

IMPLEMENTATION OF CRYPTOGRAPHIC ALGORITHMS AND PROTOCOLS

**A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF**

**Bachelor of Technology
in
Computer Science and Engineering**

By

OMKAR GURU

SANJAY MAJUMDAR

KRITHIKA K



**Department of Electronics & Instrumentation Engineering
National Institute of Technology
Rourkela
2007**



**National Institute of Technology
Rourkela**

CERTIFICATE

This is to certify that the thesis entitled, “Implementation of cryptographic algorithms and protocols” submitted by Sri Omkar Guru , Ms Krithika K and Sri Sanjay Majumdar in partial fulfillments for the requirements for the award of Bachelor of Technology Degree in Computer Science Engineering at National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date:

Prof.B.Majhi
Dept. of Computer Science Engineering
National Institute of Technology
Rourkela - 769008

ACKNOWLEDGEMENT

We would like to articulate our deep gratitude to our project guide Prof. Majhi who has always been our motivation for carrying out the project. It is our pleasure to refer Microsoft Word exclusive of which the compilation of this report would have been impossible. An assemblage of this nature could never have been attempted with out reference to and inspiration from the works of others whose details are mentioned in reference section. We acknowledge our indebtedness to all of them. Last but not the least , our sincere thanks to all of our friends who have patiently extended all sorts of help for accomplishing this undertaking.

OMKAR GURU

K.KRITHIKA

SANJAY MAJUMDAR

CONTENTS

Page No

Abstract

List of Figures

List of Tables

Chapter 1	GENERAL INTRODUCTION	1
Chapter 2	CONVENTIONAL DATA ENCRYPTION	3
2.1	Introduction	4
2.2	Cryptography	4
2.3	Conventional encryption model	6
2.4	Key Distribution-Conventional Cryptography	7
2.5	Cryptanalysis	8
2.6	Classical encryption techniques	9
2.7	Self-repetitive method for hill cipher	12
2.8	Poly-alphabetic cipher	15
2.9	Transposition scheme	16
2.10	Data Encryption Standard	16
2.11	Advanced Encryption standard	20
2.12	Conclusion	23
Chapter 3	PUBLIC KEY CRYPTOGRAPHY	24
3.1	Introduction	25
3.2	Basic Principles	25
3.3	Advantages of Public Key Cryptography	27
3.4	RSA algorithm	29
3.5	Elliptic curve cryptography	30
3.6	Conclusion	33
Chapter 4	DIGITAL SIGNATURE PROTOCOLS	34
4.1	Introduction	35
4.2	Authentication system using RSA signature	35
4.3	Digital signature algorithm	36
4.4	Conclusion	37
Chapter 5	IMPLEMENTATION AND RESULTS	38
5.1	Symmetric key cryptography	39

	5.2	Public key cryptography	41
	5.3	Digital signature protocol	43
	5.4	Hill cipher using self-repetitive matrix	45
Chapter 6		CONCLUSIONS	54
		REFERENCES	56

Chapter 1

INTRODUCTION

INTRODUCTION:

The purpose of the project is to provide a practical survey of both the principles and practice of cryptography. Cryptography has become an essential tool in transmission of information. Cryptography is the central part of several fields: information security and related issues, particularly, authentication, and access control. Cryptography encompasses a large number of algorithms which are used in building secure applications.

AUTHORS CONTRIBUTION:

The authors have implemented cryptographic algorithms of both the conventional symmetric key cryptography as well as public key cryptography. DES, AES and Hill Cipher have been implemented for the Symmetric key. RSA and ECC have been implemented for Public key cryptography. Digital Signature Verification Protocols like DSA and RSA were implemented. In DSA, SHA1 was used for hashing and in RSA, MD5 was used. A complete JAVA package was developed on these algorithms.

The authors have also implemented an alternate method to the conventional cryptographic technique of HILL-CIPHER using the concept of self repetitive matrix. A numerical method has been described and later implemented in generating a random matrix of given periodicity. The method of self repetitive matrix has then been used to simulate a communication channel with proper decompression techniques to facilitate bit saving. This method is easier to implement compared to other techniques like self invertible matrix etc. and if the block size and the modular index is suitably chosen the it becomes extremely tough to crack it by known plaintext attack as it is infeasible to find either the key matrix or the session key used for encryption.

Chapter 2

CONVENTIONAL ENCRYPTION

2.1 Introduction

Conventional Encryption is referred to as symmetric encryption or single key encryption. It was the only type of encryption in use prior to the development of public-key encryption. Conventional encryption can further be divided into the categories of classical and modern techniques. The hallmark of the classical technique is that the cipher or the key to the algorithm is shared i.e. known by the parties involved in the secured communication. So there are two types of cryptography: secret key and public key cryptography. In secret key same key is used for both encryption and decryption. In public key cryptography each user has a public key and a private key. In this chapter a very view of conventional cryptography is presented. In the section 2.8 a new method of cipher developed by author has been presented

2.2 Cryptography:

Cryptography is the study of Secret (crypto-)-Writing (-graphy). It is the science or art of encompassing the principles and methods of transforming an intelligible message into one that is intelligible and then transforming the message back to its original form. As the field of cryptography has advanced; cryptography today is assumed as the study of techniques and applications of securing the integrity and authenticity of transfer of information under difficult circumstances.

Today's cryptography is more than encryption and decryption. Authentication is as fundamentally a part of our lives as privacy. We use authentication throughout our everyday lives when we sign our name to some document and for instance and , as we move to world where our decisions and agreements are communicated electronically, we need to have electronic techniques for providing authentication. Cryptography provides mechanisms for such procedures.

A digital signature binds a document to the possessor of a particular key, while a digital timestamp binds a document to its creation of a particular time. These cryptographic

mechanisms can be used to control access to shared disk drive, a high security installation, or a pay-per-view TV channel. The field of cryptography encompasses other uses as well. With just a few basic cryptographic tools, it is possible to build elaborate schemes and protocols that allow us to pay using electronic money, to prove we know certain information without revealing the information itself, and to share quantity in such a way that a subset of the shares can reconstruct the set. While modern cryptography is growing increasingly diverse, cryptography is fundamentally based on problems that are difficult to solve. A problem may be difficult because its solution requires some secret knowledge such as decrypting an encrypted message or signing some digital document.

Cryptographic systems are generally classified along three independent dimensions:

1. Type of operations used for transforming plaintext to cipher text. All encryption algorithms are based on two general principles. Those are substitution, in which each element in the plain text is mapped into another element and transposition in which elements in the plaintext are rearranged. The fundamental requirement is that no information be lost. Most systems referred to as product systems, involved multiple stages of substitution and transposition.
2. The number of keys used: If sender and receiver use the same key, the system is referred to as symmetric, single key or secret key conventional encryption. If the sender and the receiver each uses a different key the system is referred to as asymmetric, two key, or public-key encryption.
3. The way in which the plaintext is processed: A block cipher processes the input on block of elements at a time, producing an output block for each input block. A stream cipher processes the input elements continuously, producing output one element at a time, as it goes along.

Steganography, hiding one message inside another, is an old technique that is still in use. For example, a message can be hidden inside a graphics image file by using the low order bit of each pixel to encode the message. The visual effect of these tiny changes is probably too small to be noticed by the user. The message can be hidden further by compressing it or by encrypting it with a conventional cryptosystems. Unlike

conventional cryptosystems, where we assume the attacker knows everything about the cryptosystem except for the secret key, Steganography relies on the secrecy of the method of hiding for its security. If eve does not recognize the message as cipher text, then she is not likely to attempt to decrypt it.

2.3 Conventional encryption model:

A conventional encryption model can be illustrated as assigning X_p to represent the plaintext message to be transmitted by the originator. The parties involved select an encryption algorithm represented by E . The parties agree upon the secret key represented by K . The secret is distributed in a secure manner represented by SC . Conventional encryption's effectiveness rests on keeping the key secret. Keeping the key secret rests in a large on key distribution methods. When E processes X_p and K , X_c is derived. X_c represents the cipher text output, which will be decrypted by the recipient. Upon receipt of X_c , the recipient uses a decryption algorithm represented by D to process X_c and K back to X_p . This is represented in the figure. In conventional encryption, secrecy of the encryption and decryption algorithm is not needed. In fact, the use of an established well known and tested algorithm is desirable over an obscure implementation. This brings us to the topic of key distribution.

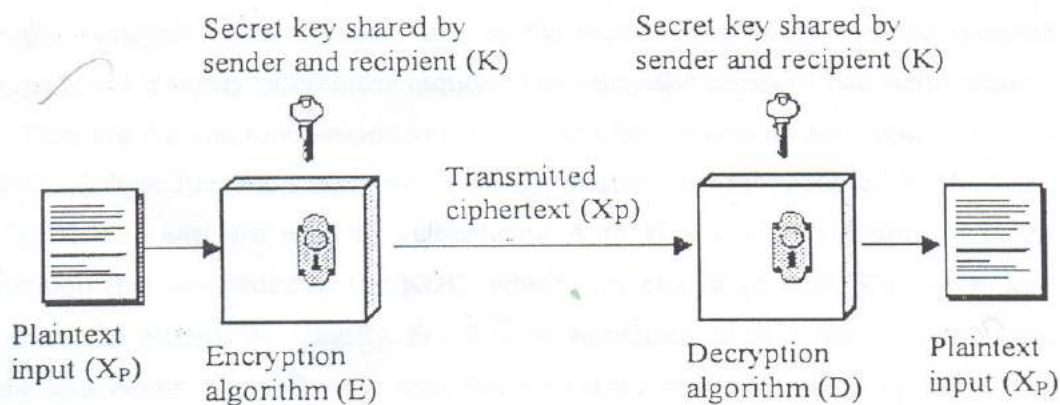


Figure 2.1: Simplified Model of Conventional Encryption

2.4 Key Distribution-Conventional Cryptography:

The strength of any cryptographic system rests on the key distribution technique. Conceivably for two parties: A could select a key and hand deliver it to B, or A and B could rely on a trusted courier. If A and B have an established secure connection, they could exchange a key via encrypted messaging, or if A and B have an established secure connection via a third party C, C could provide this trusted courier service. The first two situations could provide a logistical nightmare if the number of communicating parties increase, the number of discrete keys needed also increases exponentially. In option, weaknesses are revealed, that if an attacker ever is able to compromise any one key it can compromise all the keys i.e. attacker can masquerade as the third party. The fourth option demonstrates a Key Distribution Center (KDC) which is largely adopted. A KDC is responsible for securely delivering unique key pair to its clients. It is also responsible for key management. A key management center uses a hierarchy of keys to provide authentication, integrity, non repudiation, and confidentiality to its users. The hierarchy of keys consists of session keys, which are used for logical connection between end users. The session keys are encrypted by a master key which is shared by the KDC and an end user. Consider this classic key distribution scenario: A uses his secret key to request a session key from KDC to establish a logical connection with B. The request includes both the identities of A and B and a unique identifier for the transaction. A is a contrivance invented or used for this particular, singular occasion. The KDC replies to A with encrypted message which contains the requested one time session key and the original message with the nonce. The original message is used to verify the reply's integrity. The nonce is used by the requestor to verify that the returned message is not a replay of an older request.

