

STUDY AND IMPLEMENTATION OF CRYPTOGRAPHIC ALGORITHMS

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

**Bachelor of Technology
in
Electronics & Instrumentation Engineering**

**By
ASHISH PATEL
And
AJAY KUMAR GARG**



**Department of Electronics & Communication Engineering
National Institute of Technology
Rourkela
2008**

STUDY AND IMPLEMENTATION OF CRYPTOGRAPHIC ALGORITHMS

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

**Bachelor of Technology
in
Electronics & Instrumentation Engineering**

**By
ASHISH PATEL
And
AJAY KUMAR GARG**

**Under the Guidance of
Prof. G.S.Rath**



**Department of Electronics & Communication Engineering
National Institute of Technology
Rourkela
2008**



National Institute of Technology
Rourkela

CERTIFICATE

This is to certify that the thesis entitled, “STUDY AND IMPLEMENTATION OF CRYPTOGRAPHIC ALGORITHMS” submitted by Sri ASHISH PATEL and Sri AJAY KUMAR GARG in partial fulfillments for the requirements for the award of Bachelor of Technology Degree in Electronics & Instrumentation Engineering at National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by them 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. G. S. Rath
Dept. of Electronics & Communication Engg
National Institute of Technology
Rourkela - 769008

ACKNOWLEDGEMENT

We would like to articulate our deep gratitude to our project guide Prof. G.S.Rath who has always been our motivation for carrying out the project. An assemblage of this nature could never have been attempted without 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.

ASHISH PATEL

AJAY KUMAR GARG

CONTENTS

	Page No
Chapter 1	INTRODUCTION
	7
Chapter 2	NUMBER THEORY AND RNG
	10
2.1	Introduction
	11
2.2	Modular Arithmetic
	11
2.3	Euclidean Algorithm
	12
2.4	Prime Numbers
	13
2.5	Fermat's And Euler's Theorems
	14
2.6	The Chinese Remainder Theorem
	14
2.7	Random Number Generation
	15
Chapter 3	SYMMETRIC CIPHERS
	16
3.1	Introduction
	17
3.2	Classical Encryption Techniques
	17
3.3	Block Cipher Principles
	19
3.4	The Data Encryption Standard
	21
3.6	DES Design Criteria
	23
Chapter 4	PUBLIC KEY ENCRYPTION
	27
4.1	Introduction
	28
4.2	The Basic Principle
	28
4.3	Advantage and Disadvantage of Public Key
	29
4.4	The RSA Algorithm
	31
Chapter 5	IMPLEMENTATION RESULTS
	36
5.1	DES implementation in C++
	37

5.2	RSA implementation in MATLAB	38
	REFERENCES	39

Chapter 1

INTRODUCTION

Cryptography, defined as *the science and study of secret writing* concerns the ways in which communications and data can be encoded to prevent disclosure of their contents through eavesdropping or message interception, using codes, ciphers and other methods, so that only certain people can see the real message.

Security often requires that data be kept safe from unauthorized access. And the best line of defense is physical security (placing the machine to be protected behind physical walls). However, physical security is not always an option, due to cost and/or efficiency considerations. Instead, most computers are interconnected with each other openly, thereby exposing them and the communication channels that they use. With regards to confidentiality, cryptography is used to encrypt data residing on storage devices or traveling through communication channels to ensure that any illegal access is not successful. Also, cryptography is used to secure the process of authenticating different parties attempting any function on the system. Since a party wishing be granted a certain functionality on the system must present something that proves that they indeed who they say they are. That something is sometimes known as credentials and additional measures must be taken to ensure that these credentials are only used by their rightful owner. The most classic and obvious credential are passwords. Passwords are encrypted to protect against illegal usage.

Authorization is a layer built on top of authentication in the sense that the party is authenticated by presenting the credentials required (passwords, smart cards ... etc.). After the credentials are accepted the authorization process is started to ensure that the requesting party has the permissions to perform the functions needed.

Data integrity and Non-Repudiation are achieved by means of digital signature, a method that includes performing cryptography among other things.

Cryptography can essentially be classified into two types, the symmetric and asymmetric type. With a secret or symmetric key algorithm, the key is a shared secret between two communicating parties. Encryption and decryption both use the same key.

The Data Encryption Standard (DES) and the Advanced Encryption Standard (AES) are examples of symmetric key algorithms.

With a public key (PKA) or asymmetric key algorithm, a pair of keys is used. One of the keys, the private key, is kept secret and not shared with anyone. The other key, the public key, is not secret and can be shared with anyone. When data is encrypted by one of the keys, it can only be decrypted and recovered by using the other key. The two keys are mathematically related, but it is virtually impossible to derive the private key from the public key. The RSA algorithm is an example of a public key algorithm

Our project includes the study of the symmetric and asymmetric system of cryptography using DES as an example of symmetric system and RSA as an example of asymmetric system. DES was implemented in C++ while RSA was implemented in MATLAB.

Chapter 2

NUMBER THEORY AND RNG

2.1 Introduction

A number of mathematical concepts from number theory are essential in the design of cryptographic algorithms. This chapter provides an overview of the concepts along with the proofs of the theorems used in these algorithms. Random numbers play an important role in key generation for cryptographic algorithms. In this section, we also provide a brief overview of the use of random numbers and then look at some approaches to generating random numbers.

