Cryptography By using linear Algebra

Asma Mustafa Abu Eddla
Department of Mathematics, Faculty of Science and Nature resources, Aljafara University
Asma81farg@gmail.com

Abstract

Information is a message which is received and understood. Information can be sent one person to another over a long range but the process of sending information must be done in a secure way especially in case of a private message. Mathematicians and Engineers have historically relied on different algorithmic techniques to secure messages and signals. One field in linear algebra's broad range of applications is Cryptography, the study of securely transmitting data over insecure channels. Instances of basic cryptography are evident throughout recorded history, seen in Da Vinci's notebooks, and used heavily for secure communications in war. As long as people have been able to write, there has been a need for communicating sensitive information. With advances in mathematic operations and computing power, the need, and also capability, of cryptography is constantly increasing. Today, Cryptography plays an important part in online services such as bank transactions, online currencies, and all sorts of services where secure transmission is necessary. In this paper, I present fundamentals of the field of cryptography that rely heavily on tools defined by linear algebra.

الملخص

المعلومات هي رسالة تم استلامها و فهمها. يمكن إرسال المعلومات من شخص إلى شخص أخر, و لكن يجب أن تتم عملية إرسال المعلومات بطريقة أمنة خاصة في حالة وجود رسالة خاصة. لقد اعتمد علماء الرياضيات و المهندسون تاريخيا على تقنيات حسابية مختلفة لتامين الرسائل و الإشارات. احد المجالات في مجموعة واسعة من تطبيقات الجبر الخطي هو التشفير, دراسة نقل البيانات بشكل امن عبر قنوات غير أمنة. أقدم حالات التشفير في التاريخ المسجل التي تظهر في دفاتر دافنشي, و تستخدم بكثافة للاتصالات الآمنة في الحرب.
طالما كان الناس قادرين على الكتابة, كانت هناك حاجة لتوصيل المعلومات الحساسة. مع التقدم في العمليات الرياضية و الحوسبة , تزداد الحاجة إلى التشفير.
يلعب التشفير اليوم دورا مهما في الخدمات عبر الانترنت مثل المعاملات المصرفية وجميع أنواع الخدمات التي يكون فيها النقل الأمن ضروريا.
في هدا البحث, أقدم أساسيات مجال التشفير الذي يعتمد على أدوات الجبر الخطي.

Keywords: Linear algebra formalism. encoding matrix . decoding matrix. Binary cipher matrix. Binary cryptography.

1 Introduction

Most people, is concerned with keeping communications private. Indeed, the protection of sensitive communications has been the emphasis of cryptography throughout much of its history. Encryption is the transformation of data into some unreadable form. Its purpose is to ensure privacy by keeping the information hidden from anyone for whom it is not intended, even those who can see the encrypted data. Decryption is the Encryption and decryption require the use of some secret information, usually referred to as a key. Depending on the encryption mechanism used, the same key might be used for both encryption and decryption, while for other mechanisms, the keys used for encryption and decryption might be different reverse of encryption; it is the transformation of encrypted data back into some intelligible form.

Today governments use sophisticated methods of coding and decoding messages. One type of code, which is extremely difficult to break, makes use of a large matrix to encode a message. The receiver of the message decodes it using the inverse of the matrix. This first matrix is called the **encoding matrix** and its inverse is called the **decoding matrix**.

2 CAESAR CIPHER

Until recently, encrypting secret messages was performed by hand using relatively trivial mechanisms to disguise information. One of the most well-known ciphers was named after Julius Caesar, namely, the Caesar cipher. The Caesar cipher is an example of a substitution cipher. Each letter of a given plaintext, the information to be encrypted, is substituted with another letter some given number of positions from it in the alphabet. For example, if we had an alphabet comprised of the standard 26 letters in the English alphabet and swapped each letter with the letter three places after it in the alphabet; we would have the following Caesar cipher.

3 Cipher text:

Using this cipher, the text “TOP SECRET MESSAGE” would encode to “WRS VHFUHW PHVVDJH.” One of the main problems with the Caesar cipher is that if an individual intercepts the cipher text and guesses that the Caesar cipher was used for

the encryption, he or she could easily go through the 25 shift values until they come upon a shift that decodes the cipher text into a meaningful plaintext. For example, if a substitution cipher encoded “e” to “h,” “h” would occur in the cipher text with the same frequency as “e” in the original language, allowing for a relatively simple analysis to break the substitution cipher.

4 Hill Cipher:

As time progressed, the study of cryptography continued to mature and, more recently, began to involve higher level mathematics. With this more advanced math came more advanced ciphers based on the idea of encryption and decryption keys. Encryption keys are a special value or set of values used in an encryption algorithm to convert a plaintext into a cipher text. A decryption key is the opposite. Decryption keys are used as part of a decryption algorithm to convert the cipher text back into the original plaintext. One such example of an encryption scheme that utilizes more advanced mathematics, as well as encryption and decryption keys is a cipher from 1929 called the Hill cipher. The Hill cipher is based on linear algebra and overcomes the frequency distribution problem of the Caesar cipher that was previously discussed [1].

4.1 About Hill ciphers Letter-by-letter substitution ciphers easily succumb to frequency analysis and so are notoriously unsecure. Polygraphic ciphers, by contrast, in which each list of n consecutive letters of the plaintext—an n-**graph**—is replaced by another n-graph according to some key, can be more challenging to break. The first systematic yet simple polygraphic ciphers using more than two letters per group are the Hill ciphers, first described by Lester Hill in 1929. For a polygraphic substitution, changing just one or two plaintext letters can completely change the corresponding ciphertext! That is one reason that Hill ciphers are so difficult to crack [3].