The reply also contains two items relating to B. They are the one time session key, to be used for by session and an identifier for A. Both these items are encrypted with the master key shared by KDC and B. The items are used to authenticate A and B. Party A forwards to B the information that originated at KDC, that was encrypted with B's master key. This gives the process its integrity.

For B to be authenticated to A the following steps should occur: Party B now uses the logical connection created by the shared session key to send a half defined nonce to A. Upon receipt of B defined nonce, A performs a mathematical function on the nonce; A performs a mathematical function on the nonce and returns result to B through the logical connection. The keys have to manage across KDC domains. Keys issued to an entity by one KDC have to validate by the issuer before they can be accepted by an entity serviced by a different issuer. The issuers have to collaborate on acceptable method of authenticating inter-domain transaction. Currently, KDCs use a hierarchy for key sharing. Each local KDC negotiates keys for its subscribers through a global KDC. Session generated must have a finite lifetime. Keys are exchanged frequently to prevent an opponent from having a large amount of data encoded with the same key. This brings us to the subject of cryptanalysis.

2.5 Cryptanalysis:

Code making involves the creation of encryption products that provide protection of confidentiality. Code breaking involves defeating this protection by some means other than the standard decryption process used by an intended recipient. Five scenarios for which code breaking is used. They are ensure accessibility, spying on opponents, selling cracking products and services, pursuing the intellectual aspects of code breaking and testing whether one's codes are strong enough. Cryptanalysis is the process of attempting to discover either the plaintext X_p or the key K . Discovery of the encryption is the most desired one as with its discovery all the subsequent messages can be deciphered.

Therefore, the length of encryption key, and the volume of the computational work necessary provides for its strength i.e. resistance to breakage. The longer the key, the stronger the protection, the more brute force is needed. Neither conventional encryption nor public key encryption is more resistant to cryptanalysis than the other. All that the user of an encryption algorithm can strive for is an algorithm that meets one or both of the following criteria: the cost of breaking the cipher exceeds the value of the encrypted information, the time required to break to exceeds the normal lifetime of the information.

2.6 Classical encryption techniques:

The technique enables us to illustrate the basic approaches to conventional encryption today. The two basic components of classical ciphers are substitution and transposition. Then other systems described that combines both substitution and transposition.

2.6.1 Substitution techniques

In this technique letters of plaintext are replaced by or by numbers and symbols. If plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with cipher text bit patterns.

2.6.1.1 Caesar Cipher

Caesar Cipher replaces each letter of the message by a fixed letter a fixed distance away e.g. uses the third letter on and repeatedly used by Julius Caesar.

Can describe the Cipher as:

Encryption: $C = E(P) = (P + 3) \bmod 26$ ----- (2.1)

Decryption: $P = D(C) = (C - 3) \bmod 26$ ----- (2.2)

2.6.1.2 Mono-alphabetic Cipher:

The Caesar cipher uses only 26 rotations out of the 26 permutations on the alphabet. The mono-alphabetic cipher uses them all. A key k is an arbitrary permutation of the alphabet. $E_k(m)$ replaces each letter of a of m by $k(a)$ to yield c . To decrypt $D_k(c)$ replaces each letter b of c by $k^{-1}(b)$. The size of the key space is $26! > 2^{74}$, which is too large for a successful brute force attack. However, mono-alphabetic ciphers can be easily broken down using letter frequency analysis, given a long enough message. Because each occurrence of a letter a in the message is replaced by the same letter $k(a)$, the most frequently occurring letter of m will correspond with the most frequently occurring letter of c . While Jack might know what the most frequently occurring letter of m is, if the message is long enough and she knows that it is English, then it is quite likely that the most frequently occurring letter in m is one of the most frequently occurring letters in

English i.e. 'e' or maybe 't'. She can assume then that the most frequent letter 'b1' in c is 'e', the next most frequent letter b2 in 't', and so forth. Of course, not all these guesses will be correct, but the number of likely candidates for each cipher text letter is greatly reduced. Moreover, many wrong guesses can quickly be discarded even without constructing the entire trial key because they lead to unlikely letter combinations.

2.6.1.3 Playfair Cipher

The Playfair is a substitution cipher bearing the name of the man who popularized but not created it. Sir Charles Wheatstone invented the method, in around 1854; however he named it after his friend Baron Playfair. The Playfair Cipher was developed for telegraph secrecy and it was the first literal digraph substitution cipher. This method is quite easy to understand and learn but not easy to break, because you would need to know the "keyword" to decipher the code. The system functions on how letters are positioned in a 5*5 alphabet matrix. A "KEYWORD" sets the pattern of letters with the other letters the cells of the matrix in alphabetical order (I and j are usually combined in one cell). For instance, suppose we use a keyword Charles then matrix would look like this:

```
c h a r l
e s b d f
g i/j k m n
o p q t u
v w x y z
```

Now supposing the message to be enciphered here is "the scheme really works". First of all the plaintext is divided into two letter groups. If there are double letters occurring, in the message, either an 'x' will be used to separate the double letters or an 'x' will be added to make a two letter group combination. In our example, the phrase becomes:

Enciphered text: th es c hem er ea lx ly wo rk sx

Each of the above two letter combinations will have 3 possible relationships with each other in the matrix: they can be in the same column, same row, or neither. The following rules for replacement should be used:

If two letters are in the same column of the matrix, use letter below it as the cipher text. (columns are cyclical).

If two letters are in the same row of the matrix, use letter to the right as the cipher text.(columns are cyclical).

If neither the same column or row, then each are exchanged with the letter at the intersection of its own row and the other column.

From our example:

Plaintext: th es ch em er lx ly wo rk sx

Cipher text: pr sb ha dg bc az rz vp am bw

For deciphering, the rules are exact opposite.

2.6.1.4 Hill Cipher:

The core of Hill-cipher is matrix manipulations. It is a multi-letter cipher, developed by the mathematician Lester Hill in 1929.

For encryption, algorithm takes m successive plaintext letters and instead of that substitutes m cipher letters. In Hill cipher each character is assigned a numerical value like:

a=0,

b=1,

.....

.....

z=25.

The substitution of cipher text letters in place of plaintext leads to m linear equations. For m=3, the system can be described as follows:

$$C_1 = (K_{11}P_1 + K_{12}P_2 + K_{13}P_3) \text{ MOD } 26 \text{-----} \quad (2.3)$$

$$C_2 = (K_{21}P_1 + K_{22}P_2 + K_{23}P_3) \text{ MOD } 26 \text{-----} \quad (2.4)$$

$$C_3 = (K_{31}P_1 + K_{32}P_2 + K_{33}P_3) \text{ MOD } 26 \text{-----} \quad (2.5)$$

This can be expressed in terms of column vectors and matrices:

$$C=KP$$

Where C and P are column vectors of length 3, representing the plaintext and the cipher text and K is a 3*3 matrix, which is the encryption key. All operations are performed mod 26 here. Decryption requires the inverse of matrix K. The inverse K^{-1} of a matrix K is defined by the equation.

$KK^{-1}=I$ where I is the Identity matrix.

NOTE: The inverse of a matrix doesn't always exist, but when it does it satisfies the preceding equation.

K^{-1} is applied to the cipher text, and then the plain text is recovered. In general terms we can write as follows:

For encryption: $C=E(K, P)=KP$

For decryption: $P=D(K, C)=K^{-1}C=K^{-1}KP=P$

2.7 Self-repetitive method for hill cipher:

As we have seen in Hill cipher decryption, it requires the inverse of a matrix. So while one problem arises that is:

Inverse of the matrix doesn't always exist. Then if the matrix is not invertible then encrypted text cannot be decrypted.

In order to overcome this problem author suggests the use of self-repetitive matrix.

This matrix if multiplied with itself for a given mod value (i.e. mod value of the matrix is taken after every multiplication) will eventually result in an identity matrix after N multiplications. So, after N+ 1 multiplication the matrix will repeat itself.

Hence, it derives its name i.e. self-repetitive matrix. It should be non-singular square matrix.

Note: Historically, there are already many available methods, like self-invertible matrix etc. But this method as suggested by the author is both easy to compute and simpler to implement.

2.7.1 Modular Arithmetic: A brief Analysis

The analysis presented here for generation of self-repetitive matrix is valid for matrix of positive integers that are the residues of modulo arithmetic on a prime number. So in analysis the arithmetic operations presented here are addition, subtraction, Unary operation, Multiplication and division.

The Modulo operator have the following properties:

1. $a \equiv b \pmod{p}$ if $n \mid (a-b)$
2. $(a \pmod{p}) \equiv (b \pmod{p}) \Rightarrow a \equiv b \pmod{p}$
3. $a \equiv b \pmod{p} \Rightarrow b \equiv a \pmod{p}$
4. $a \equiv b \pmod{p}$ and $b \equiv a \pmod{p} \Rightarrow a \equiv c \pmod{p}$

The modulo arithmetic have the following properties:

Let $Z = [0, 1, \dots, p-1]$, the set residues modulo p . If modular arithmetic is performed within the set Z_n , the following equations present the arithmetic identities:

1. Addition: $(a+b) \pmod{p} = [(a \pmod{p}) + (b \pmod{p})] \pmod{p}$
2. Subtraction: $(a-b) \pmod{p} = [(a \pmod{p}) - (b \pmod{p})] \pmod{p}$
3. Multiplication: $(a*b) \pmod{p} = [(a \pmod{p}) * (b \pmod{p})] \pmod{p}$
4. Negation: $-a \pmod{p} = p - (a \pmod{p})$
5. Division: $(a/b) \pmod{p} = c$ when $a = (c*b) \pmod{p}$
6. Multiplicative inverse: $(a^{-1}) = c$ if there exists $(c*z) \pmod{p} = 1$

Table exhibits the properties of modulo arithmetic:

SL. No.	Property	Expression
1.	Commutative Law	$(w + x) \pmod{p} = (x + w) \pmod{p}$ $(w*x) \pmod{p} = (x*w) \pmod{p}$
2.	Associative Law	$[(w + x) + y] \pmod{p} = [w + (x + y)] \pmod{p}$
3.	Distributive Law	$[w*(x + y)] \pmod{p} = [w*x + w*y] \pmod{p}$ $[w*(x * y)] \pmod{p} = [(w*x \pmod{p}) * (w*y \pmod{p})] \pmod{p}$
4.	Identities	$(0+a) \pmod{p} = a \pmod{p}$ and $(1*a) \pmod{p} = a \pmod{p}$

5.	Inverse	<p>For each X belongs to z_p, there exists y such that $(x + y) \bmod p = 0$ then $y = -x$</p> <p>For each X belongs to z_p, there exists y such that $(x * y) \bmod p = 1$</p>
----	---------	--

2.7.2 Complex modular numbers:

Used while calculating negative or non-existent square roots.

Say, for example, for mod 7

Sqrt (1)=1;

Sqrt (2)=4;

Sqrt (4)=2;

But square root of 3, 5, 6 doesn't exist

So sqrt (6)=sqrt (-1) sqrt (-6)=j where j stand for square of -1 in modular arithmetic

Similarly, sqrt (5)=4j; sqrt (3)=2j

Extensive no of simulation exercises were carried out to understand the properties of modular arithmetic.