2.2 Modular Arithmetic

Modular arithmetic is a system of arithmetic for integers, where numbers *wrap around* after they reach a certain value - the *modulus*. Given any positive integer n and any nonnegative integer a , if we divide a by n , we get an integer quotient q and an integer remainder r that obey the following relationship:

$$a = qn + r \quad 0 \leq r < n; q = \lfloor a/n \rfloor$$

Where x is the largest integer less than or equal to $\lfloor x \rfloor$. The remainder r is often referred to as a residue. If a is an integer and n is a positive integer, we define $a \bmod n$ to be the remainder when a is divided by n . The integer n is called the modulus. Thus, for any integer a , we can always write:

$$a = \lfloor a/n \rfloor \times n + (a \bmod n) \quad \text{hence } 12 \bmod 7 = 5; \quad -12 \bmod 7 = 2$$

Two integers a and b are said to be congruent modulo n , if $(a \bmod n) = (b \bmod n)$. This is written as $a \equiv b \pmod{n}$, for example $73 \equiv 4 \pmod{23}$

Properties of Modular Arithmetic for Integers

Property	Expression
Commutative laws	$(w + x) \bmod n = (x + w) \bmod n$ $(w \times x) \bmod n = (x \times w) \bmod n$
Associative laws	$[(w + x) + y] \bmod n = [w + (x + y)] \bmod n$ $[(w \times x) \times y] \bmod n = [w \times (x \times y)] \bmod n$
Distributive laws	$[w + (x \times y)] \bmod n = [(w \times x) + (w \times y)] \bmod n$ $[w \times (x + y)] \bmod n = [(w \times x) + (w \times y)] \bmod n$
Identities	$(0 + w) \bmod n = w \bmod n$ $(1 \times w) \bmod n = w \bmod n$
Additive inverse (-w)	For each $w \in \mathbb{Z}_n$, there exists a z such that $w + z \equiv 0 \bmod n$

2.3 Euclidean Algorithm

Euclidean algorithm is a simple procedure for determining the greatest common divisor of two positive integers. Nonzero b is defined to be a divisor of a if $a = mb$ for some m , where a , b , and m are integers. We will use the notation $\gcd(a, b)$ to mean the greatest common divisor of a and b . The positive integer c is said to be the greatest common divisor of a and b if c is a divisor of a and of b and any divisor of a and b is a divisor of c .

An equivalent definition is the following: $\gcd(a, b) = \max[k, \text{such that } k|a \text{ and } k|b]$

Because we require that the greatest common divisor be positive, $\gcd(a, b) = \gcd(|a|, |b|)$.

Also, because all nonzero integers divide 0, we have $\gcd(a, 0) = |a|$.

We stated that two integers a and b are relatively prime if their only common positive integer factor is 1. This is equivalent to saying that a and b are relatively prime if $\gcd(a, b) = 1$.

The Euclidean algorithm is based on the following theorem: For any nonnegative integer a and any positive integer b , $\gcd(a, b) = \gcd(b, a \bmod b)$

2.4 Prime Numbers

A central concern of number theory is the study of prime numbers. Indeed, whole books have been written on the subject. An integer $p > 1$ is a prime number if and only if its only divisors are ± 1 and $\pm p$. Prime numbers play a critical role in number theory and in the cryptographic techniques discussed later.

If P is the set of all prime numbers, then any positive integer a can be written uniquely in the following form:

$$a = \prod_{p \in P} p^{a_p} \quad \text{where each } a_p \geq 0$$

The right-hand side is the product over all possible prime numbers p ; for any particular value of a , most of the exponents a_p will be 0.

It is easy to determine the greatest common divisor of two positive integers if we express each integer as the product of primes.

Example:

$$300 = 2^2 \times 3^1 \times 5^2$$

$$18 = 2^1 \times 3^2$$

$$\gcd(18, 300) = 2^1 \times 3^1 \times 5^0 = 6$$

For many cryptographic algorithms, it is necessary to select one or more very large prime numbers at random. Thus we are faced with the task of determining whether a given large number is prime. There is no simple yet efficient means of accomplishing this task.

Miller-Rabin Algorithm yields a number that is not necessarily a prime. However, the algorithm can yield a number that is almost certainly a prime. It is based on the conclusion that if n is prime, then either the first element in the list of residues, or

remainders, $(a^q, a^{2q}, \dots, a^{2^{k-1}q}, a^{2^kq})$ modulo n equals 1, or some element in the list equals $(n-1)$; otherwise n is composite (i.e., not a prime). On the other hand, if the condition is met, that does not necessarily mean that n is prime. For example, if $n = 2047 = 23 \times 89$, then $n-1 = 2 \times 1023$. Computing, $2^{1023} \bmod 2047 = 1$, so that 2047 meets the condition but is not prime.

2.5 Fermat's And Euler's Theorems

Two theorems that play important roles in public-key cryptography are Fermat's theorem and Euler's theorem. Fermat's theorem states the following: *If p is prime and a is a positive integer not divisible by p , then* $a^{p-1} \equiv 1 \pmod{p}$

Before presenting Euler's theorem, we need to introduce an important quantity in number theory, referred to as *Euler's totient function* and written $\phi(n)$, defined as the number of positive integers less than n and relatively prime to n . By convention, $\phi(1) = 1$.