A **Hill n-cipher** works as follows. Start with an m-character **alphabet** and code each character with a unique integer in $\{0, 1, 2, \ldots, m-1\}$. The alphabet could consist of just the usual 26 letters in English or, as here, those letters supplemented

with the 3punctuation characters . and ? and _ (where _ denotes a blank space). For simplicity, the coding is done here by numbering the 29 characters in order as shown in Table 1. Next, choose as **key matrix** an $n \times n$ matrix A with entries in $\{0, 1, 2, \ldots, m-1\}$ that is invertible modulo m.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	.	?
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28

Table 1: Numerical coding of 29-letter alphabet

Given a ciphertext string, encode it to the corresponding vector **v** of integers in the set $\{0, 1, 2, \ldots, m-1\}$. If the length of **v** is not an integral multiple k of the key size n, then **pad v** as needed by repeating its last entry. Partition **v** into k column vectors v_j. Form each of the products Av_j; equivalently, form the matrix product $A[v_1 | v_2 | \ldots | v_k]$. Reduce the result modulo m. Reassemble the consecutive columns of this product into a new vector **w**. Finally, decode **w** into the corresponding ciphertext string.

The following Hill 3−cipher illustrates the procedure for our 29−character alphabet with its encoding given in Table 1. The key matrix is:

$$A = \begin{bmatrix} 17 & 5 & 20 \\ 23 & 9 & 3 \\ 11 & 2 & 12 \end{bmatrix}$$

The plaintext is the following 10-character message (including the space and period):

$$WANT \cup HELP.$$

First, encode the plaintext to the corresponding numbers:

22 0 13 19 28 7 4 11 15 26

Second, group these into trigraphs, repeating the final number twice to fill out the fourth group:

22 0 1319 28 7 4 11 15 26 26 26

Third, form the 3×4 matrix having these groups of three numbers as its columns. Fourth, apply the key:

$$A\begin{bmatrix} 22 & 19 & 4 & 26 \\ 0 & 28 & 11 & 26 \\ 13 & 7 & 15 & 26 \end{bmatrix} = \begin{bmatrix} 25 & 23 & 17 & 19 \\ 23 & 14 & 4 & 11 \\ 21 & 1 & 14 & 12 \end{bmatrix} \pmod{29}$$

Fifth, decode the numbers

25 23 21 23 14 1 17 4 14 19 11 12

from the unraveled columns of the last matrix to obtain the letters of the ciphertext :

Z X V X O B R E O T L M

A Hill cipher is relatively immune from attack if its key size n is large enough to preclude frequency analysis of n-graphs. But it is easy to crack if you have "captured" enough plaintext along with the corresponding ciphertext, for then the method of the following theorem applies.

Cracking Theorem. Suppose the alphabet length m is prime. Let p_1 , p_2,.....,p_n be n plaintext vectors for a Hill n-cipher having (unknown) key matrix A, and let c_1, c_2,....,c_n be the corresponding ciphertext vectors. Suppose these plaintext vectors are linearly independent over Z_m. Form the matrices $P = [P_1|P_2|....|P_n\ ]$ and $C = [C_1|C_2|..|C_n\ ]$ having the plaintext vectors and ciphertext vectors, respectively, as their columns. Then the same sequence of elementary row operations that reduces C^T to the identity matrix reduces P^T to the transpose $\left(A^{-1}\right)^T$ of the inverse key matrix A^{-1} [2].

5 Block Ciphers

Plaintext is divided into blocks of fixed length and every block is encrypted one at a time.

A block cipher is a set of 'code books' and every key produce a different code book. The encryption of a plaintext block is the corresponding ciphertext block entry in the code book. There are several methods to encrypt *M*, which is referred to as *block-cipher modes* of operations. Standard block-cipher modes of operations:

- Electronic Code Book mode (ECB)
- Cipher Block Chaining mode (CBC)

• Electronic Code Book Mode (ECB)

There are innumerable methods of increasing the security of an encryption algorithm. The method of encoding that the basic Hill cipher uses, such that each unique plaintext input corresponds to a unique ciphertext output, and each input is

encrypted independently from all others, is known as the Electronic Code Book (ECB) method. Let *Ci* be the *i*–th ciphertext block: ECB is often used to encrypt short plaintext messages. However, if we break up our string into blocks, there could be a chance that two blocks are identical: $Mi = Mj$ $(i \neq j)$. This provides the attacker with some information about the encryption.

Enacting the example cipher from above string “EASYTOBREAK” results in the ciphertext “XYXSCYVTXYAD” including the calculated appended character, while the slightly modified string “EASYTOCREAK” results in the ciphertext “XYXSCYKXXYBI”.

• Cipher Block Chaining (CBC)

A way to increase the difficulty of breaking a cipher is to use Cipher Block Chaining (CBC) as a method of encryption instead of ECB. Instead of having a direct map to from plaintext to ciphertext as in the ECB, CBC saves the ciphertext value of previous encryptions, and enacts a function involving that value on the plaintext before encrypting it with the main cipher. This method increases security in two ways. Using this method adds a level of abstraction to the encryption process, making it harder to crack the cipher, as well as introduces a method of validation for all messages. To illustrate CBC, I will extend the Hill cipher example used previously. The way a computer fundamentally deals with numbers is in binary notation, meaning each digit is either a 0 or a 1. For a CBC, the function often used between the previous ciphertext and the current plaintext is a bitwise XOR. A bitwise XOR iterates through binary strings, comparing each significant digit, and returning 1 if there are an odd number of 1’s between the two, and 0 otherwise.