2.7.3 Generation of a self repetitive Matrix A for a Given N:

The initial conditions for the existence of a self-repetitive matrix are:

1. The matrix should be square.
2. It should be non-singular.

But trying to find out the value of N (the value where the matrix becomes a identity matrix) through the method of brute force may not be the best idea always; because the matrix is of dimension greater than 5*5 and with mod index (i.e.) greater than 91 then the brute force technique might take very long time and N value may be in the range of millions. A normal Pentium 4 machine might hang if asked to do the computations for 15*15 matrixes or more.

Hence, it would be comfortable to know the value of N and then generate a random matrix accordingly.

This can be done as follows:

1. First a diagonal matrix A is chosen, and then the values powers of each individual element when they reach unity is calculated and denoted as n1, n2, n3.... Now LCM of these values is taken to given the value of N.
2. Now the next step is generate a random square matrix whose N value is same as the N calculated in the previous step.
3. Pick up any random invertible square matrix B
4. Generate $C=B^{-1}AB$
5. The N value of C is also N

Mathematical proof:

$$(B^{-1}AB)^N = (B^{-1})^N * (A)^N * (B)^N$$

$A^N=I$ as calculated before as it is a diagonal matrix and N is the LCM of all elements
 $(B^{-1}B)*(B^{-1}*B)...n \text{ times}=I$

NOTE: The algorithm for hill cipher matrix using self-repetitive matrix is given in the implementation chapter. For further illustration, a simple example has been explained with each step in the implementation chapter.

2.8 Poly-alphabetic Cipher:

A poly-alphabetic cipher is any cipher based on substitution, using multiple substitution alphabets. Mono-alphabetic Cipher can be broken. The reason: Same plain letters are encoded to same cipher letters; the underlying letter frequencies remain unchanged.

Cryptographers have tried to overcome this dilemma simply by assigning various cipher letters or symbols to same plain letters. Such ciphers are called Poly-alphabetic Ciphers.

The most popular of such ciphers is the “Vigenere Cipher”. The Vigenere Cipher is an improvement of the Caesar Cipher key is multiple letters long $K=k_1, K_2, k_3 \dots k_d$. To encrypt a message, a key is needed that is long as message. Usually, the key is a repeating keyword.

Decryption is equally simple. The key letter again identifies the row. The position of the cipher text letter in that row determines the column, and the plaintext letter is at the top of the column.

2.9 Transposition Scheme:

Transposition (or permutation) does not alter any of the bits in the plaintext, but instant moves the position around within it. If thr resultant Cipher text is then put through more transpositions, the end result has increasing security.

2.9.1 Rail Fence Technique:

The simplest transposition ciphering technique is the rail fence cipher, in which we write message with letters, alternate rows and read off Cipher row by row. For example, to encipher the message “I CAME I SAW I CONQUERED” with a rail fence of depth 2, we write the following:

Plain: I A E S W C N U R D

Cipher: C M I A I O Q E E

Cipher-Text: IAESWCNURDCMIAIOQEE

Amore complex scheme is to write the message in a rectangle, row by row, and read the message off, column by column, but permute the order of columns. The order of the columns then becomes the key to the algorithm.

2.10 Data Encryption Standard:

Data encryption standard is the most widely used method of data encryption using a secret key. There are 72,000,000,000,000,000 (72 quadrillion) or more possible encryption keys that can be used. For each given message, the key is chosen at random from among this enormous number of keys. Like other private key cryptographic methods, both the sender and the receiver must know and use the same private key.

The National Bureau of Standards with the help of the National Security Agency developed it in the 1970s. Its purpose is to provide a standard method for sensitive commercial and unclassified data. IBM created the first draft of the algorithm, calling it LUCIFER. DES officially became a standard in November of 1976.

Data encryption algorithm has a 64-bit block size and uses a 56bit key during execution (8 parity bits are stripped of from the full 64-bit key). The DEA can also be used for single user encryption, such as to store files in a hard in encrypted form. NIST re-certifies DES every 5 years. DES has been in world wide use for over 20 years, and due to the fact

that it is a defined standard that any system implementing DES can communicate with any other system using it.

2.10.1 DES Encryption:

DES is a symmetric, block-cipher algorithm with a key length of 64 bits, and the algorithm operates on successive 64 bit blocks of plain text. Due to symmetric, the same key is used for encryption and decryption, and also uses the same algorithm for encryption and decryption.

Initially a transposition is carried out according to a set table (the initial permutation), the 64-bit plaintext block is then split into two 32-bit block, and 16 identical operations called rounds are carried out on each half. The two halves are joined back together, and the reverse of the initial permutation is carried out. The purpose of the first transposition is clear; it does not affect the security of the algorithm, but is through for the purpose of allowing plaintext and cipher text to be loaded into 8-bit chip in byte-sized pieces. In any round, only one half of the original 64-bit block is operated on. The rounds alternate between the two halves. One round in DES consists of the following:

2.10.2 Key Transformation:

The keys reduced from 64-bit to 56-bit by removing every eighth bit, which are sometimes used for error checking. 16 different 48 bit sub-keys are then generated i.e. one for each round. This is achieved by splitting the 56-bit key into the two halves, and then circularly shifting them left by one or two bits, depending on the round. After this, 48 of the bits are selected. Because they are shifted, different groups of key bits are used in each sub key. This process is called compression permutation due to transposition of bits & reduction of the overall size.

2.10.3 Expansion Permutation:

Whichever half of the block is being operated on undergoes a permutation after key transformation. In this operation, the expansion & the transposition are achieved

simultaneously by allowing the first and fourth bits in each block for bit block to appear twice in the output that is the fourth input bit becomes the fifth and the seventh output bit.

The expansion permutation achieves 3 things. Those are described below:

- 1) It increases the size of the half block from 32 to 48 bit, the same number of bit as in compressed key subset, which is important as the next operation is to XOR the two together.
- 2) It produces a long string of data for the substitution operation that subsequently comprises it.
- 3) Because in the subsequent substitutions on the first & 4th bits appearing in 2 S-boxes, they affect two substitutions. The effect of this is the dependency of the output on the input bits is that the dependency of the output on the input bits increases rapidly.

2.10.4 XOR:

XOR operation performed with the appropriate subset key for that round & the resulting 48 block.

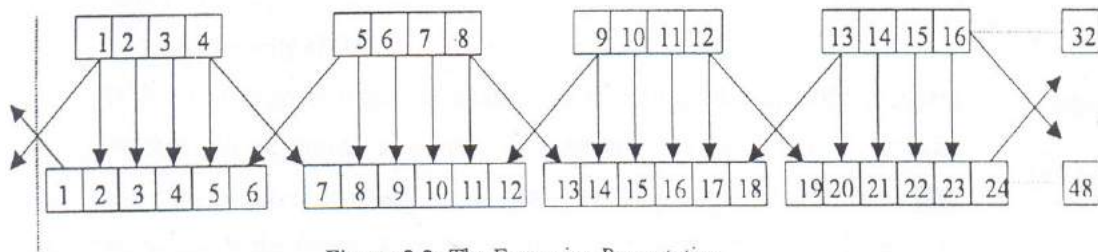


Figure 2.2: The Expansion Permutation

2.10.5 Substitution:

After XOR operation the next operation is to perform substitution on the expanded block. There are 8 substitution boxes called S-boxes. The first S-Box operates on the first 6 bits of the 48 bit expanded block, the second S-box on the next 6 & so on. Each S-box operates from a table of 4 rows & 16 columns; each entry on a table is a 4 bit number. The 6 bit number the s-box takes as input is used to look up the appropriate entry in the table in the following way. The first and the 6 bits combine to form a two bit number

corresponding to a row number, the second and fifth bit combine to form a 4 bit number corresponding to a particular column. The net result of the substitution phase is 8 4bit blocks that are then combined to form a 32 bit block. It is the non-linear relationship of the S-boxes that really provides DES with its security.

2.10.6 Permutation:

The 32 bit output of a substitution phase then undergoes a straight forward transposition using a table called P-Box. After all the round has been completed, the two half blocks of 32 bits are recombined to form a 64 bit output. The final permutation is performed on it, and the resulting 64 bit block is the desired DES encrypted cipher text of the input plain text block.

2.10.7 DES Decryption:

If one has the correct key decrypting DES is very easy. The decryption algo is identical to the encryption algo. The only change is to decrypt DES cipher text; the subsets of the keys use in each round are used in reverse, which is the 16th subset is used first.

2.10.8 Security of DES:

DES can no longer be considered a sufficiently secured algorithm if the DES secured message can be broken in minutes by a super computer. Then the rapidly increasing power of computer means, it'll be trivial to break DES in future. An extension of DES called DESX is considered virtually immune to key search.

2.11 Advanced Encryption Standard:

The Rijndael proposal for AES defined a cipher in which the block length and the key length can be independently specified to be 128, 192, 256 bits. A number of AES parameters depend on the key length.

Rijndael was designed to have the following characteristics:

- Resistance against all known attacks.
- Speed and code compactness on a wide range of platforms.

- Design simplicity.

2.11.1 AES Structure:

We can make several comments about the overall AES structure.

1. The AES structure is not a feistel structure. Rijndael do not use a Feistel structure but process the entire data block in parallel during each round using substitution and permutation.

2. The key that is provided as input is expanded into an array of forty-four 32 bit words. Four distinct words serve as a round key for each round.

Four different stages are used, one of permutation and three of substitution:

1. Substitute bytes: Uses an S-box to perform a byte –by-byte substitution of the block.
2. ShiftRows: A simple permutation.
3. MixColumns: A substitution that makes use of arithmetic over $GF(2^8)$.
4. AddRoundKey: A simple bit wise XOR of the current block with a portion of the expanded key.

3. The structure is quite simple. For both encryption and decryption, the cipher begins with an AddRoundKey stage, followed by nine rounds that each includes all four stages, followed by a tenth round of three stages.