Example : $\phi(37)=36$ and $\phi(35) = 24$

For a prime number p , $\phi(p) = p-1$

Now suppose that we have two prime numbers p and q , with p not equal q . Then it can be show that for $n = pq$, $\phi(n) = \phi(pq) = \phi(p) \times \phi(q) = (p-1) \times (q-1)$

Euler's theorem states that for every a and n that are relatively prime:
 $a^{\phi(n)} \equiv 1 \pmod{n}$

Example : $a = 2; n = 11; \phi(11) = 10$ $a^{\phi(n)} = 2^{10} = 1024 \equiv 1 \pmod{11} = 1 \pmod{n}$

2.6 The Chinese Remainder Theorem

In essence the Chinese remainder theorem (CRT) says it is possible to reconstruct integers in a certain range from their residues modulo a set of pair wise relatively prime moduli.

The integers 0 through 9 in Z_{10} , can be reconstructed from their two residues modulo 2 and 5 (the relatively prime factors of 10). Say the known residues of a decimal digit x $\text{mod } 2 = 0$ and $x \text{ mod } 5 = 3$. Therefore, x is an even integer in Z_{10} whose remainder, on division by 5, is 3. The unique solution is $x = 8$.

One of the useful features of the Chinese remainder theorem is that it provides a way to manipulate (potentially very large) numbers mod M in terms of tuples of smaller numbers. This can be useful when M is 150 digits or more. However, note that it is necessary to know beforehand the factorization of M .

2.7 Random Number Generation

Traditionally, the concern in the generation of a sequence of allegedly random numbers has been that the sequence of numbers be random in some well-defined statistical sense. The following two criteria are used to validate that a sequence of numbers is random:

- *Uniform distribution*: The distribution of numbers in the sequence should be uniform; that is, the frequency of occurrence of each of the numbers should be approximately the same.
- *Independence*: No one value in the sequence can be inferred from the others.

Cryptographic applications typically make use of algorithmic techniques for random number generation. These algorithms are deterministic and therefore produce sequences of numbers that are not statistically random. However, if the algorithm is good, the resulting sequences will pass many reasonable tests of randomness. Such numbers are referred to as **pseudorandom numbers**. For cryptographic applications, it makes some sense to take advantage of the encryption logic available to produce random numbers. A number of means have been used, namely *Cyclic Encryption* method, the *DES Output Feedback Mode* method and the *ANSI X9.17 PRNG* method.

Chapter 3

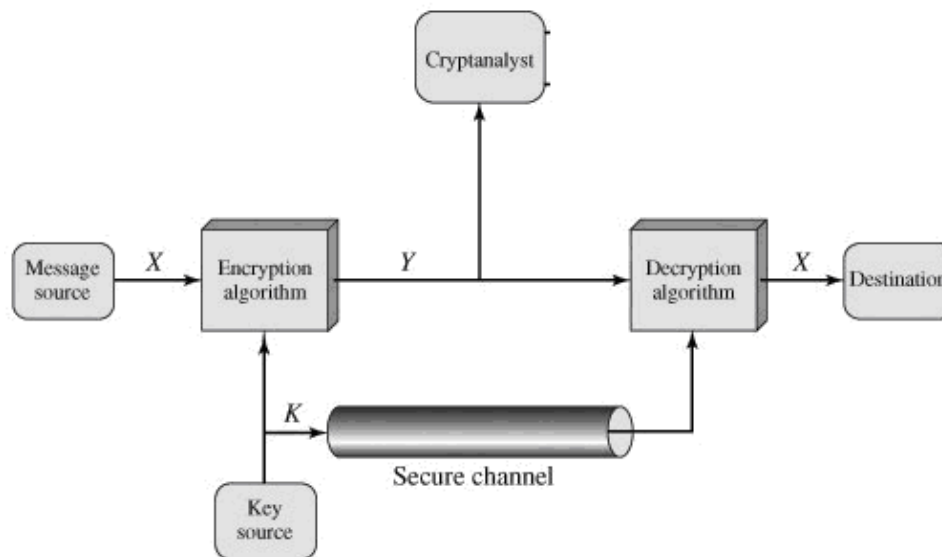
SYMMETRIC CIPHERS

3.1 Introduction

Symmetric encryption is referred to as Conventional 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.

3.2 Classical Encryption Techniques

We examine a variety of algorithms in use before the computer era, before we start discussing DES, the most widely used symmetric cipher algorithm.



Conventional encryption model

The basic structure of all the ciphers is same. A source produces a message in plaintext, X . For encryption, a key K is generated. If the key is generated at the message source, then it must also be provided to the destination. With the message X and the encryption key K as input, the encryption algorithm forms the cipher text $Y = E(K, X)$. The intended receiver, in possession of the key, is able to invert the transformation: $X = D(K, Y)$ to receive the transmitted message.

Caesar Cipher

The Caesar cipher involves replacing each letter of the alphabet with the letter standing n places further down the alphabet.

Monoalphabetic Ciphers

Caesar cipher is far from secure. A dramatic increase in the key space can be achieved by allowing an arbitrary substitution. A single cipher alphabet (mapping from plain alphabet to cipher alphabet) is used per message.