Say I want to encrypt the plaintext string “ALWAYSBETTER”, referred to as p1, using the ciphertext string “XYXSCYVTXYAD” from earlier, referred to as c0, as the block. Next, I convert p1 to its integer form, which is “$1\ 12\ 23\ 1\ 25\ 19\ 2\ 5\ 20\ 20\ 5\ 18$”, and recall the integer form of c0, which is “$24\ 25\ 24\ 29\ 3\ 25\ 22\ 20\ 24\ 25\ 1\ 4$”.

For this cipher, the largest integer value we need to represent in binary is 26. Recall that each binary digit represents a power of 2. Representing a decimal number n in binary involves a linear combination of m powers of 2, where $n < 2^m$. Solve this

relation for m and add a ceiling operator to establish the following equation for determining m, the

number of bits needed to encode a set of n values. m = $\lceil \log_2 m \rceil$

Using this equation, it is clear that only need 5 bits to represent $1-26$ in binary. Determining each number's representation is a simple greedy algorithm, which subtracts each power of 2 from the desired number. If the subtraction is greater than or equal to 0, then perform it and set that bit equal to 1. Otherwise, do not perform the subtraction and set that bit equal to 0. Calculate and compare each significant digit of c0 and p1.

c0 = 11000 11001 11000 10011 00011 11001 10110 10100 11000 11001 00001 00100

XOR

p1 = 00001 01100 10111 00001 11001 10011 00010 00101 10100 10100 00101 10011

The result of this calculation will be referred to as p1' and is shown below

p1'= 11001 10101 01111 10010 11010 01010 10100 10001 01100 01101 00100 10111

Now convert this binary string to back to decimal and get the integer string "25 21 15 18 26 10 20 17 12 13 4 23". Then, assemble a 3×4 plaintext matrix and multiply it by the key matrix.

$$c1 = k.\ p1 = \begin{bmatrix} 3 & 3 & 4 \\ 0 & 1 & 1 \\ 4 & 3 & 4 \end{bmatrix} \begin{bmatrix} 25 & 18 & 20 & 13 \\ 21 & 26 & 17 & 4 \\ 15 & 10 & 12 & 23 \end{bmatrix} = \begin{bmatrix} 198 & 172 & 159 & 143 \\ 36 & 36 & 29 & 27 \\ 223 & 190 & 179 & 156 \end{bmatrix}$$

This sort of transformation seems as though it would be one way, and an inverse function to decrypt the text could not be found. However, the bitwise XOR function is actually its own inverse, meaning applying the function to a string twice will return the original string. The recipient first applies the inverse matrix key to the ciphertext c1 to get p1' back.

$$p1' = k^{-1}.c1 = \begin{bmatrix} -1 & 0 & 1 \\ -4 & 4 & 3 \\ 4 & -3 & -3 \end{bmatrix} \begin{bmatrix} 198 & 172 & 159 & 143 \\ 36 & 36 & 29 & 27 \\ 223 & 190 & 179 & 156 \end{bmatrix} = \begin{bmatrix} 25 & 18 & 20 & 13 \\ 21 & 26 & 17 & 4 \\ 15 & 10 & 12 & 23 \end{bmatrix}$$

We then perform a bitwise XOR on c0 and p1'

c0 = 11000 11001 11000 10011 00011 11001 10110 10100 11000 11001 00001 00100

XOR

p1'= 11001 10101 01111 10010 11010 01010 10100 10001 01100 01101 00100 10111

and get back

p1 = 00001 01100 10111 00001 11001 10011 00010 00101 10100 10100 00101 10011

This is the exact bit string that was sent earlier! Which is “1 12 23 1 25 19 2 5 20 20 5 18”, Since the ciphertext string c0 is transmitted publicly over the insecure channel, every intended recipient knows the value they need to use for the bitwise operation. While only a marginal increase in security, CBC allows an encrypted storage of message history, which is computationally infeasible to edit [1] .

6 The proposed BME method: Materials and methods

This section will focus on the encryption of a binary sequence of two bytes at a time, so this implementation of the proposed method is destined to cipher each sequence of sixteen bits into an encrypted binary sequence of the same size.

6.1 The binary matrix encryption model

Suppose that b_0 is a a16-bit sequence in column vector disposal, extracted from the original data. Therefore, an encrypted sequence b_{enc} can be obtained by multiplying a binary cipher matrix **C** by b_0, such as

$$b_{enc} = cb_0 \qquad (1)$$

Some requirements or rules are needed for setting the binary matrix **C**:
(i) The matrix must have sixteen columns or its matrix multiplication with b_0 could not be performed.
(ii) The columns of **C** must be a linear independent set of vectors, otherwise the linear transformation performed by **C** would not be “one-to-one” [3, 16] and, therefore, different vectors b_0 would generate the same b_{enc}, leading to confusion on the decryption process. This is the uniqueness requirement for the recovering of the original bytes.