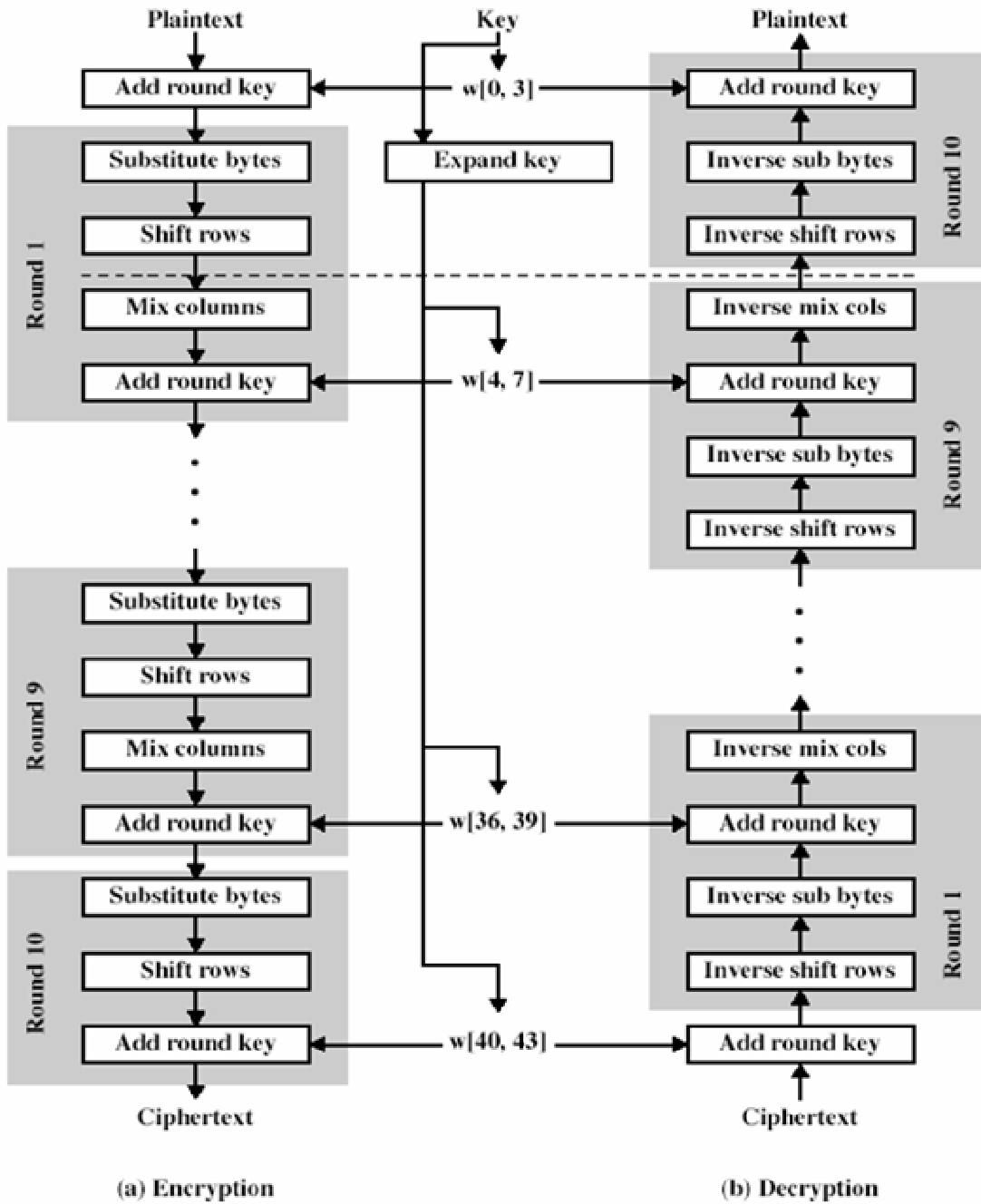
4. Only the AddRoundKey stage makes use of the key. For this reason, the cipher begins and ends with an AddRoundKey.

5. The AddRoundKey is a form of vernam cipher and by itself is not formidable. The other three stages together provide confusion, diffusion and nonlinearity, but by themselves would provide no security because they do not use the key.

6. Each stage is easily reversible. For the Substitute Byte, ShiftRows and MixColumns stages, an inverse function is used in decryption algorithm. For the AddRoundKey stage, the inverse is achieved by XORing the same round key to the block, using the result that $A \text{ XOR } A \text{ XOR } B=B$.

7. The decryption algorithm makes use of the expanded key in reverse order. The decryption algorithm is not identical to the encryption algorithm.

8. The final round of both encryption and decryption consists of only three stages. This is a consequence of the particular structure of AES and is required to make the cipher reversible.



2.12 Conclusion:

There are several methods of conventional cryptography, and since it is not possible to present all the methods, very important and popular methods were presented and implemented.

It is seen that the modified Hill cipher Encryption and Decryption requires generating random Matrix, which is essentially the power of security. As we know in Hill cipher Decryption requires inverse of the matrix. Hence while decryption one problem arises that is. Inverse of the matrix does not always exist. Then if the matrix is not invertible then encrypted text cannot be decrypted. But this drawback is completely eliminated in modified Hill cipher algorithm.

At the same time, this method requires the cracker to find the inverse of many square matrices, which is not computationally easy. So this modified Hill-Cipher method is both easy to implement and difficult to crack.

Chapter 3

PUBLIC-KEY CRYPTOGRAPHY

3.1 Introduction:

Public-Key Algorithms are asymmetric, that is to say the key that is used to encrypt the message is different from the key used to decrypt the message. The encryption key, known as the Public key is used to encrypt a message, but the message can only be decoded by the person that has the decryption key, known as the private key.

This type of encryption has a number of advantages over traditional symmetric Ciphers. It means that the recipient can make their public key widely available- anyone wanting to send them a message uses the algorithm and the recipient's public key to do so. An eavesdropper may have both the algorithm and the public key, but will still not be able to decrypt the message. Only the recipient, with the private key can decrypt the message.

A disadvantage of public-key algorithm is that they are more computationally intensive than symmetric algorithms, and therefore encryption and decryption take longer. This may not be significant for a short text message, but certainly is for bulk data encryption.

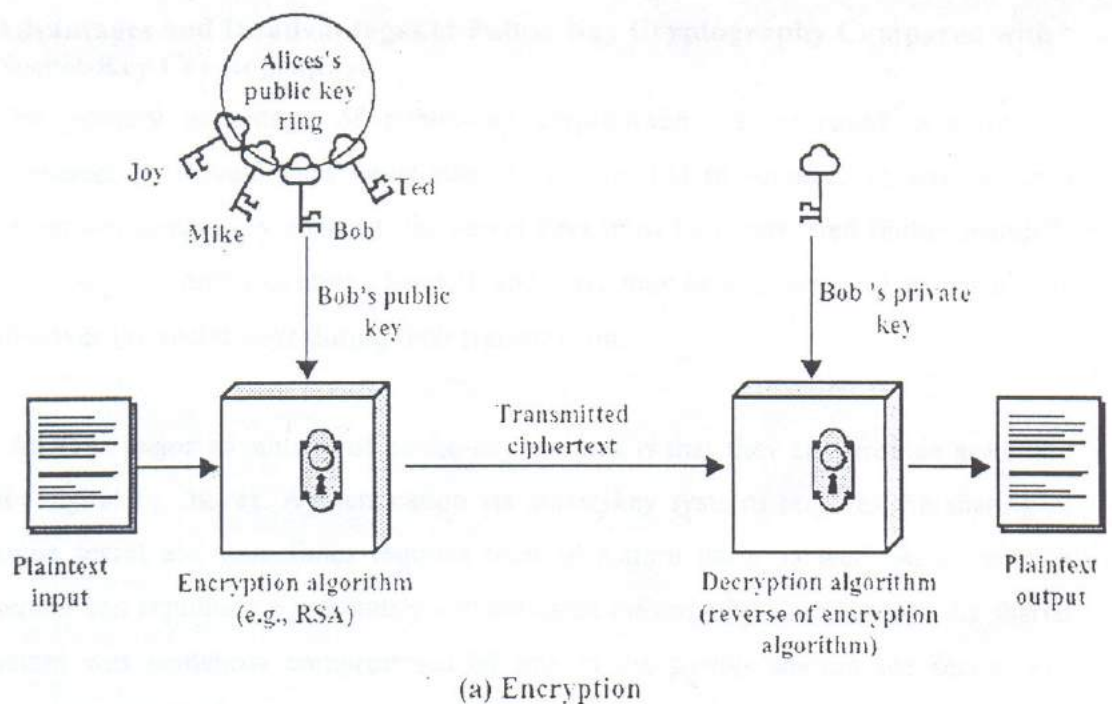
3.2 The Basic Principle:

In order to decrypt a message, Bob (the recipient) has to know the key. However, it may be difficult for Alice (the sender) to tell Bob what the key is. If they simply agree on a key by e-mail for example, Eve could be listening in on their e-mail conversation and thus also learn what the key is. Public key cryptography was invented to solve this problem.

When using public-key cryptography, Alice and Bob both have their own key pairs. A key pair consists of a public key and a private-key. If the public-key is used to encrypt something, then it can be decrypted only using the private-key. And similarly, if the private-key is used to encrypt something, then it can be decrypted only using the public-

key. It is not possible to figure out what the private-key is given only the public-key, or vice versa.

This makes it possible for Alice and Bob to simply send their public keys to one another, even if the channel they are using to do so is insecure. It is no problem that Eve now gets a copy of the public keys. If Alice wants to send a secret message to Bob, she encrypts the message using Bob's public key. Bob then takes his private key to decrypt the message. Since Eve does not have a copy of Bob's private key, she cannot decrypt the message. Of course this means that Bob has to carefully guard his private key. With public key cryptography it is thus possible for two people who have never met to securely exchange messages. Figure 3.1 illustrates the public-key encryption process.



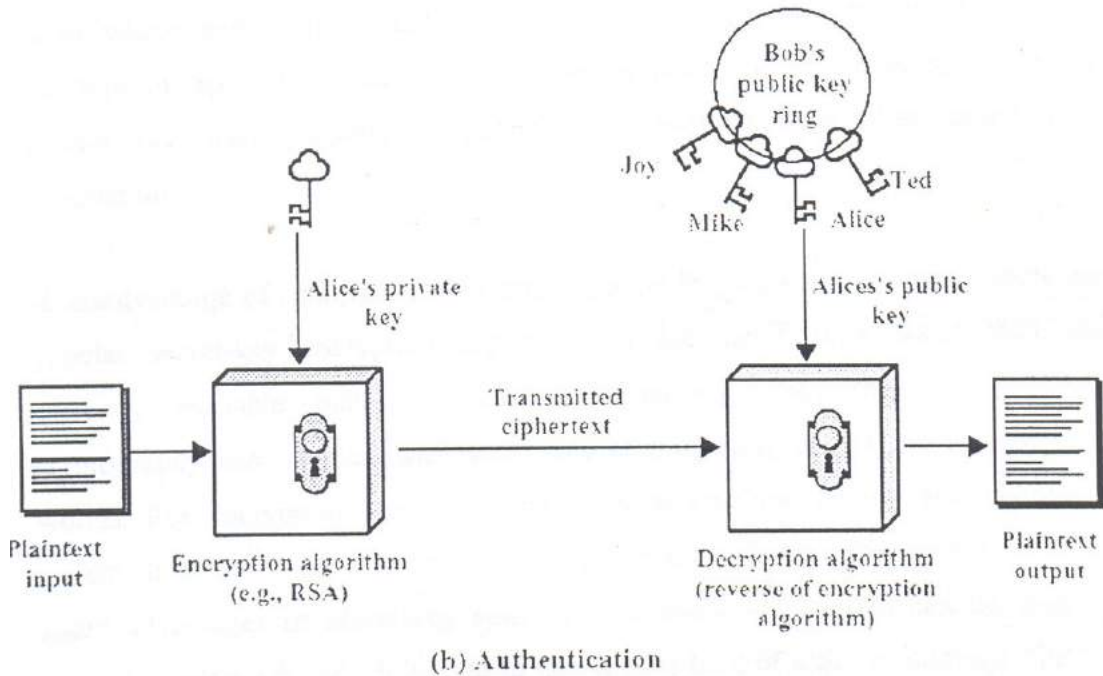


Figure 3.1: Public-Key Encryption

3.3 Advantage and Disadvantage of Public-Key Cryptography:

1. The primary advantage of public-key cryptography is increased security and convenience. Private keys never need to be transmitted or revealed to anyone. In a secret-key system, by contrast, the secret keys must be transmitted (either manually or through a communication channel), and there may be a chance that an enemy can discover the secret keys during their transmission.
2. Another major advantage of public-key systems is that they can provide a method for digital signatures. Authentication via secret-key systems requires the sharing of some secret and sometimes requires trust of a third party as well. As a result, a sender can repudiate a previously authenticated message by claiming that the shared secret was somehow compromised by one of the parties sharing the secret. For example, the Kerberos secret-key authentication system involves a central database that keeps copies of the secret keys of all users; an attack on the database would allow widespread forgery. Public-key authentication, on the other hand, prevents this type of repudiation; each user has sole responsibility for protecting

his or her private key. This property of public-key authentication is often called non-repudiation.

