key Distribution of Secret Key

This method has problem if the third party Z generates PU_{Z1} and send it to Y instead of PU_X , also PU_{Z2} and send it to X instead of PU_Y , as in figure 3 the third party can be know the secret key.

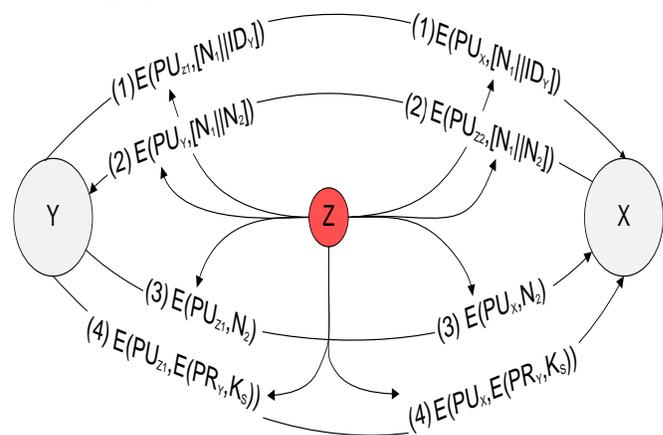


Figure3: Broken Public Key Distribution of Secret Key protocol

In this method also Z know K_S , although it is difficult to know the secret key in first procedure and in second procedure, however still this hacking technique mathematically and logically Possible [10].

Other power solution is by using Diffie-Hellman key exchange, but to exchange public-private generation key not to exchange secret key my technique are as following:

Sender select (q, α, X_p) where

q : Prime number.

α : $\alpha < q$ And α a primitive root of q .

Select any private integer X_p .

Calculate public integer X_U from the following equation:

$$X_U = (\alpha)^{X_p} \text{ mod } q.$$

Recipient selects (Y_p) and receive (q, α) from sender, to calculate public integer (Y_U) form private integer (Y_p) from the following equation:

$$Y_U = (\alpha)^{Y_p} \text{ mod } q.$$

And send Y_U to the sender, and then the public-private generator key will calculate in the two sides as following:

At the sender $K_{PU} = (Y_U)^{X_p} \text{ mod } q$.

At the recipient $K_{PU} = (X_U)^{Y_p} \text{ mod } q$

Consider figure4 to understand the previous steps of generating the public-private generator key.

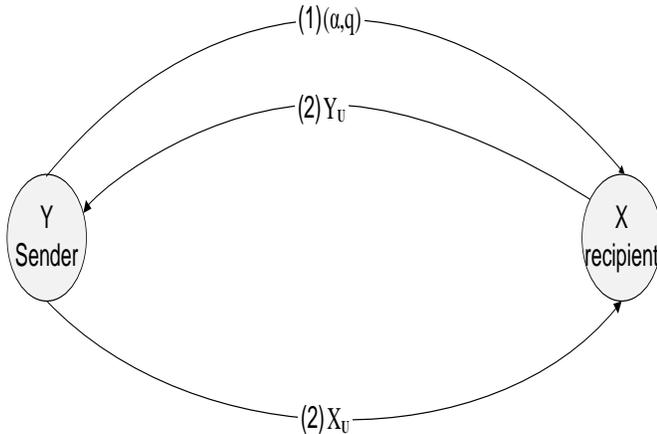


Figure4: Public Key Generation To Distribution of Secret Key

From this point each side start to calculate the pair of public and private key as following:

- 1- Calculate s that sum the digits of K_{PU} .
- 2- Select $p = \text{prime}_{\text{int}(\frac{2 \times s}{2})}$ less than K_{PU} .
- 3- Select $h = \text{prime}_{\text{int}(\frac{2 \times s}{3})}$ less than K_{PU} .
- 4- Calculate $n = p \times h$.
- 5- Calculate $\phi(n) = (h - 1) \times (p - 1)$.
- 6- Select integer e such that $\text{gcd}(\phi(n), e) = 1$ and e is $\text{int}(\frac{2 \times s}{3})$ succeeded number.
- 7- Calculate $d = (1 + i \times \phi(n)) / e$ where i start from 1 until the d become integer value.

Then the public key $\{e, n\}$ and the private key $\{d, n\}$ are in the two side.

Note this procedure in each step has options large enough and under each step new step with options also large enough, so the overall options are more than power of

passive or active attacker to broken the key, also the key is kept from third party attacking that happened in the two previous procedure, sine the public key not send from one side to other and also the generator public-private.

key not send but they calculated, also any step that has the term $\text{int}(\frac{\text{integer}_1 \times s}{\text{integer}_2})$ can be simplified or complicated by changing the values of integer_1 and integer_2 and depends on the importance of the transfer of data and the time to transfer these data and other constraints, note that these calculation need only one time over the transmission of large message.