(iii) In order to preserve the data size, the encrypted sequence b_{enc} must also have the same size of b_0, in this case, 16 bits. Furthermore, b_{enc} cannot have fewer bits than b_0, otherwise the requirement (ii) for **C** could not be fulfilled. Therefore, **C** must have strictly 16 rows in order to generate a 16-bit sequence for b_{enc}. To conform to the linear algebra semantics, this is denoted here as the $T : B^{16} \rightarrow B^{16}$ requirement, where **T** represents a linear transformation and B^{16} is the sixteen-dimensional binary vector space. Some caution must be directed to use of the term “binary vector space”, since it not obeys all properties of a usual vector space. One of them is that a binary vector space is not infinite, given that B^{16}, for instance, has “only” $2^{16} = 65536$ vectors (different sequences of 16 bits). This justifies the next restriction for the cipher matrix.

(iv) Each row of **C** must not have more than one element with the value "1", or some results for b_{enc} would not be a binary vector. The in-depth meaning of this rule will be explained later in this section.

Therefore, based on the above requirements, the encrypted matrix **C** is square of order sixteen (the number of bits in b_0) and must formed by a scramble of the identity matrix columns. Suppose that **I** is the identity matrix of order16, where

$$I = [I_1 \ I_2 \ I_3 \ I_4 \ I_5 \ I_6 \ I_7 \ I_8 \ I_9 \ I_{10} \ I_{11} \ I_{12} \ I_{13} \ I_{14} \ I_{15} \ I_{16}] \qquad (2)$$

such as I_n (n = 1, 2,…, 16) is the n^{th} column of **I**. The column I_n has sixteen elements (one per row), only a element "1" at its n^{th} row and all the remaining elements are "0". The cipher matrix can be obtained by a scramble of the columns of **I**, for example

$$C = [I_5 \ I_7 \ I_{15} \ I_{13} \ I_3 \ I_2 \ I_8 \ I_4 \ I_{16} \ I_{11} \ I_{14} \ I_9 \ I_{10} \ I_1 \ I_{12} \ I_6] \qquad (3)$$

In this case, the columns order "|5|7|15|13|3|2|8|4|16|11|14|9|10|1|12|6|" can be considered as the key for the encryption/decryption process using this cipher matrix. The total of possible permutations or keys is given by 16! = 20922789888000, which is a huge number of keys to be considered by a potential spy, presuming that he knows about the encryption method being used. Some permutations will not be efficient as an encryption key. If **C** = **I**, for example, (1) shows that the encrypted byte will be the same as the original byte and there will not be any encryption at all for the key "|1|2|3|4|5|6|7|8|9|10|11|12|13|14|15|16|". The higher the shuffling of columns of the identity matrix, the stronger the encryption will be.

The elements of the matrix **C** = [c_{ij}] are not bits, but instead c_{ij} represents a binary transmission coefficient. If $c_{ij} = 1$, then the j^{th} bit in the original binary sequence (b_0) is transferred to the i^{th} position of the encrypted sequence (b_{enc}). This is another interpretation of why there cannot be more than an element "1" in a same row of **C**, otherwise two distinct bits in b_0 would be directed into the same position in b_{enc}, and this cannot be expressed as a binary output. Furthermore, two or more bits in the same column of **C** is also not advisable, because the same bit in b_0 would be transferred into two or more distinct positions in b_{enc}, and the remaining bits in b_0 would have less positions to occupy in the encrypted bit b_{enc}, causing information loss or the need to vainly increase the size of b_{enc} (and the number of lines of **C**) to make room for the redundant bits. However, this would violate the rule (iii), since the size of the encrypted sequence would be higher than the size of the original sequence. Therefore, a encipher matrix **C** formed by a scramble of the identity matrix columns is the only way to grant that all exposed restrictions are fulfilled in order that no bit redundancy or data loss will undermine the decryption process of the encrypted bit sequence. There is not also a miraculous procedure to reduce the

size of the encrypted data at the encryption process without bit loss. It can only be achieved if one or more bits of the original sequence are not relevant and then can be discarded.

We obtain the decryption equation by left multiplying each side of (1) by

the inverse of **C**, which leads to

$$b_{dec} = c^{-1} b_{enc} \qquad (4)$$

where is expected that $b_{dec} = b_0$ after decryption. The use of the inverse of **C** provides another way to explain that **C** must be square with its columns forming a linear independent set of vectors, otherwise **C** would not be invertible.
The inverse linear transform performed by $C^{-1} = [d_{ij}]$ will relocate each bit to its start relocate each bit to its start relocate each bit to its start
Therefore, $d_{ij} = c_{ij}$ and in this case the property $C^{-1} = C^T$ is valid, where C^T is the transpose of **C**. This occurs because the established cipher matrix is an orthogonal matrix , with its rows and columns formed
by orthogonal unit vectors. to The decipher matrix related the example in (3) is described by (5). Figure 1 shows a block diagram describing the encryption and decryption processes via the proposed BME method.

$$C^{-1} = C^T = [I_{14}\ I_6\ I_5\ I_8\ I_1\ I_{16}\ I_2\ I_7\ I_{12}\ I_{13}\ I_{10}\ I_{15}\ I_4\ I_{11}\ I_3\ I_9] \qquad (5)$$