3. A disadvantage of using public-key cryptography for encryption is speed; there are popular secret-key encryption methods that are significantly faster than any currently available public-key encryption method. Nevertheless, public-key cryptography can be used with secret-key cryptography to get the best of both worlds. For encryption, the best solution is to combine public- and secret-key systems in order to get both the security advantages of public-key systems and the speed advantages of secret-key systems. The public-key system can be used to encrypt a secret key, which is used to encrypt the bulk of a file or message. Such a protocol is called a digital envelope.
4. Public-key cryptography may be vulnerable to impersonation, however, even if users' private keys are not available. A successful attack on a certification authority will allow an adversary to impersonate whomever the adversary chooses to by using a public-key certificate from the compromised authority to bind a key of the adversary's choice to the name of another user.

In some situations, public-key cryptography is not necessary and secret-key cryptography alone is sufficient. This includes environments where secure secret key agreement can take place, for example by users meeting in private. It also includes environments where a single authority knows and manages all the keys (e.g., a closed banking system) Since the authority knows everyone's keys already, there is not much advantage for some to be "public" and others "private" Also, public-key cryptography is usually not necessary in a single-user environment. For example, if you want to keep your personal files encrypted, you can do so with any secret-key encryption algorithm using, say, your personal password as the secret key. In general, public-key cryptography is best suited for an open multi-user environment.

Public-key cryptography is not meant to replace secret-key cryptography, but rather to supplement it, to make it more secure. The first use of public-key techniques was for secure key exchange in an otherwise secret-key system; this is still one of its primary

functions. Secret-key cryptography remains extremely important and is the subject of ongoing study and research. Some secret-key cryptosystems are discussed in the sections on Block Cipher and Stream Cipher.

3.4 The RSA Algorithm:

The RSA cryptosystem, named after its inventors R. Rivest, A. Shamir, and L. Adleman, is the most widely used public key Cryptosystem. It may be used to provide both secrecy and digital signatures and its security is based on the intractability of the integer factorization.

3.4.1 Description of the Algorithm:

Two very large prime numbers, normally of equal length, are randomly chosen then multiplied together.

$$N = A \times B \quad (3.1)$$

$$T = (A-1) \times (B-1) \quad (3.2)$$

A third number is then also chosen randomly as the public key (E) such that it has no common factors (i.e. is relatively prime) with T. The private key (D) is then:

$$D = E^{-1} \text{ mod } T \quad (3.3)$$

To encrypt a block of plaintext (M) into cipher text (C):

$$C = M^E \text{ mod } N \quad (3.4)$$

To Decrypt:

$$M = C^D \text{ mod } N \quad (3.5)$$

3.4.2 Security of RSA:

The security of RSA algorithm depends on the ability of the hacker to factorize numbers. New, faster and better methods for factoring numbers are constantly being devised. The best for long numbers is the Number Field Sieve. Prime Numbers of a length that was unimaginable a mere decade ago are now factored easily. Obviously the longer a number is, the harder is to factor, and so the better the security of RSA. As theory and computers improve, large and large keys will have to be used. The advantage in using extremely long keys is the computational overhead involved in encryption / decryption. This will only become a problem if a new factoring technique emerges that requires keys of such lengths to be used that necessary key length increases much faster than the increasing average speed of computers utilizing the RSA algorithm. RSA's future security relies solely on advances in factoring techniques.

3.5 Elliptic Curve Cryptography:

Public key cryptography systems are usually based on the assumption that a particular mathematical operation is easy to do, but difficult to undo unless you know some particular secret. This particular secret serves as the secret key. A recent development in this field is the so-called Elliptic Curve Cryptography.

Elliptic Curve Cryptography works with point on a curve. The security of this type of public key cryptography depends on the elliptic curve discrete logarithm problem. Elliptic curve cryptography was invented by Neil Koblitz in 1987 and by Victor Miller in 1986. The principles of elliptic curve cryptography can be used to adapt many cryptographic algorithms, such as Diffie-Hellman or ElGamal. Although no general patent on elliptic curve cryptography appears to exist, there are several patents that may be relevant depending on the implementation. The main advantage of elliptic curve cryptography is that the keys can be much smaller. Recommended key sizes are in the order of 160 bits rather than 1024 bits for RSA.

3.5.1 Elliptic Curves:

An elliptic curve is a set of points (x, y) , for which it is true that

$$y^2 = x^3 + ax + b \quad (3.21)$$

Given certain chosen numbers a and b . Typically the numbers are integers (whole numbers), although in principle the system also works with real (fractional) numbers. Despite what the name suggests, the curves do not have an elliptic shape. For example, -4 and $b = 0.67$ gives the elliptic curve with equation $y^2 = x^3 - 4x + 0.67$. This curve is illustrated in Figure 3.2.

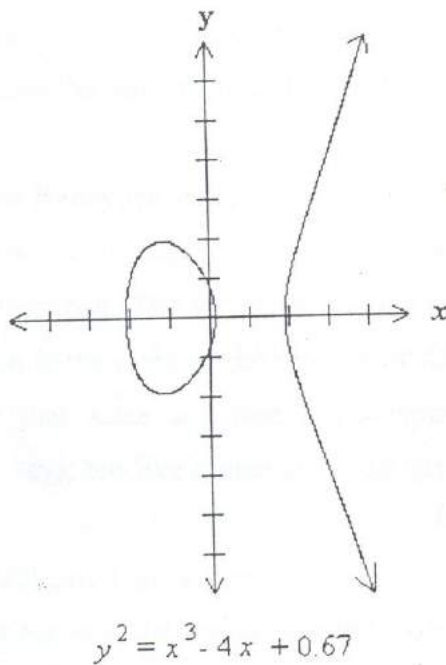


Figure 3.2: Example of Elliptic Curves

If $x^3 + ax + b$ contains no repeated factors, or equivalently if $4a^3 + 27b^2$ is not 0, then the elliptic curve can be used to form a group. A group is simply a set of points on the curve. Because it is a group, it is possible to "add up" points, which give another point on the curve. On the graph, drawing a line through both and taking as the outcome add up two points where the line intersects the curve.

For cryptographic purposes, an elliptic curve must have only points with all coordinates whole numbers (integers) in the group.

The trick with elliptic curve cryptography is that if you have a point F on the curve, all multiples of these points are also on the curve [22, 23].

3.5.2 Generating an Elliptic Curve Public key:

Alice and Bob agree on an elliptic curve and pick a certain point F on this curve. They can do this with Eve listening in. Alice next picks a random number A_s (which does not have to be a point on the curve) and computes $AI' = A_s * F$. The point AI' is on the curve, [because it is a multiple of F . This is easy to compute Alice's public key is AI' and her secret key is A_s . Bob does the same thing and ends up with B_p and B_s .

3.5.3 Encrypting and Decrypting Messages:

Alice and Bob can now secretly agree on a key with which they can encrypt messages using secret key cryptography. The key simply is the product of Alice's public key and Bob's secret key (which is the same as the product of Alice's secret key and Bob's public key). It will be clear that Alice and Bob can compute this product after they have exchanged their public keys, but Eve cannot since she has none of the secret keys.

3.5.4 Cracking the Elliptic Curve Key:

If Eve wanted to crack the key, she would have to reconstruct one of the secret keys. This means having to compute A_s given AI' and F (because $AI' = A_s * F$). And that is very difficult.

The number of discrete points on the curve (points with both X and Y coordinates being integers) is called the order of the curve. If the order of the point F is a prime number of n bits, then computing A_s from $A_s * F$ and F takes roughly 2^{n^2} operations. If F is, say, 160 bits long, then Eve needs about 2^{80} operations. If she can do a billion operations per

second, this takes about 38 million years. This problem is commonly referred to as the elliptic curve discrete logarithm problem.

3.6 Conclusion:

Brief review of public-key cryptography is presented in this chapter. Also some popular public-key cryptography methods presented briefly. Public-key algorithms are based on mathematical functions rather than on substitution and permutation. Hence the advantage of public key algorithm is that they are more computationally intensive than symmetric algorithms. More important, public-key cryptography is asymmetric involving the use of two separate keys, in contrast to symmetric conventional encryption, which uses only one key. The use of two keys has profound consequences in the areas of confidentiality, key distribution and authentication. The implementation of RSA algorithm and ECC are shown in the implementation chapter.

Chapter 4

DIGITAL SIGNATURE PROTOCOLS

4.1 Introduction

A signature scheme is a method of signing a message stored in electronic form. As such, a signed message can be transmitted over a computer network.

A signature scheme consists of two components: a signing algorithm and a verification algorithm. Bob can sign a message x using a (secret) signing algorithm sig . The resulting signature $sig(x)$ can subsequently be verified using a public verification algorithm ver . Given a pair (x, y) , the verification algorithm returns an answer “true” or “false” depending on whether the signature is authentic.

4.2 RSA Authentication system using MD5 for Hashing

RSA can be used for Digital Signature. For Digital signature the message is first encrypted by private key of sender and encryption is done by public key of receiver. This gives confidentiality and authentication. MD5 can be used for preparing a message digest which is encrypted along with the message and sent to the receiver for verification.

The general algorithm used can be summed up as:

4.2.1 Sender:

1. Apply hash function to the message $h(m)$ -MD5.
2. Choose two primes p and q , and computes $n = pq$.
3. Choose e_A such that $1 < e_A < \phi(n)$ with $\gcd(e_A, \phi(n)) = 1$.
4. Calculate d_A such that $e_A d_A \equiv 1 \pmod{\phi(n)}$. Keep d_A, p, q secret and publish (E, n)
5. Sign the message $S \equiv h(m)d_A \pmod{n}$. The pair (m, S) is made public.

4.2.2 Receiver:

1. Apply hash function to the message $h(m)$.
2. Decrypt the signature $z \equiv S e_A \pmod{n}$.

3. If $z = h(m)$, the signature is valid

4.3 Digital Signature Algorithm

DSA is based on Digital Signature Standard and essentially a modification in ElGamal Signature Scheme. The security of the algorithm is based on problem of finding discrete logarithm of large values of prime.]