Playfair Cipher

The best-known multiple-letter encryption cipher is the Playfair, which treats digrams in the plaintext as single units and translates these units into ciphertext digrams

Hill Cipher

The encryption algorithm takes m successive plaintext letters and substitutes for them m ciphertext letters. The substitution is determined by m linear equations in which each character is assigned a numerical value ($a = 0, b = 1 \dots z = 25$). For $m = 3$, the system can be described as follows:

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \mod 26$$

Or $\mathbf{C} = \mathbf{KP} \mod 26$, where \mathbf{C} and \mathbf{P} are column vectors of length 3, representing the plaintext and cipher text, and \mathbf{K} is a 3×3 matrix, representing the encryption key.

$\mathbf{C} = \mathbf{E}(\mathbf{K}, \mathbf{P}) = \mathbf{KP} \mod 26$ for decryption we get $\mathbf{P} = \mathbf{D}(\mathbf{K}, \mathbf{P}) = \mathbf{K}^{-1}\mathbf{C} \mod 26 = \mathbf{K}^{-1}\mathbf{KP} = \mathbf{P}$ where \mathbf{K}^{-1} is the matrix inverse of \mathbf{K} .

Polyalphabetic Ciphers

A set of related monoalphabetic substitution rules is used. A key determines which particular rule is chosen for a given transformation.

One-Time Pad

A random key that is as long as the message, so that the key need not be repeated. In addition, the key is to be used to encrypt and decrypt a single message, and then is discarded. Each new message requires a new key of the same length as the new message. Such a scheme, known as a one-time pad

Transposition Techniques

A very different kind of mapping is achieved by performing some sort of permutation on the plaintext letters, simplest such cipher is the rail fence technique, in which the plaintext is written down as a sequence of diagonals and then read off as a sequence of rows

Rotor Machines

Before the introduction of DES, the most important application of the principle of multiple stages of encryption was a class of systems known as rotor machines. The machine consists of a set of independently rotating cylinders through which electrical pulses can flow. Each cylinder has 26 input pins and 26 output pins, with internal wiring that connects each input pin to a unique output pin.

Steganography

The methods of steganography conceal the existence of the message. A simple form of steganography, is one in which an arrangement of words or letters within an apparently innocuous text spells out the real message. For example, the sequence of first letters of each word of the overall message spells out the hidden message

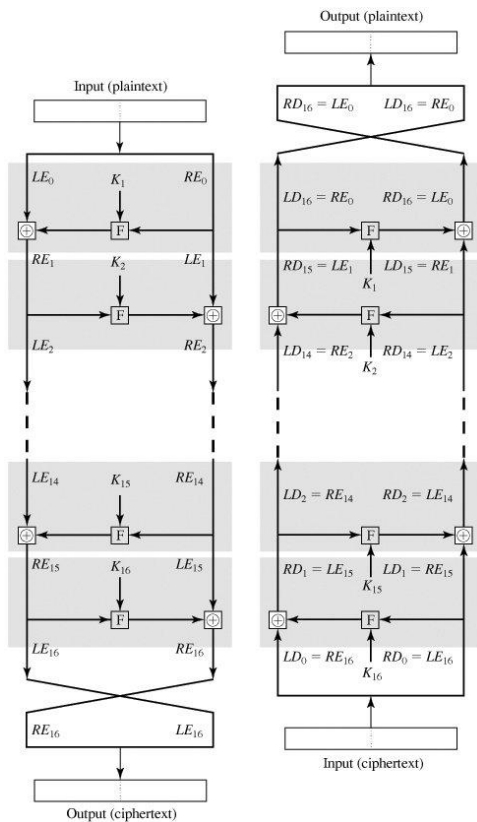
3.3 Block Cipher Principles

Most symmetric block encryption algorithms in current use are based on a structure referred to as a Feistel block cipher. A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits. There are 2^n possible different plaintext blocks and, for the encryption to be reversible (i.e., for decryption to be possible), each must

produce a unique ciphertext block. Such a transformation is called reversible, or nonsingular.

Feistel proposed that we can approximate the ideal block cipher by utilizing the concept of a product cipher, which is the execution of two or more simple ciphers in sequence in such a way that the final result or product is cryptographically stronger than any of the component ciphers. The essence of the approach is to develop a block cipher with a key length of k bits and a block length of n bits, allowing a total of 2^k possible transformations, rather than the $2^n!$ Transformations available with the ideal block cipher.

In particular, Feistel proposed the use of a cipher that alternates substitutions and permutations. In fact, this is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates confusion and diffusion functions.