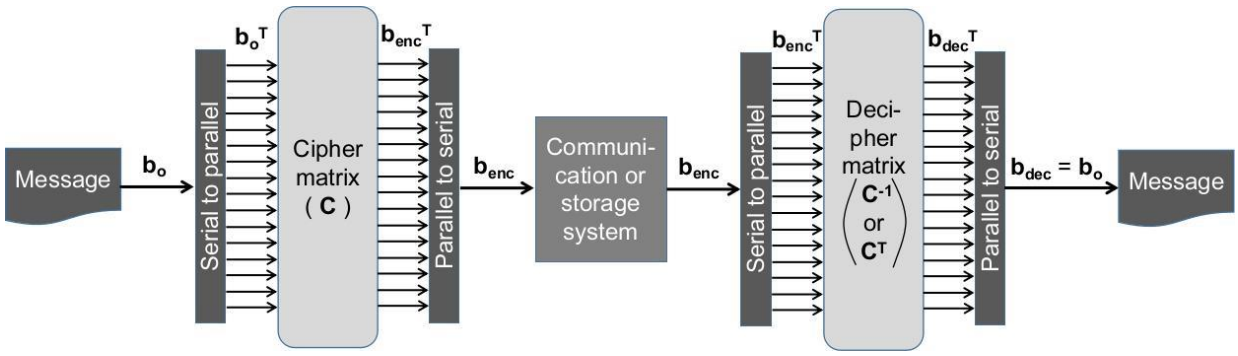

Figure 1: Block diagram of the proposed BME method

6.2 Security aspects

The proposed BME method has not the same drawback of the XOR cipher techniques. With the BME it is very difficult to retrieve the key from repetitions of a given sequence of encrypted bits. First, suppose a 16-bit sequence with only zeros

(**0**). The encrypted sequence will have also only zeros since it obeys the property **T**(**0**) = **0** inherent in all linear transformations. Therefore, this property is valid for all possible BME keys, because each key represents a cipher binary matrix that performs a linear transformation. On the other hand, the XOR ciphering is not a linear transformation and **T**(**0**) will directly give the key used for encryption to an external observer.

Suppose now that b_0 is a usual binary sequence appearing repeatedly at many points of the original data. As commented before, the total number of keys in the proposed 16-bit BME method is 16! = 20922789888000. However, the total number of possible 16-bit sequences is 216= 65536, which means that many different keys or linear transformations will generate the same encrypted sequence from b_0. In contrast, the XOR cipher techniques using a stream of 16 bits only have also a total of 65536 keys and each key will generate a different encrypted sequence from b_0. If a spy identifies the repeated encrypted sequence by a spectral analysis and relates them to b_0, he can easily retrieve the key and decipher the entire message. For this reason, XOR cipher techniques use bit streams as long as the message or other strategies are added in order to increase security.

Regarding 16-bit sequences, Figure 2 compares the number of ways to generate a specific encrypted sequence (b_{enc}) from a given original sequence (b_0), considering the two cipher techniques. The number of BME keys that perform the same encryption for the same pair of original and encrypted sequence depends on the number of bits “0” and “1” that compose these sequences, which is shown in Table

1, where Num(“0”) and Num(“1”) are the number of occurrences of each binary value within these sequences. With the XOR cipher technique, only one key can relate the two sequences at any case. From Table 1 we can conclude that the example shown at Figure 2 is the worst case for the BME technique, i.e., when the occurrence numbers of each binary value are equal, a spy has the best yet very small chance to discover the key [4] .

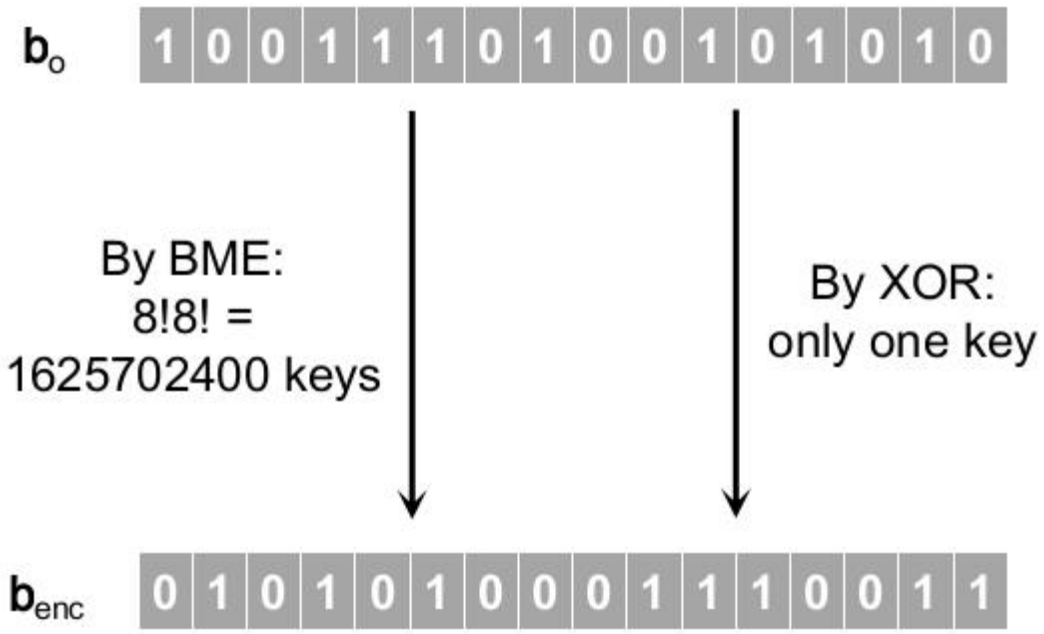

Figure 2: A comparison between BME and XOR cipher techniques, showing the number of keys that can generate a specific 16-bit encrypted sequence (b_{enc}) from a given 16-bit original sequence (b_0).