The Digital Signature Algorithm can be summed up in three steps :

4.3.1 Key generation:

1. Choose a 160-bit prime q .
2. Choose an L -bit prime p , such that $p=qz+1$ for some integer z , $512 = L = 1024$, and L is divisible by 64.
3. Choose h , where $1 < h < p - 1$ such that $g = hz \bmod p > 1$. (Recall that $z = (p-1) / q$.)
4. Choose x by some random method, where $0 < x < q$.
5. Calculate $y = gx \bmod p$.
6. Public key is (p, q, g, y) . Private key is x .

4.3.2 Signing:

1. Generate a random per-message value k where $0 < k < q$
2. Calculate $r = (gk \bmod p) \bmod q$
3. Calculate $s = (k^{-1}(\text{SHA}(m) + x*r)) \bmod q$, where $\text{SHA}(m)$ is the SHA-1 hash function applied to the message m
4. Recalculate the signature in the unlikely case that $r=0$ or $s=0$
5. The signature is (r, s)

The extended Euclidean algorithm can be used to compute the modular inverse $k^{-1} \pmod q$.

4.3.3 Verifying:

1. Reject the signature if either $0 < r < q$ or $0 < s < q$ is not satisfied.
2. Calculate $w = (s)^{-1} \pmod q$
3. Calculate $u_1 = (\text{SHA}(m) * w) \pmod q$
4. Calculate $u_2 = (r * w) \pmod q$
5. Calculate $v = ((g^{u_1} * y^{u_2}) \pmod p) \pmod q$

The signature is valid if $v = r$

4.4 Conclusion:

Digital signature is present advancement in the field of cryptography which has immense practical usage in the field of military, banking creation of authentication system applications. The above described algorithms were implemented in JAVA platform.

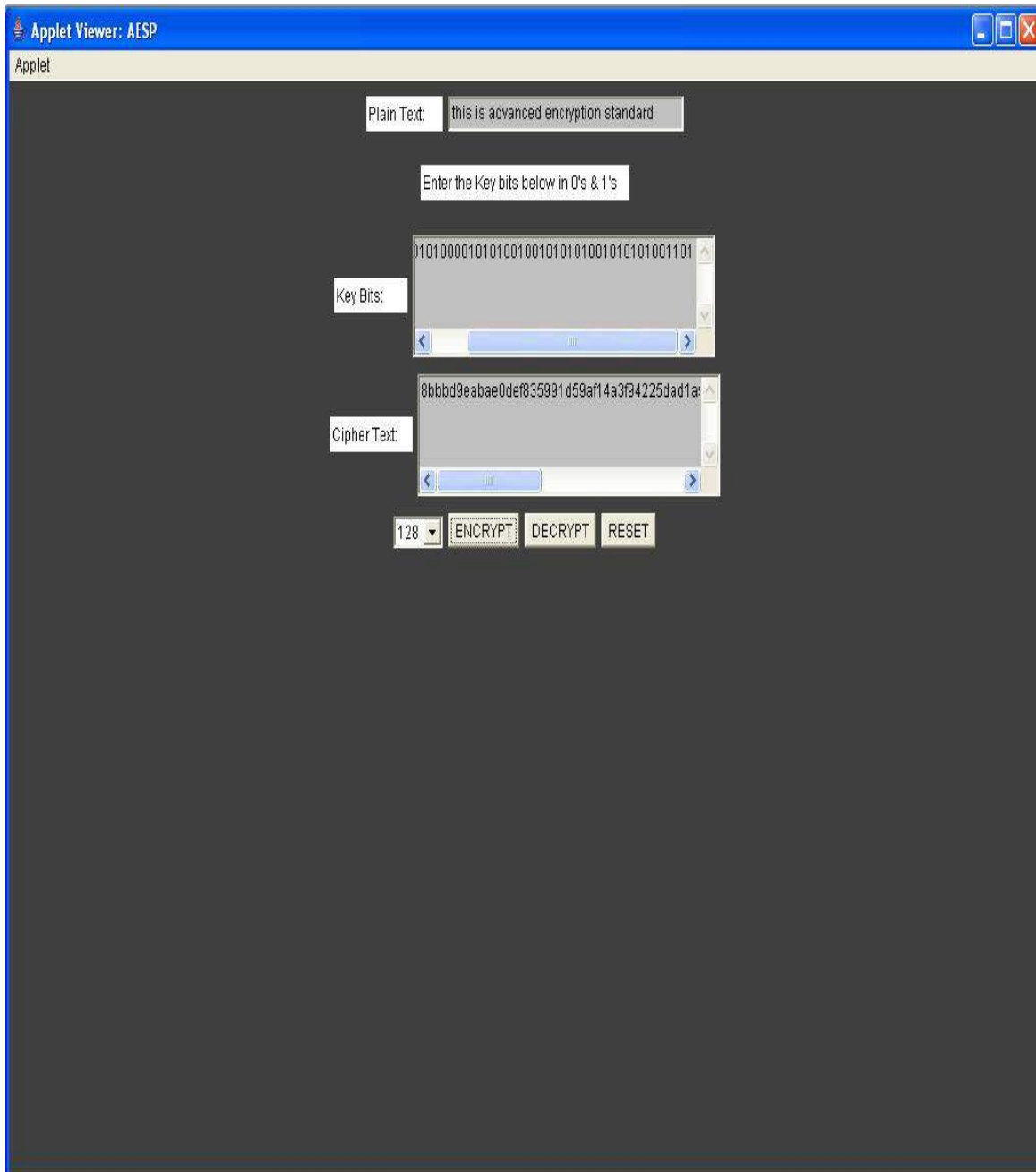
Chapter 5

IMPLEMENTATION AND RESULTS

5.1 SYMMETRIC KEY CRYPTOGRAPHY:

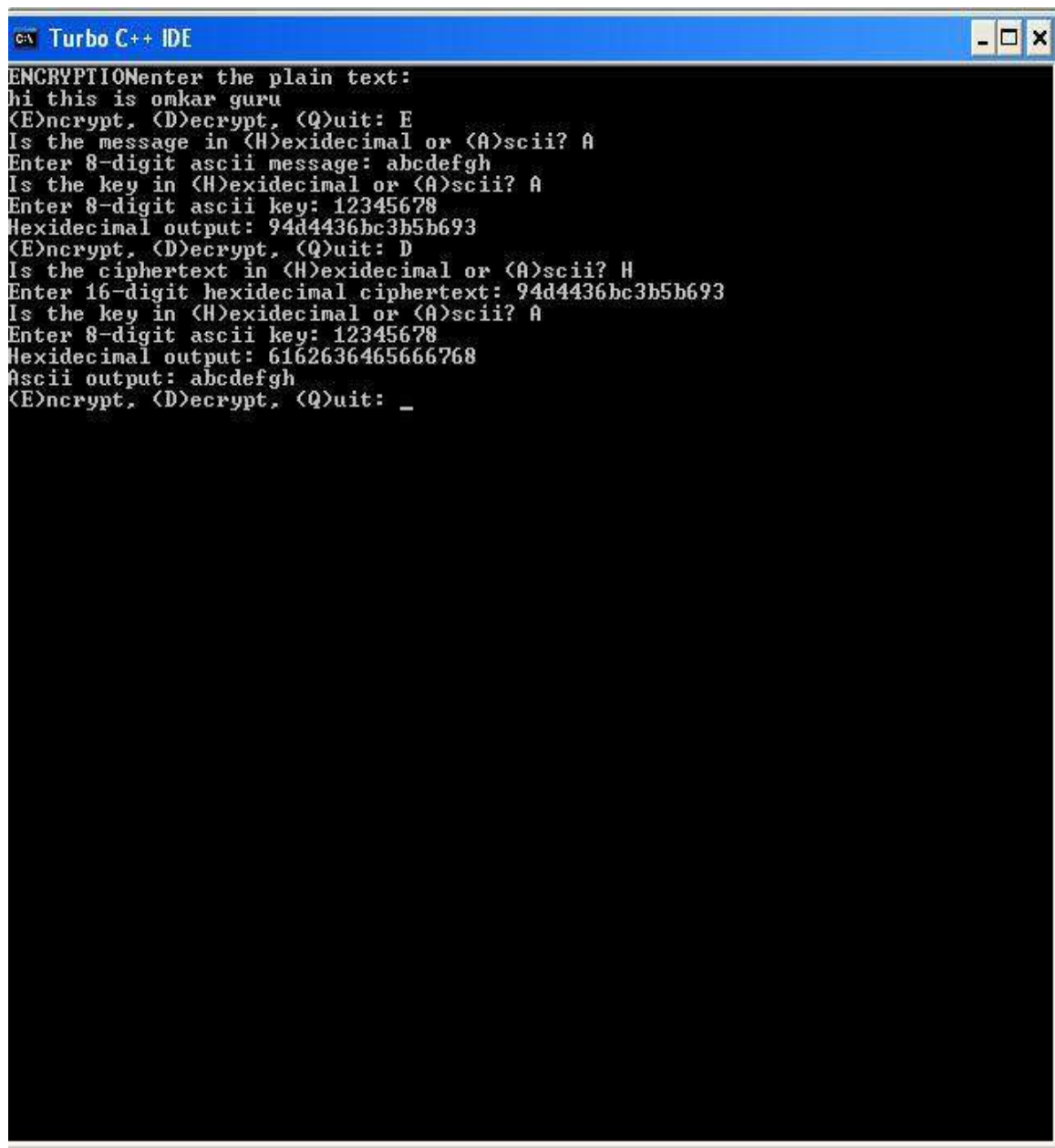
1. AES

AES is implemented in java platform with variable key length. The output is shown in an interactive applet interface.



2. DES

DES is implemented in C++. The user has choice of encryption or decryption. The input is taken of 64 bits. Either 8 ASCII characters or 16 HEX characters can be given for converting to cipher text. The key is again of 64 bits given by the user. The output (cipher text) is shown in hexadecimal. The cipher text can be decrypted using shared secret key.



```
ENCRYPTIONEnter the plain text:
hi this is omkar guru
(E)ncrypt, (D)ecrypt, (Q)uit: E
Is the message in (H)exidecimal or (A)scii? A
Enter 8-digit ascii message: abcdefgh
Is the key in (H)exidecimal or (A)scii? A
Enter 8-digit ascii key: 12345678
Hexidecimal output: 94d4436bc3b5b693
(E)ncrypt, (D)ecrypt, (Q)uit: D
Is the ciphertext in (H)exidecimal or (A)scii? H
Enter 16-digit hexadecimal ciphertext: 94d4436bc3b5b693
Is the key in (H)exidecimal or (A)scii? A
Enter 8-digit ascii key: 12345678
Hexidecimal output: 6162636465666768
Ascii output: abcdefgh
(E)ncrypt, (D)ecrypt, (Q)uit: _
```

5.2 PUBLIC KEY CRYPTOGRAPHY:

1. RSA

RSA algorithm is used for encryption and decryption (public key cryptography). Key length is variable up to 1024 bits.