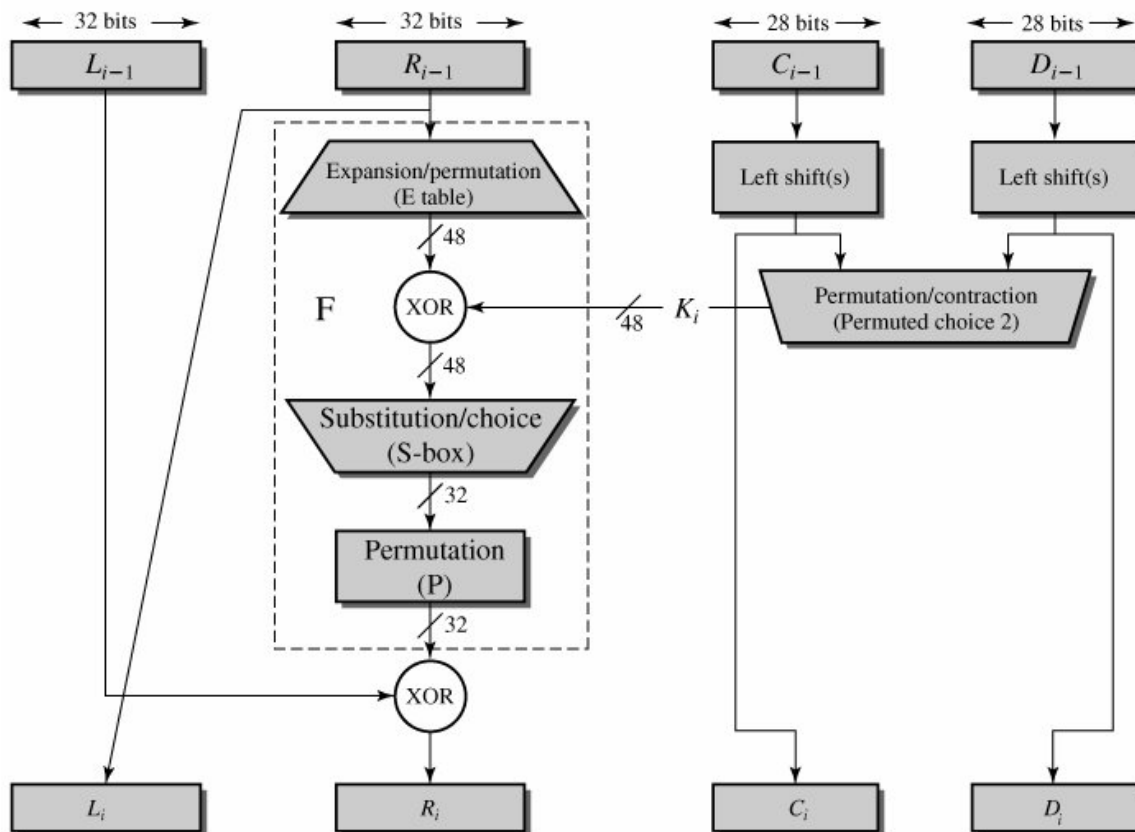
The figure given alongside depicts the structure proposed by Feistel. They are a plaintext block of length $2w$ bits and a key K . The plaintext block is divided into two halves, L_0 and R_0 . The two halves of the data pass through n rounds of processing and then combine to produce the ciphertext block. Each round i has as inputs L_{i-1} and R_{i-1} , derived from the previous round, as well as a subkey K_i , derived from the overall K . In general, the subkeys K_i are different from K and from each other. All rounds have the same structure. A **substitution** is performed on the left half of the data. This is done by applying a *round function* F to the right half of the data and then

taking the exclusive-OR of the output of that function and the left half of the data. The round function has the same general structure for each round but is parameterized by the round subkey K_i . Following this substitution, a **permutation** is performed that consists of

the interchange of the two halves of the data. The process of decryption with a Feistel cipher is essentially the same as the encryption process. The rule is as follows: Use the ciphertext as input to the algorithm, but use the subkeys K_i in reverse order. That is, use K_n in the first round, K_{n-1} in the second round, and so on until K_1 is used in the last round. This is a nice feature because it means we need not implement two different algorithms, one for encryption and one for decryption.

3.4 The Data Encryption Standard

The most widely used encryption scheme is based on the Data Encryption Standard (DES) adopted in 1977 by the National Bureau of Standards, now the National Institute of Standards and Technology (NIST), as Federal Information Processing Standard 46 (FIPS PUB 46). The algorithm itself is referred to as the Data Encryption Algorithm (DEA). For DES, data are encrypted in 64-bit blocks using a 56-bit key. The algorithm transforms 64-bit input in a series of steps into a 64-bit output. The same steps, with the same key, are used to reverse the encryption.