Table 2 : Number of BME keys that generates a specific b_{enc} from a specific b_0

Num("0")	Num("1")	Number of BME keys
16	0	16!0! = 20922789888000 (all keys)
15	1	15!1! = 1307674368000
14	2	14!2! = 174356582400
13	3	13!3! = 37362124800
12	4	12!4! = 11496038400
11	5	11!5! = 4790016000
10	6	10!6! = 2612736000
9	7	9!7! = 1828915200
8	8	8!8! = 1625702400 (see Figure 2)
7	9	7!9! = 1828915200
6	10	6!10! = 2612736000
5	11	5!11! = 4790016000
4	12	4!12! = 11496038400
3	13	3!13! = 37362124800
2	14	2!14! = 174356582400
1	15	1!15! = 1307674368000
0	16	0!16! = 20922789888000 (all keys)

7 Modern symmetric cryptosystems: DES and AES

Modern symmetric ciphers such as the Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) are based on ad hoc mixing operations, rather than on intractable mathematical problems used by asymmetric ciphers. The reason that DES and AES and other symmetric ciphers are used in practice is that they are much faster than asymmetric ciphers. Thus if Alice wants to send Bob a long message, she first uses an asymmetric cipher such as RSA to send Bob a key for a symmetric cipher, and then she uses a symmetric cipher such as DES or AES to send the actual data.

DES was created by a team of cryptographers at IBM in the early 1970s, and with some modifications suggested by the United States National Security Agency (NSA), it was officially adopted in 1977 as a government standard suitable for use in commercial applications.

DES uses a 56-bit private key and encrypts blocks of 64 bits at a time. Most of DES's mixing operations are linear, with the only nonlinear component being the use of eight *S*-boxes (substitution boxes). Each S-box is a look-up table in which six input bits are replaced by four output bits. Figure 3 illustrates one of the S-boxes used by DES.

	0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
0	14 4 13 1 2 15 11 8 3 10 6 12 5 9 0 7
1	0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8
2	4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0
3	15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13

Figure 3: The first of eight S-boxes used by DES

Here is how an S-box is used. The input is a list of six bits, say

Input = $\beta_1\beta_2\beta_3\beta_4\beta_5\beta_6$.

First use the 2-bit binary number $\beta_1\beta_6$, which is a number between 0 and 3, to choose the row of the S-box, and then use the 4-bit binary number $\beta_2\beta_3\beta_4\beta_5$, which is a number between 0 and 15, to choose the column of the S-box. The

output is the entry of the S-box for the chosen row and column. This entry, which is between 0 and 15, is converted into a 4-bit number.

For example, suppose that the input string is '110010'. Binary '10' is 2, so we use row 2, and binary '1001' is 9, so we use column 9. The entry of the S-box in Figure 3 for row 2 and column 9 is 12, which we convert to binary '1100'.

The S-boxes were designed to prevent various sorts of attacks, including especially an attack called differential cryptanalysis, which was known to IBM and the NSA in the 1970s, but published only after its rediscovery by Biham and Shamir in the 1980s. Differential cryptanalysis and other non-brute-force attacks are somewhat impractical because they require knowledge of a large number ($>2^{40}$) of plaintext/ciphertext pairs.

A more serious flaw of DES is its comparatively short 56-bit key. As computer hardware became increasingly fast and inexpensive and computing power more distributed in the 1990s, it became feasible to break DES by a brute-force search of all possible keys, either using many machines over the Internet or building a dedicated DES cracking machine. Comparatively inexpensive machines now exist that are capable of breaking a DES key in less than a week.

One solution to this problem, which has been widely adopted, is to use DES multiple times. There are a number of different versions of *Triple DES*, the simplest of which is to simply encrypt the plaintext three times using three different keys. Thus if we write DES(k,m) for the DES encryption of the message m using the key k, then one version of triple DES is

TDES(k_1, k_2, k_3, m) = DES(k_3, DES(k_2, DES(k_1,m))).

A variation replaces the middle DES encryption by a DES decryption; this has the effect that setting $k_1 = k_2 = k_3 = k$ yields ordinary DES encryption. Another variation, used by the electronics payment industry, takes $k_1 = k_3$, which reduces key size at the cost of some security reduction. Finally, since three DES encryptions triple the encryption time, another version called DES-X uses a single DES encryption combined with initial and final XOR operations with two 64-bit keys. Thus DES-X looks like

DESX(k_1, k_2, k_3, m) = k_3 xor DES(k_2, m xor k_1).

Although DES and its variants were widely deployed, it suffers from short and inflexible key and block sizes. Further, although DES is fast when implemented in specialized hardware, it is comparatively slow in software. So in 1997 the United

States National Institute of Standards (NIST) organized an open competition to choose a replacement for DES. There were many submissions, and after several years of analysis and several international conferences devoted to the selection process, NIST announced in 2000 that the Rijndael cipher, invented by the Belgian cryptographers J. Daemen and V. Rijmen, had been chosen as AES. Since that time AES has been widely adopted, although variants of DES are still in use.[13]
AES is a block cipher in which the plaintext–ciphertext blocks are 128 bits in length and the key size may be 128, 192, or 256 bits. AES is similar to DES in that it encrypts and decrypts by repeating a basic operation several times [5] [6].