Example:

Sender select $q = 131, \alpha = 2$ and $X_p = 21$

$$X_U = (2)^{21} \text{ mod } 131 = 104$$

Recipient select $Y_p = 60$

$$Y_U = (2)^{60} \text{ mod } 131 = 45$$

To calculate the public-private generator key:

$$\text{At the sender } K_{PU} = (45)^{21} \text{ mod } 131 = 39.$$

$$\text{At the recipient } K_{PU} = (104)^{60} \text{ mod } 131 = 39.$$

To calculate s :

$$s = 3 + 9 = 12.$$

$$p = 13.$$

$$h = 19.$$

$$n = 13 \times 19 = 247.$$

$$\phi(n) = 12 \times 18 = 216.$$

$$e = 25$$

Since

$$\text{gcd}(216, 5) =$$

$$\text{gcd}(216, 7) = \text{gcd}(216, 11) = \text{gcd}(216, 13) =$$

$$\text{gcd}(216, 17) = \text{gcd}(216, 19) = \text{gcd}(216, 23) =$$

$$\text{gcd}(216, 25) = 1$$

$$d = ((1 + 14 \times 216) / 25) = 121.$$

Then $PU = \{25, 247\}$ and $PR = \{121, 247\}$ at the sender and recipient.

6. Proposed method for Key generation

In this method we generate new key matrix every block of data by using the feedback approach, the first used key send by using RSA algorithm from the sender to the recipient, after generate public and private key at sender and recipient.

The plaintext vector has m characters and the key matrix has m columns vector, each entry of plaintext vector swapping with min value of corresponding row and Summation with the max value of corresponding row to produce the new key matrix.

This method first neglect the need of third party since the key used for encryption and decryption is

encrypted using RAS Public key of the recipient and send from sender to recipient.

Second when the key never send as is between sender and recipient or from third party the seriousness of discover the key also neglected, third advantages is when the attacker know the part of plaintext and the ciphertext he can't discover the key since the key change every block of data and the way of change have large possibilities, and haven't inverse.

Generating new key steps:

At the encryption side:

- 1- Send the key matrix using public key of recipient.
- 2- Using the key to encryption one block.
- 3- Using the block of data to swapping with min value of corresponding row and Summation with the max value of corresponding row to produce the new key matrix.
- 4- Checks if the determinant of new key zero then adds identity matrix for it, else do nothing.
- 5- Repeat from second to fourth step.

At the decryption side:

- 1- Receive and decrypt the key matrix using private key of recipient.
- 2- Using the key to decryption one block.
- 3- Using the block of data that decrypted to swapping with min value of corresponding row and Summation with the max value of corresponding row to produce the new key matrix.
- 4- Checks if the determinant of new key zero then adds identity matrix for it, else do nothing.
- 5- Repeat from second to fourth step.

All previous steps can be summarized in figure 5.

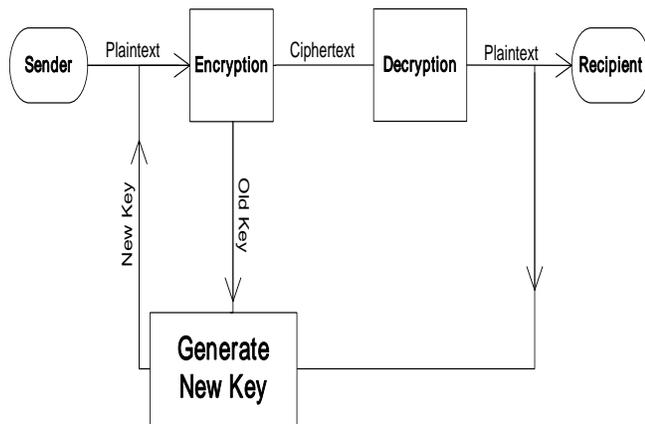


Figure5: Secret Key Generation

Complete algorithm

Now after the public-private generator key exchange and the first secret key securely transfer from sender to the recipient the tow problems of noninvertible key matrix and known-plaintext attack can be summarized using one encryption and one decryption algorithm as follow:

Algorithm For Encryption

```

{
  p1 = first block of plain text.
  k1 = first key matrix used.
  n = modular value
  l = number of plain text blocks.
  r = size(k1).
  for i = 1 To l Do
  {
    for j = 1 To r Do
    {
      pi = ToInt(pi)
      if(|ki| == 0)
      ki = ki + I(r)
      ci = ki × pi
      ci1 = int(ci/n)
      ci2 = mod(ci, n)
      ci1 = ToChar(ci1)
      ci2 = ToChar(ci2)
      if(i == 1)
        Send(ci1, ci2, enc_RSA(ki))
      else
        Send(ci1, ci2)
      end if
      min = min value of jth row in Key matrix
      max = max value of jth row in Key matrix
      s1 = column of min value
      s2 = column of max value
      ki+1(j,s1) = swap(pi, min)
      ki+1(j,s2) = mod(pi + max, n)
    }
  }
}

```

Algorithm For Decryption

```

{
  Receive ci1, ci2, n, enc_RSA(k1) ... .. i = 1 To l
  k1 = dec_RSA(k1)
  for i = 1 To l Do
  {
    for j = 1 To r Do
    {
      pi = ToInt(pi)
      if(|ki| == 0)
      ki = ki + I(r)
      d = |ki|
      k = adj(ki)
    }
  }
}

```

```

    pi =  $\left(\frac{1}{d}\right) \times k \times (n \times c_{i1} + c_{i2})$ 
    min = min value of jth row in Key matrix
    max = max value of jth row in Key matrix
    s1 = column of min value
    s2 = column of max value
    ki+1(j,s1) = swap(pi, min)
    ki+1(j,s2) = mod(pi + max, n)
  }
}
}

```

7. Conclusion

When the key matrix is sent for the first block of data encryption using public key at each side, sender and receiver, we drop the need of secure channel. We also overcome the problem of key distribution in symmetric encryption. And we avoid the danger of third party introduced in section 5.

Any matrix including noninvertible matrix over \mathbb{Z}_m can be used, as well as zero determinant matrix in modified hill cipher algorithm. Therefore, there will be no restriction on key generation or failure of choosing key matrix. And so, it will be very difficult for the attacker to get the key but easier to choose and generate the key in the algorithm.

We need to transmit $2 \times n$ characters in order to encrypt size n character data. But this is a simple problem compared with the problems solved in this paper, also comparing with very large capabilities of data transition in 2009.

The method of generating a new key is secure enough since the key changes every sent block. So the number of unknowns become more than the number of equation available to the attacker. Therefore, there is no mathematical solution for this system. In addition, the idea of generating a new key in each block has no unique inverse, because it swaps max and min unknown values with the old key values. And so, the attacker has no mathematical model to retrieve the key.

If the attacker will be able to break the modified hill cipher, he needs to follow the transmitted encrypted data from the very beginning till the end. He will also need other unlimited abilities.

References

[1] Ismail I.A, "How to repair the Hill cipher", Journal of Zhejiang University Science, 2006.
 [2] Bibhudendra Acharya "Novel Methods of Generating Self-Invertible Matrix for Hill Cipher Algorithm", National Institute of Technology Rourkela. 2006.

[3] Yi-Shiung Yeh, "A New Cryptosystem Using Matrix Transformation", IEEE. 1991.
 [4] Bibhudendra Acharya "Invertible, Involutory and Permutation Matrix Generation Methods for Hill Cipher System" IEEE International Conference on Advanced Computer Control, 2008.
 [5] Simmons, G. J, "Symmetric and Asymmetric Encryption", ACM Computing Surveys, 1979.
 [6] Omar Elkeelany, "Performance Comparisons, Design, and Implementation of RC5 Symmetric Encryption Core using Reconfigurable Hardware", Journal Of Computers, 2008.
 [7] Amit Parnerkar, "Secret key distribution protocol using public key cryptography", Rocky Mountain Conference. 2003.
 [8] S. Udaya Kumar, "An Iterative Process Involving Interlacing and Decompositions in the Development of a Block Cipher", International Journal of Computer Science and Network Security. 2006.
 [9] Chu-Hsing, "Comments on Saeednia's improved scheme for the Hill Cipher", Journal of the Chinese Institute of Engineers, 2004.
 [10] William Stallings, "Cryptography and Network Security Principles and Practices", Prentice Hall. 2006.
 [11] Gurpreet Dhillon, "Principles of Information Systems Security: Texts and Cases", John Wiley & Sons .2006.
 [12] Forouzan - Behrouz .A "Cryptography And Network Security", McGraw Hill. 2008.
 [13] Bibhudendra Acharya Sarat Kumar Patra , Ganapati Panda "Invertible, Involutory and Permutation Matrix Generation Methods for Hill Cipher System", IEEE International Conference on Advanced Computer Control, Jan 2009.
 [14] Adam J. Elbirt , Christof Paar "An Instruction-Level Distributed Processor for Symmetric-Key Cryptography", IEEE Transactions on Parallel and Distributed Systems, May 2005.