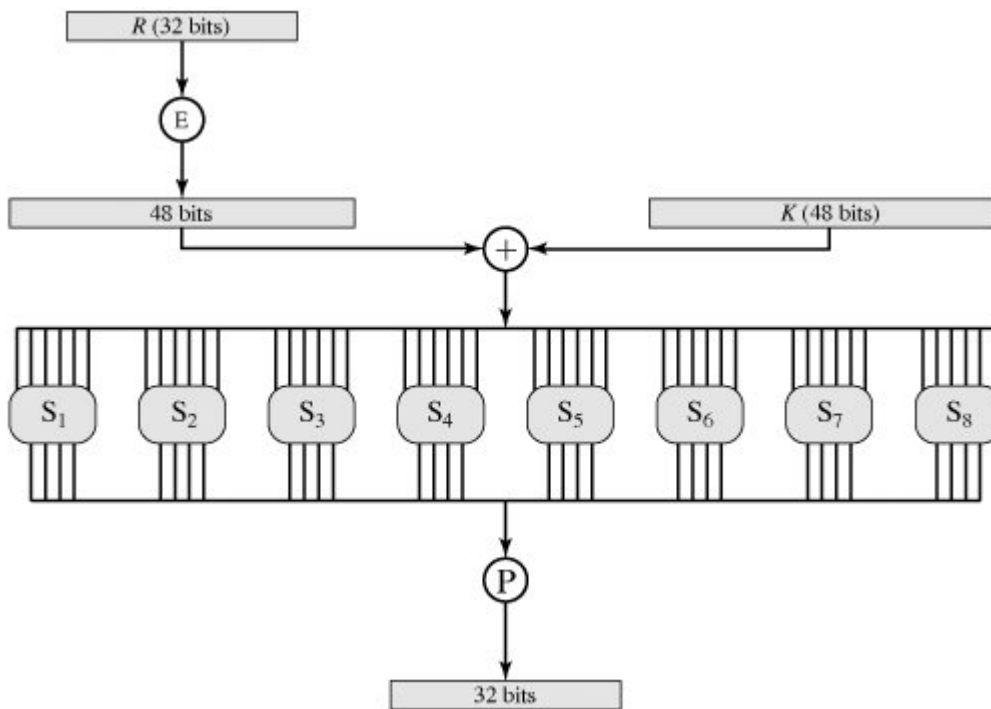


The above shows the internal structure of a single round of DES. The left and right halves of each 64-bit intermediate value are treated as separate 32-bit quantities, labeled L (left) and R (right). As in any classic Feistel cipher, the overall processing at each round can be summarized in the following formulas:

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$

The round key K_i is 48 bits. The R input is 32 bits. This R input is first expanded to 48 bits by using a table that defines a permutation plus an expansion that involves duplication of 16 of the R bits. The resulting 48 bits are XORed with K_i . This 48-bit result passes through a substitution function that produces a 32-bit output, which is permuted as defined by a table. The role of the S-boxes in the function F is illustrated in the following figure.



The substitution consists of a set of eight S-boxes, each of which accepts 6 bits as input and produces 4 bits as output. These transformations are defined in a given table, which is interpreted as follows: The first and last bits of the input to box S_i form a 2-bit binary number to select one of four substitutions defined by the four rows in the table for S_i . The

middle four bits select one of the sixteen columns. The decimal value in the cell selected by the row and column is then converted to its 4-bit representation to produce the output.

Key Generation

Returning to earlier figure, we see that a 64-bit key is used as input to the algorithm. The bits of the key are numbered from 1 through 64; every eighth bit is ignored. The key is first subjected to a permutation governed by a table labeled Permuted Choice One. The resulting 56-bit key is then treated as two 28-bit quantities, labeled C_0 and D_0 . At each round, C_{i-1} and D_{i-1} are separately subjected to a circular left shift, or rotation, of 1 or 2 bits, as governed by another table. These shifted values serve as input to the next round. They also serve as input to Permuted Choice Two, which produces a 48-bit output that serves as input to the function $F(R_{i-1}, K_i)$.

DES Decryption

Decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed.

3.6 DES Design Criteria

The criteria used in the design of DES, focused on the design of the S-boxes and on the P function that takes the output of the S boxes. The criteria for the S-boxes are as follows:

- No output bit of any S-box should be too close a linear function of the input bits. Specifically, if we select any output bit and any subset of the six input bits, the fraction of inputs for which this output bit equals the XOR of these input bits should not be close to 0 or 1, but rather should be near 1/2.
- Each row of an S-box (determined by a fixed value of the leftmost and rightmost input bits) should include all 16 possible output bit combinations.
- If two inputs to an S-box differ in exactly one bit, the outputs must differ in at least two bits.
- If two inputs to an S-box differ in the two middle bits exactly, the outputs must differ in at least two bits.

- If two inputs to an S-box differ in their first two bits and are identical in their last two bits, the two outputs must not be the same.
- For any nonzero 6-bit difference between inputs, no more than 8 of the 32 pairs of inputs exhibiting that difference may result in the same output difference.
- This is a criterion similar to the previous one, but for the case of three S-boxes.

The criteria for the permutation P are as follows:

- The four output bits from each S-box at round i are distributed so that two of them affect (provide input for) "middle bits" of round $(i + 1)$ and the other two affect end bits. The two middle bits of input to an S-box are not shared with adjacent S-boxes. The end bits are the two left-hand bits and the two right-hand bits, which are shared with adjacent S-boxes.
- The four output bits from each S-box affect six different S-boxes on the next round, and no two affect the same S-box.
- For two S-boxes j, k , if an output bit from S_j affects a middle bit of S_k on the next round, then an output bit from S_k cannot affect a middle bit of S_j . This implies that for $j = k$, an output bit from S_j must not affect a middle bit of S_j .

Number of Rounds

The greater the number of rounds, the more difficult it is to perform cryptanalysis, even for a relatively weak F. In general, the criterion should be that the number of rounds is chosen so that known cryptanalytic efforts require greater effort than a simple brute-force key search attack.

Design of Function F

The heart of a Feistel block cipher is the function F, this function relies on the use of S-boxes. Function F provides the element of confusion in a Feistel cipher. Thus, it must be difficult to "unscramble" the substitution performed by F. One obvious criterion is that F be nonlinear. The more nonlinear F, the more difficult any type of cryptanalysis will be. We would like the algorithm to have good avalanche

properties, in general, this means that a change in one bit of the input should produce a change in many bits of the output. Another criterion is the bit independence criterion (BIC), which states that output bits j and k should change independently when any single input bit i is inverted, for all i , j , and k .