8 Public Key Cryptography

Recall that in symmetric key cryptography each communicating party needed to have a copy of the same secret key. This led to a very difficult key management problem. In public key cryptography we replace the use of identical keys with two keys, one public and one private.

The public key can be published in a directory along with the user's name. Anyone who then wishes to send a message to the holder of the associated private key will take the public key, encrypt a message under it and send it to the owner of the corresponding private key. The idea is that only the holder of the private key will be able to decrypt the message. More clearly, we have the transforms

Message + Alice's public key = Ciphertext,

Ciphertext + Alice's private key = Message.

Hence anyone with Alice's public key can send Alice a secret message. But only Alice can decrypt the message, since only Alice has the corresponding private key. Public key systems work because the two keys are linked in a mathematical way, such that knowing the public key tells you nothing about the private key. But knowing the private key allows you to unlock information encrypted with the public key. This may seem strange, and will require some thought and patience to understand. The concept was so strange it was not until 1976 that anyone thought of it. The idea was first presented in the seminal paper of Diffie and Hellman entitled New Directions in Cryptography. Although Diffie and Hellman invented the concept of public key cryptography it was not until a year or so later that the first (and most successful) system, namely RSA, was invented [7].

8.1 Principles of Public-Key Cryptosystems

Asymmetric algorithms rely on one key for encryption and a different but related key for decryption.

These algorithms have the following important characteristic:

- It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

In addition, some algorithms, such as RSA, also exhibit the following characteristic:

- Either of the two related keys can be used for encryption, with the other used for decryption.

A public-key encryption scheme has six ingredients (Figure 5 a; compare with Figure 4):

- **Plaintext:** This is the readable message or data that is fed into the algorithm as input.
- **Encryption algorithm:** The encryption algorithm performs various transformations on the plaintext.
- **Public and private keys:** This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
- **Ciphertext:** This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
- **Decryption algorithm:** This algorithm accepts the ciphertext and the matching key and produces the original plaintext

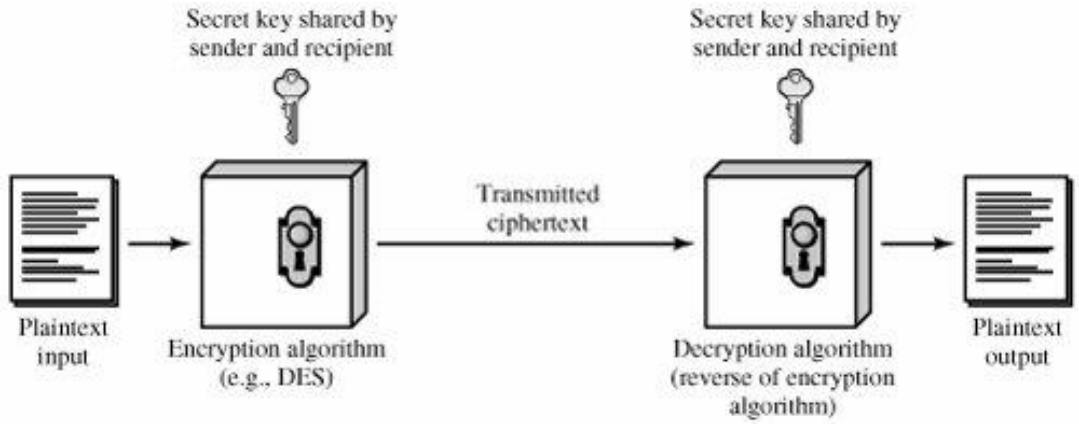

Figure 4. Simplified Model of Conventional Encryption

The essential steps are the following:

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.

2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. As Figure 5 a suggests, each user maintains a collection of public keys obtained from others.

3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.

4. When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

With this approach, all participants have access to public keys, and private keys are generated locally by each participant and therefore need never be distributed. As long as a user's private key remains Protected and secret, incoming communication is secure. At any time, a system can change its private key and publish the companion public key to replace its old public key.

Table 3 summarizes some of the important aspects of symmetric and public-key encryption. To discriminate between the two, we refer to the key used in symmetric Encryption as a **secret key**. The two keys used for asymmetric encryption are referred to as the **public key** and the **private key**.

Invariably, the private key is kept secret, but it is referred to as a private key rather than a secret key to avoid confusion with symmetric encryption [8].

9. CONCLUSION

If two people who wish to communicate have access to a secure channel over which they can share the key, then this is easy to do, but ultimately, they could just share their message over that channel. Also, in today's age of global communication, there is almost no such thing as a secure channel. A major issue in cryptography is how to share a key over an insecure channel. An encompassing term for the method most commonly used to deal with this issue is public-key cryptography, in which every user has access to the key, which encrypts the data. Typically, this key is a special function in which it is very easy to calculate one way, making it trivial to