The screenshot shows a Java applet window titled "Applet Viewer: RSA demo". The interface is as follows:

- Header:** "Applet Viewer: RSA demo" with standard window controls.
- Instruction:** "Enter prime 'p' and 'q' values or use the button below to generate them:"
- Inputs:**
 - p:** 20407
 - q:** 97579
 - Generate p and q** which are of average bit size: 16
 - n:** 1991294653
 - Calculate n**
 - e:** 241
 - Generate e** which is of bit size: 8
 - d:** 1272370153
 - Calculate d**
- Text Input:** "Enter text, numbers or encoded numbers below." with a text area containing "THIS IS RSA ENCRYPTION DECRYPTION ALGORITHM".
- Buttons:** "Convert to Number" and "Convert to Text".
- Outputs:**
 - Encrypted numbers: 11798187030112533086127896179, 55604947768719467234273799478, 02542194321380374842174419601.
 - Decrypted numbers: 16834568789915839440459373969, 49834553980658967807806000848, 97150651776082372036889334888.
- Buttons:** "Encrypt" and "Decrypt".
- Status:** "Applet started." at the bottom.

2. ECC

ECC algorithm is used for encryption and decryption. The program is implemented in JAVA. Elliptic curve public key and private key are generated on each instance and corresponding points on curve for given input generated and converted to cipher

```
Command Prompt
Enter the plain text
1234
Exception in thread "main" java.lang.ArithmeticException: BigInteger not invertible.
    at java.math.MutableBigInteger.modInverse(MutableBigInteger.java:1229)
    at java.math.MutableBigInteger.mutableModInverse(MutableBigInteger.java:1104)
    at java.math.BigInteger.modInverse(BigInteger.java:1980)
    at ECC.addpoints(ECC.java:230)
    at ECC.encrypt(ECC.java:159)
    at ECC.<init>(ECC.java:91)
    at ECC.main(ECC.java:310)

I:\Program Files\Java\jdk1.5.0_new\bin>java ECC

THE PRIME FIELD IS:167

THE ELLIPTIC CURVE IS:Y^2=X^3+73X+79

Enter the plain text
645645

Receiver's public key is <<103,6>,<130,89>>

Receiver's private key is 89

THE CIPHER TEXT IS
<140,84><40,108><140,84><64,8><140,84><121,25>

THE PLAIN TEXT IS
645645

I:\Program Files\Java\jdk1.5.0_new\bin>
```

5.3 DIGITAL SIGNATURE PROTOCOLS:

1. DSA

Digital signature algorithm is implemented. SHA1 is used for hashing

Applet Viewer: DSA

Applet

INPUT MESSAGE: HELO

HASH

THE MESSAGE AFTER HASHING IS :

461531014843358827295132685910662481060895531840

SIGN

THE VALUE OF r IS :

2347823498890890890834909023

THE VALUE OF s IS :

342349940-234990-940-99940-2390-423493490-2390-4

THE VALUE OF y IS :

45234890239048890490823488903892384903

To Be Verified

VERIFY

RESET

Applet started

2. AUTHENTICATION SYSTEM- RSA SIGNATURE USING MD5 FOR HASHING

The input message is hashed using MD5 to give 128 bit Message Digest. The message as well as message digest is encrypted using RSA.

The screenshot shows an applet window titled "Applet Viewer: Project". The applet interface is as follows:

- Input Message:** THIS IS RSA ENCRYPTION AND SIGNATURE USING MD5
- Hash:** 60d80484cc031db489a42af18c5c543a
- Signature:** 8076078284410562641845686105365198250967484667289
- Decrypted Signature:** 2451039417207391171350673757749710606675654884192
- Verifying Message:** 2451039417207391171350673757749710606675654884192
- Receive Message:** THIS IS RSA ENCRYPTION AND SIGNATURE USING MD5

The applet includes buttons for "Hash", "Sign", "Decrypt", and "Verify". The "Verify" button is currently disabled, indicated by a dotted border.

5.3 HILL CIPHER USING SELF-REPETITIVE MATRIX:

5.3.1 GENERAL STEPS:

1. SELECT A PRIME NUMBER WHOSE MODULAR ARITHMETIC WILL DETERMINE THE EVENTUAL COURSE OF DATA TRANSFER.
2. LIKE IN THIS CASE THE WHOLE ASCII CHARACTER SET IS TO TAKEN AS RESULT 97 HAVE BEEN CHOSEN AND ALL THE ASCII CHARACTERS CORRESPOND TO A PARTICULAR RESIDUE. E.G. 0=A, 1=B...
3. SELECT A VALUE OF N YOU LIKE TO WORK ON.
4. DECIDE THE BLOCK SIZE I.E. NO OF CHARACTERS TO BE TRANSFERRED AT THE SAME TIME.GREATER THE BLOCK SIZE GREATER WILL BE DIMENSIONS OF THE MATRIX...HARDER IT WILL BE TO FIND ITS INVERSE AND HARDER IT WILL BE TO CRACK THE CODE BY BRUTE FORCE. HERE THE BLOCK SIZE HAS BEEN CHOOSEN AS 5.
5. GENERATE A RANDOM DIAGONAL MATRIX 'A' WHOSE DIAGONAL ELEMENTS GIVE A N VALUE LCM OF 96.(AS PREVIOUSLY CHOSEN)
6. NOW SELECT A SQUARE INVERTIBLE MATRIX B AND GENERATE $C=B^{-1}AB$
7. NOW C IS THE DESIRED HILL CIPHER MATRIX.

5.3.2 STEPS AT THE TRANSMITTER SIDE:

1. TAKE THE DATA IN BLOCKS OF 5 AS A COLUMN MATRIX DENOTED BY E
2. MULTIPLY E THE MATRIX GENERATED EARLIER WITH $C^{(N-M)}$ TO GENERATE THE ENCRYPTION CODE. (C WAS GENERATED EARLIER)
3. M IS SOME RANDOM NUMBER SELECTED AND KNOWN TO BOTH THE ENDS. $M < N$
4. NOW CONVERT THIS CODE INTO MACHINE CODE TO GIVE BETTER COMPRESSION AND BIT SAVING.

NOTE: THE ALGORITHM FOR CONVERSION INTO MACHINE CODE HAS BEEN DISCUSSED LATER.

5. TRANSMIT (USING MANCHESTER CODING OR ELSE)

5.3.3 STEPS AT THE RECEIVER SIDE:

1. DECOMPRESS THE HEX CODE INTO MOD-97 CODE
2. THEN MULTIPLY THE CODE COLUMN MATRIX WITH C^M .
3. THE CORRESPONDING SUSTITUTIONS ARE MADE AND TEXT RECOVERED.

NOTE: MATHEMATICAL BACKGROUND FOR THE ABOVE TECHNIQUE HAS BEEN GIVEN IN THE SECTION FOR HILL-CIPHER PREVIOUSLY

5.3.4 METHOD FOR COMPRESSION INTO HEX CODE:

1. TAKE THE CODE IN THE PLACE VALUE OF 97 AND GENERATE A POLYNOMIAL.
2. NOW DIVIDE THE POLYNOMIAL BY 16 TO GENERATE A REMAINDER.
3. THE QUOTIENT GENERATED FORMS THE NEXT DIVIDEND POLYNOMIAL AND DIVISION IS CARRIED OUT ONCE MORE AND REMAINDER COLLECTED.
4. THIS PROCESS IS CARRIED ON UNTILL DIVISOR IS LARGER THAN THE DIVIDEND.
5. ALL THE REMAINDERS ARE COLLECTED AND THE ARRAY IS INVERTED. THIS IS THE COMPRESSED CODE.

5.3.5 A SMALL ILLUSTRATIVE EXAMPLE FOR HILL CIPHER USING SELF REPETITIVE MATRIX:

For easy illustration, we have taken a two block matrix, here to explain.

STEPS:

Take a plaintext of two blocks to be $P = [1\ 2]$

1. $MOD_VAL = 5$
2. Calculate N :
3. Assuming 2×2 diagonal matrix A
 $A = [2\ 0; 0\ 3]$
Now we have diagonal elements 2 and 3
 $2^8 \bmod 5 = 1$
 $3^4 \bmod 5 = 1$
Then LCM of (8, 4) gives the value of N . i.e The value of N is 8 in this case.
4. We are now interested in a matrix C whose value will be the same identity matrix if raised to the power N under MOD_VAL .
5. Select a random invertible matrix B .
We took $B = [2\ 1; 1\ 2]$
6. Now $C = BAB^{-1}$
 $C = [2\ 1; 1\ 2] * [2\ 0; 0\ 3] * [2\ 4; 4\ 2]$
 $C = [0\ 2; 3\ 0]$
Now C is the desired hill cipher matrix.
6. Select a random number $M < N$
Let $M = 3$

Steps at the transmitting side:

1. Encryption $E = C^{N-M} * P$

$$E = [0\ 2; 3\ 0]^{5*} [1\ 2]$$

$$E = [4\ 3]$$

E is the cipher text obtained.

Steps at the receiving side:

1. Decryption $D = C^M * E$

2. $D = [0 \ 2; 3 \ 0]^3 * [4 \ 3]$

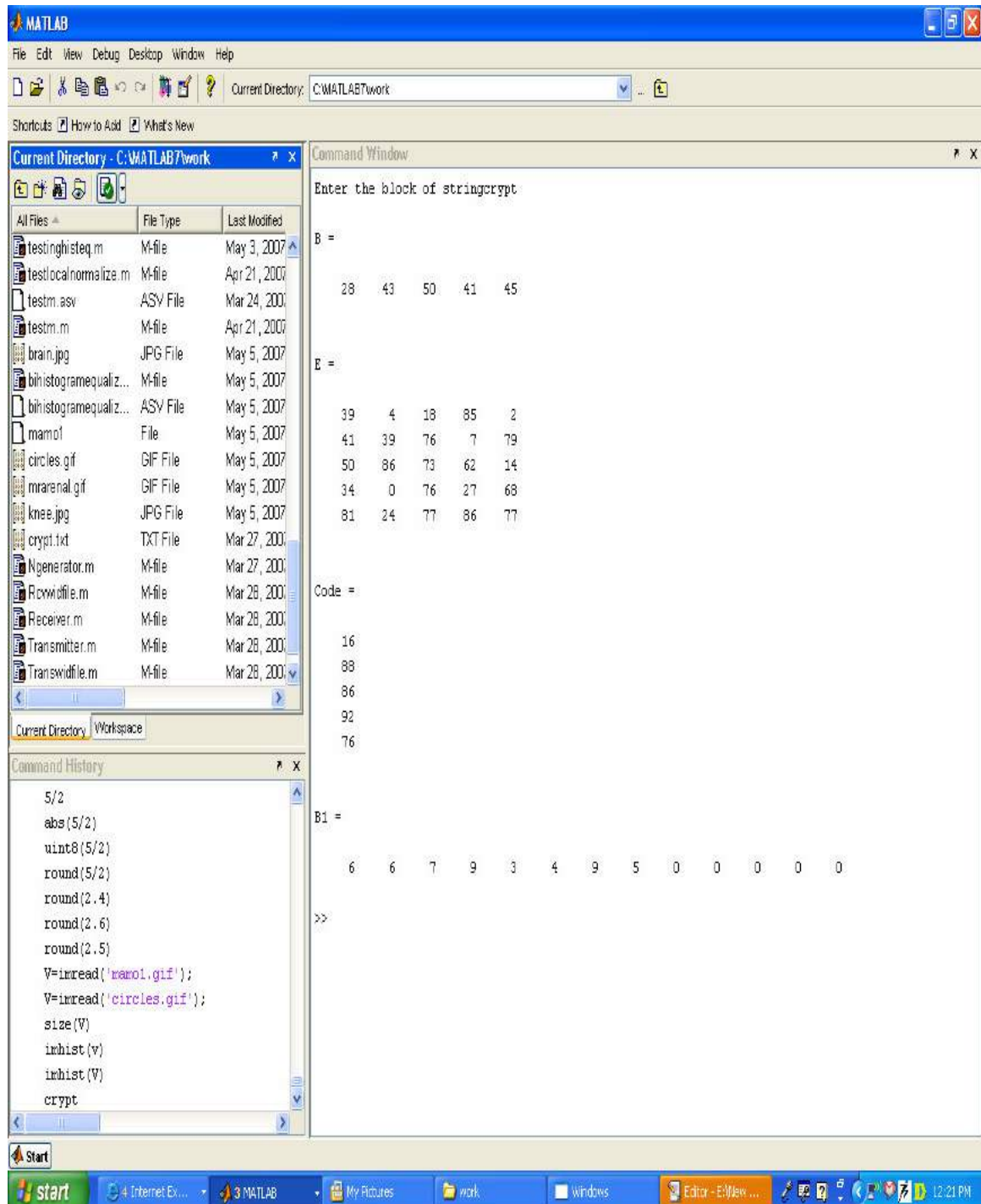
3. $D = [1 \ 2] = P$

Thus the plaintext P is obtained.