S-Box Design

One of the most intense areas of research in the field of symmetric block ciphers is that of S-box design. In essence, we would like any change to the input vector to an S-box to result in random-looking changes to the output. The relationship should be nonlinear and difficult to approximate with linear functions. One obvious characteristic of the S-box is its size. An $n \times m$ S-box has n input bits and m output bits. DES has 6×4 S-boxes.. Larger S-boxes, by and large, are more resistant to differential and linear cryptanalysis. On the other hand, the larger the dimension n , the (exponentially) larger the lookup table. Thus, for practical reasons, a limit of n equal to about 8 to 10 is usually imposed. Another practical consideration is that the larger the S-box, the more difficult it is to design it properly. All linear combinations of S-box columns should be bent. Bent functions are a special class of Boolean functions that are highly nonlinear according to certain mathematical criteria. For larger S-boxes, such as 8×32 , the question arises as to the best method of selecting the S-box entries in order to meet the type of criteria we have been discussing. Nyberg, suggests the following approaches

- Random: Use some pseudorandom number generation or some table of random digits to generate the entries in the S-boxes. This may lead to boxes with undesirable characteristics for small sizes (e.g., 6×4) but should be acceptable for large S-boxes (e.g., 8×32).
- Random with testing: Choose S-box entries randomly, then test the results against various criteria, and throw away those that do not pass.
- Human-made: This is a more or less manual approach with only simple mathematics to support it. This approach is difficult to carry through for large S-boxes.

- Math-made: Generate S-boxes according to mathematical principles. By using mathematical construction, S-boxes can be constructed that offer proven security against linear and differential cryptanalysis, together with good diffusion.

Key Schedule Algorithm

A final area of block cipher design, and one that has received less attention than S-box design, is the key schedule algorithm. With any Feistel block cipher, the key is used to generate one subkey for each round. In general, we would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key.

Chapter 4

PUBLIC-KEY CRYPTOGRAPHY

4.1 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.

Advantage 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.

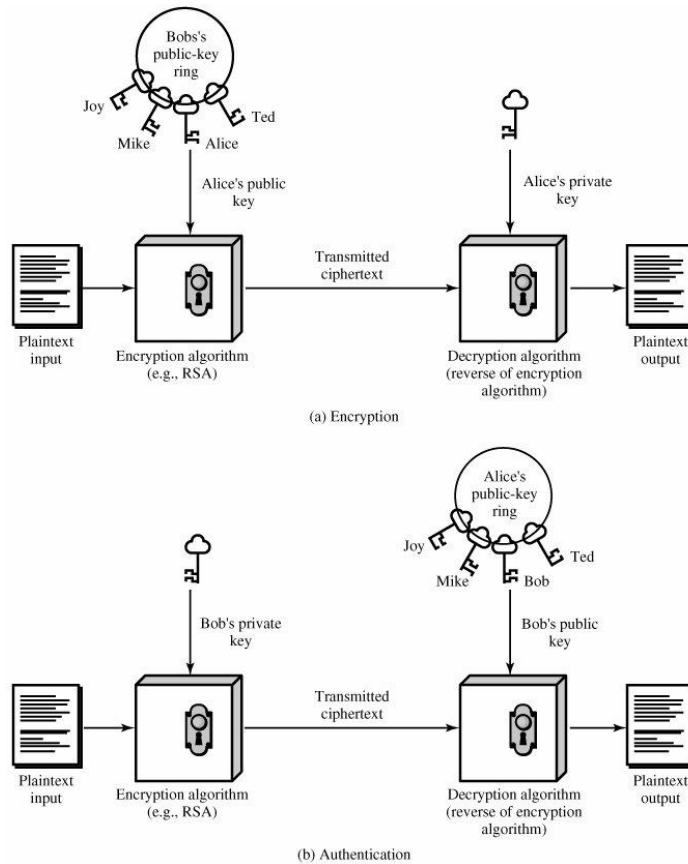
4.2 The Basic Principle:

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 below illustrates the public-key encryption process.



4.3 Advantage and Disadvantage of Public-Key Cryptography Compared with Secret-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.
5. 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.

6. 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.

4.4 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. The RSA scheme is a block cipher in which the plaintext and ciphertext are integers between 0 and $n-1$ for some n . A typical size for n is 1024 bits, or 309 decimal digits. That is, n is less than 2^{1024} .

Description of the Algorithm

The scheme makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n . That is, the block size must be less than or equal to $\log_2(n)$; in practice, the block size is i bits, where $2^i < n < 2^{i+1}$. Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C :

$$C = M^e \bmod n$$

$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know the value of n . The sender knows the value of e , and only the receiver knows the value of d . Thus, this is a public-key encryption algorithm with a public key of $PU = \{e, n\}$ and a private key of $PU = \{d, n\}$. For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:

1. It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$.
2. It is relatively easy to calculate $M^e \bmod n$ and C^d for all values of $M < n$.
3. It is infeasible to determine d given e and n .

We need to find a relationship of the form $M^{ed} \bmod n = M$

The preceding relationship holds if e and d are multiplicative inverses modulo $\phi(n)$, where $\phi(n)$ is the Euler totient function. For p, q prime, $\phi(pq) = (p-1)(q-1)$. The relationship between e and d can be expressed as

$$ed \bmod \phi(n) = 1$$

This is equivalent to saying

$$ed \equiv 1 \bmod \phi(n) \quad \text{and} \quad d \equiv e^{-1} \bmod \phi(n)$$

That is, e and d are multiplicative inverses mod $\phi(n)$. According to the rules of modular arithmetic, this is true only if d (and therefore e) is relatively prime to $\phi(n)$. Equivalently, $\gcd(\phi(n), d) = 1$

Key Generation	
Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption	
Ciphertext:	C
Plaintext:	$M = C^d \pmod{n}$

Exponentiation in Modular Arithmetic

Both encryption and decryption in RSA involve raising an integer to an integer power, mod n . If the exponentiation is done over the integers and then reduced modulo n , the intermediate values would be gargantuan, we can make use of a property of modular arithmetic:

$$[(a \pmod{n}) \times (b \pmod{n})] \pmod{n} = (a \times b) \pmod{n}$$

Thus, we can reduce intermediate results modulo n . This makes the calculation practical.