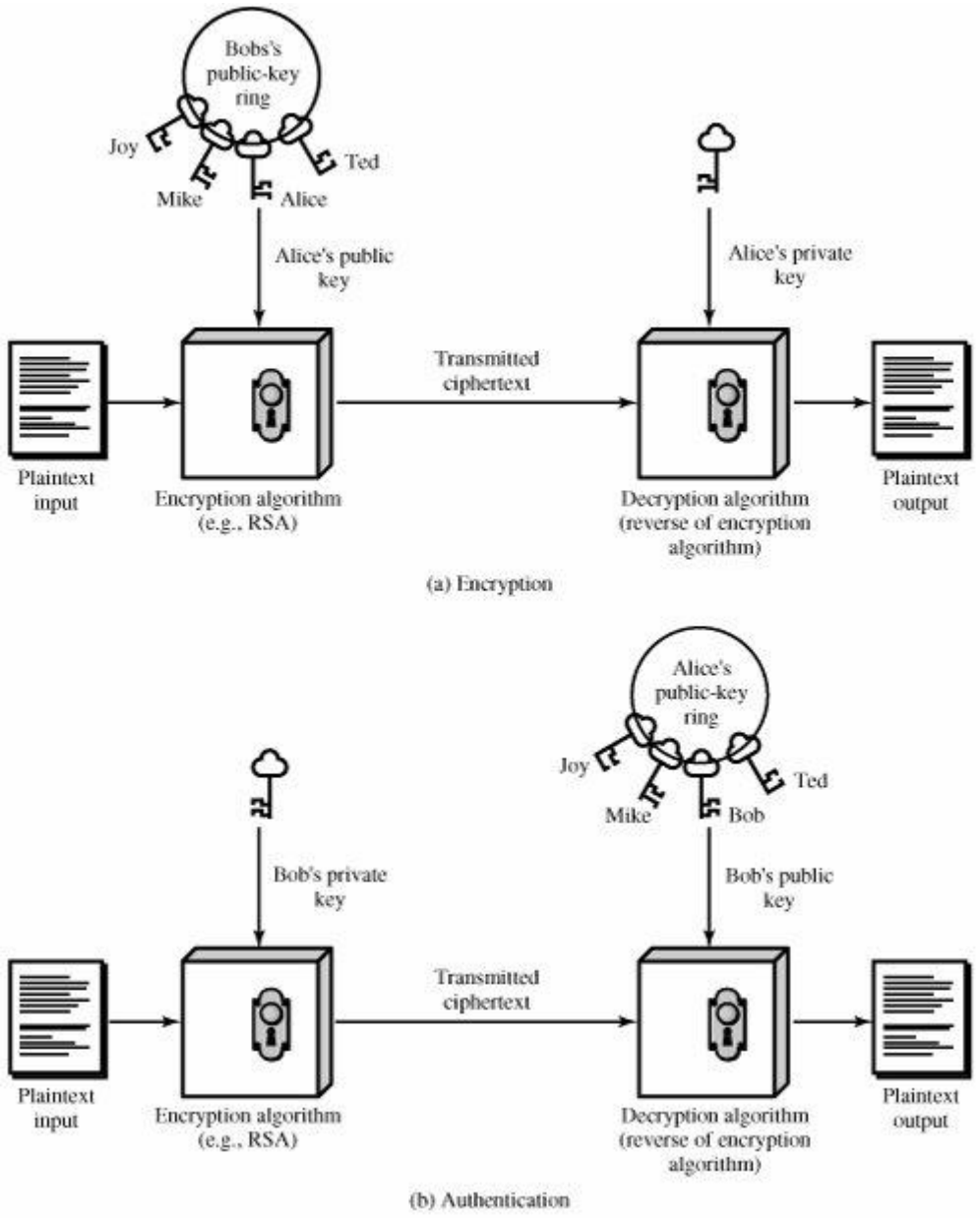

Figure 5. Public-Key Cryptography

Table 3. Conventional and Public-Key Encryption

Conventional Encryption	**Public-Key Encryption**
Needed to Work:	**Needed to Work:**
1.The same algorithm with the same key is used for encryption and decryption.	1.One algorithm is used for encryption and decryption with a pair of keys, one for encryption and one for decryption.

2.The sender and receiver must share the algorithm and the key.	2.The sender and receiver must each have one of the matched pair of keys (not the same one).
Needed for Security:	**Needed for Security:**
1.The key must be kept secret. 2.It must be impossible or at least impractical to decipher a message if no other information is available. 3.Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	1.One of the two keys must be kept secret. 2.It must be impossible or at least impractical to decipher a message if no other information is available. 3.Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.

encrypt items, but is computationally impossible to calculate the inverse function, and instead, each user keeps a separate function to decrypt data private and secure.

References

[1] M. Thiruchelvi , Application of Linear Algebra in Cryptography, International Conference on Information and Image Processing (ICIIP-2014)

[2] Lester S. Hill, Concerning certain linear transformation apparatus of cryptography,Amer. Math. Monthly 38 (1931) 135-154.

[3] David Kahn, The Codebreakers: The Story of Secret Writing,Weidenfeld and Nicolson,London, 1967. See especially pp. 404-410.

[4] Licinius Dimitri Sá de Alcantara1, Towards a simple and secure method for binary cryptography via linear algebra, Revista Brasileira de Computação Aplicada (ISSN 2176-6649), Passo Fundo, v. 9, n. 3, p. 44-55, out. 2017

[5] Jeffrey Hoffstein, Jill Pipher, Joseph H. Silverman
An Introduction to Mathematical Cryptography

[6] NBS–AES. Advanced Encryption Standard (AES). FIPS Publication 197,National Bureau of Standards, 2001. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[7] Nigel Smart, Cryptography: An Introduction (3rd Edition)

[8] Cryptography and Network Security Principles and Practices, Fourth Edition By William Stallings, Publisher: Prentice Hall, Pub Date: November 16, 2005