5.3.1 HILL CIPHER USING SELF-REPETITIVE MATRIX:

Hill cipher using self-repetitive matrix has been implemented in MATLAB. At first, we considered the message for 5 blocks and simulated the algorithm on both sender and receiver side.

1. TRANSMITTER SIDE



2. RECIEVER SIDE

The image shows a MATLAB workspace with the following components:

- File Explorer:** Shows the current directory as C:\MATLAB7\work. The file list includes:

All Files	File Type	Last Modified
testingshsteq.m	M-file	May 3, 2007
testlocalnormalize.m	M-file	Apr 21, 2007
testm.asv	ASV File	Mar 24, 2007
testm.m	M-file	Apr 21, 2007
brain.jpg	JPG File	May 5, 2007
bihistogramequalz...	M-file	May 5, 2007
bihistogramequalz...	ASV File	May 5, 2007
mamof	File	May 5, 2007
circles.gif	GIF File	May 5, 2007
mrarenal.gif	GIF File	May 5, 2007
knee.jpg	JPG File	May 5, 2007
crypt.txt	TXT File	Mar 27, 2007
Ngenerator.m	M-file	Mar 27, 2007
Rowwrite.m	M-file	Mar 28, 2007
Receiver.m	M-file	Mar 28, 2007
Transmitter.m	M-file	Mar 28, 2007
Transwritefile.m	M-file	Mar 28, 2007
- Command Window:** Displays the results of MATLAB commands:


```

lambda =
    0    5    9    4    3    9    7    6    6

B1 =
    16    88    86    92    76

E =
     1    48    73    96    24
    74    12    34    18     4
    93    14     9    44    75
    57    63    82    93    51
    17    68    35     0    28

Code =
    28
    43
    50
    41
    45

ins =

crypt

>>
      
```
- Command History:** Shows the sequence of commands entered:


```

5/2
abs(5/2)
uint8(5/2)
round(5/2)
round(2.4)
round(2.6)
round(2.5)
V=imread('mamof.gif');
V=imread('circles.gif');
size(V)
imhist(v)
imhist(V)
crypt
      
```

The Windows taskbar at the bottom shows the Start button, network status, 3 MATLAB instances, My Pictures, work folder, windows, Editor - E..., transmitter..., and system tray with the time 12:22 PM.

2. TRANSMITTING WITH FILE

The image shows the MATLAB software interface. The top menu bar includes File, Edit, Debug, Desktop, Window, and Help. The current directory is C:\MATLAB7\work. The file explorer on the left shows a list of files in the current directory, including testm.m, brain.jpg, and mam01.gif. The Command Window on the right displays the output of the 'crypt' function, showing a 10x10 matrix of binary values. The Command History window at the bottom left shows the sequence of commands entered, including 'crypt'.

Current Directory - C:\MATLAB7\work

All Files	File Type	Last Modified
testm.m	M-file	May 3, 2007
testlocalnormalize.m	M-file	Apr 21, 2007
testm.asv	ASV File	Mar 24, 2007
testm.m	M-file	Apr 21, 2007
brain.jpg	JPG File	May 5, 2007
bihistogramequalz...	M-file	May 5, 2007
bihistogramequalz...	ASV File	May 5, 2007
mam01	File	May 5, 2007
circles.gif	GIF File	May 5, 2007
mirarenal.gif	GIF File	May 5, 2007
knee.jpg	JPG File	May 5, 2007
crypt.txt	TXT File	Mar 27, 2007
Ngenerator.m	M-file	Mar 27, 2007
Rowwfile.m	M-file	Mar 28, 2007
Receiver.m	M-file	Mar 28, 2007
Transmitter.m	M-file	Mar 28, 2007
Transwfile.m	M-file	Mar 28, 2007

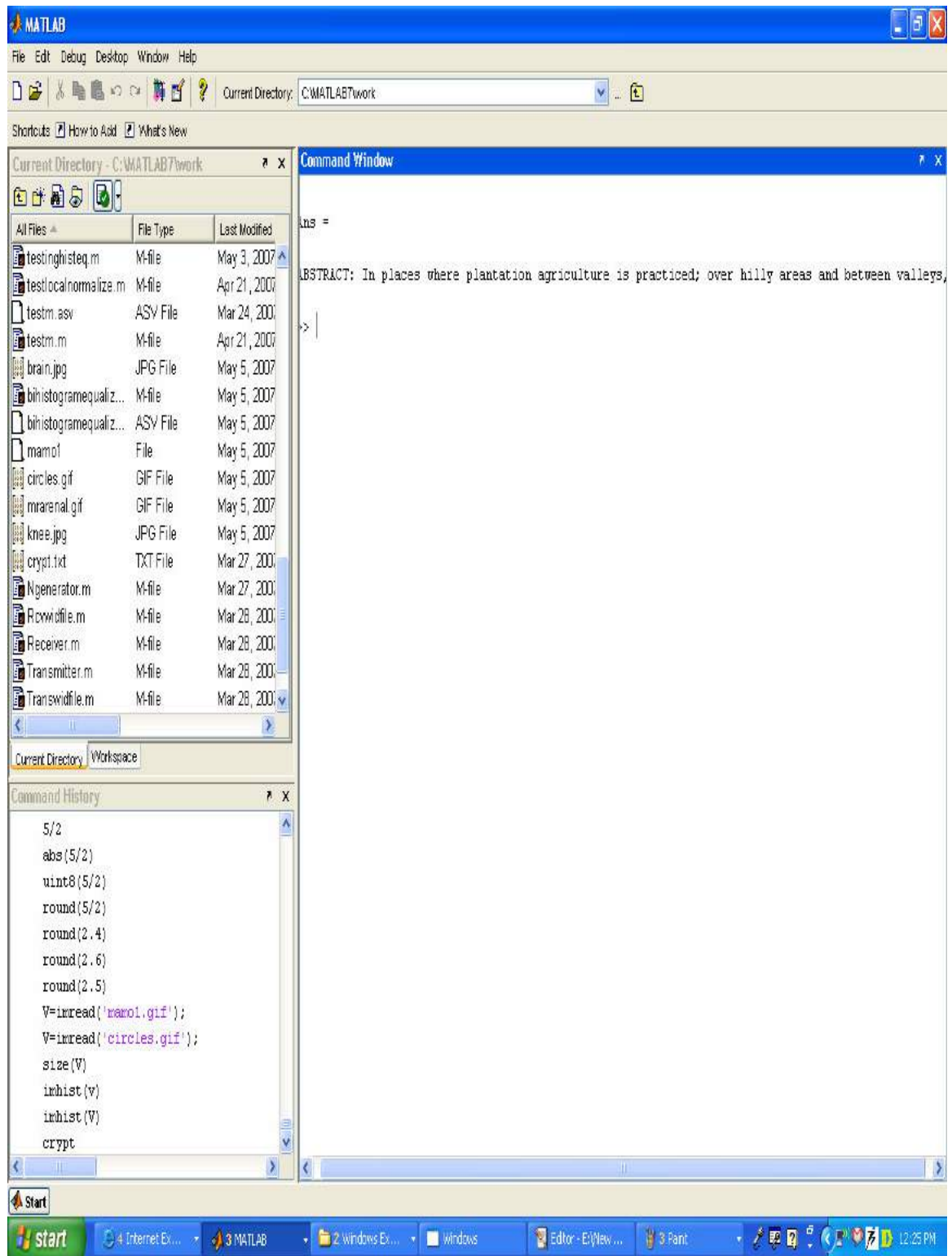
Command Window

```
oo
65
30
B1 =
    6    12    13    14    5    13    4    9    1    0    0    0    0
B =
    37    30    50    44    85
E =
    39    4    18    85    2
    41    39    76    7    79
    50    86    73    62    14
    34    0    76    27    68
    81    24    77    86    77
Code =
    68
    27
    67
    95
    48
B1 =
    1    7    11    8    3    5    8    6    1    0    0    0    0
>>
```

Command History

```
5/2
abs(5/2)
uint8(5/2)
round(5/2)
round(2.4)
round(2.6)
round(2.5)
V=imread('mam01.gif');
V=imread('circles.gif');
size(V)
imhist(v)
imhist(V)
crypt
```


3. RECEIVE WITH FILE



Chapter 6

CONCLUSION

CONCLUSION

The existing algorithms of symmetric key and public key cryptography are studied in detail and implemented using platforms like C++ and Java. A complete package which includes all the algorithms was developed.

The Hill cipher technique using self-repetitive matrix was successfully implemented in MATLAB. A communication channel was successfully modeled which used proper decompression techniques for effective communication. The numerical method suggested to find N value of a matrix was successfully tested and used in the implementation. It was found to be easier to compute and simpler to implement and difficult to crack.

REFERENCES

- [1] W.Stallings; “Cryptography and Network Security” 2nd Edition, Prentice Hall, 1999
- [2] Bruce Schneir: Applied Cryptography, 2nd edition, John Wiley & Sons, 1996
- [4] Abrams, M., and Podell, H. “Cryptography” Potentials, IEEE Page No 36-38. Issue: 1, Volume: 20, Feb-Mar, 2001
- [5] Eskioglu, A. Litwin,L “ Cryptography and Network Security” LOS Alamitos,CA: IEEE computer society press,1987
- [6] Garfinkel, S.L; “Public Key Cryptography”, Computer, IEEE, Volume: 29, Issue: 6, June 1996.
- [8] E.Biham and A.Shmir; “Differential C for Cryptanalysis of the Encryption Standard”; Springer- Verilag, 1993
- [9] Bidjos, J; “Threats to Private and Public Key Protection”, Comcon Spring '91. Digest of Papers, 25 Feb-1 March 1991
- [10] V. Miller; “Uses of Elliptic Curves in Cryptography. In advances in Cryptography, Springer Verlag Crypto 95.