Another consideration is the efficiency of exponentiation, because with RSA we are dealing with potentially large exponents. To see how efficiency might be increased,

consider that we wish to compute x^{16} . A straightforward approach requires 15 multiplications:

However, we can achieve the same final result with only four multiplications if we repeatedly take the square of each partial result, successively forming x^2, x^4, x^8, x^{16} . As another example, suppose we wish to calculate $x^{11} \bmod n$ for some integers x and n . Observe that $x^{11} = x^{1+2+8} = (x)(x^2)(x^8)$. In this case we compute $x \bmod n, x^2 \bmod n, x^4 \bmod n$, and $x^8 \bmod n$ and then calculate $[(x \bmod n) \times (x^2 \bmod n) \times (x^8 \bmod n)] \bmod n$

Efficient Operation Using the Public Key

To speed up the operation of the RSA algorithm using the public key, a specific choice of e is usually made. The most common choice is 65537 ($2^{16} + 1$); two other popular choices are 3 and 17. Each of these choices has only two 1 bits and so the number of multiplications required to perform exponentiation is minimized.

However, with a very small public key, such as $e = 3$, RSA becomes vulnerable to a simple attack. It is required that during key generation the user selects a value of e that is relatively prime to $\phi(n)$. Thus, for example, if a user has preselected $e = 65537$ and then generated primes p and q , it may turn out that $\gcd(\phi(n), e) \neq 1$. Thus, the user must reject any value of p or q that is not congruent to 1 (mod 65537).

Key Generation

Before the application of the public-key cryptosystem, each participant must generate a pair of keys. This involves the following tasks:

- Determining two prime numbers, p and q
- Selecting either e or d and calculating the other

First, consider the selection of p and q . Because the value of $n = pq$ will be known to any potential adversary, to prevent the discovery of p and q by exhaustive methods, these primes must be chosen from a sufficiently large set (i.e., p and q must be large numbers). On the other hand, the method used for finding large primes must be reasonably efficient.

In summary, the procedure for picking a prime number is as follows.

- Pick an odd integer n at random (e.g., using a pseudorandom number generator).
- Pick an integer $a < n$ at random.
- Perform the probabilistic primality test, such as Miller-Rabin, with a as a parameter. If n fails the test, reject the value n and go to step 1.
- If n has passed a sufficient number of tests, accept n ; otherwise, go to step 2.

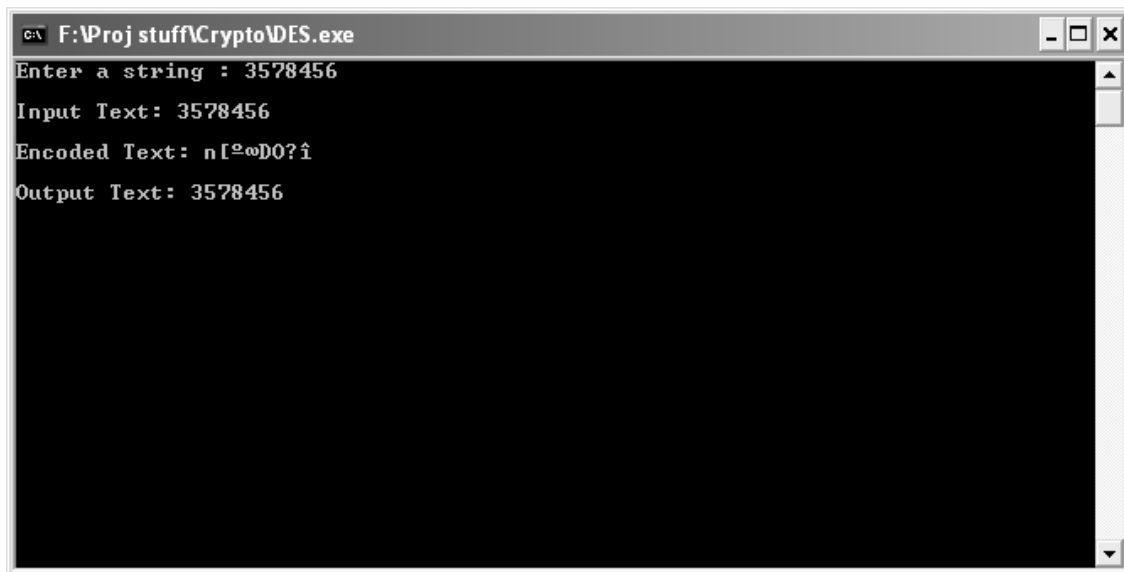
This is a somewhat tedious procedure. However, remember that this process is performed relatively infrequently: only when a new pair (PU, PR) is needed.

Having determined prime numbers p and q , the process of key generation is completed by selecting a value of e and calculating d or, alternatively, selecting a value of d and calculating e . Assuming the former, then we need to select an e such that $\gcd(\phi(n), e) = 1$ and then calculate $d \equiv e^{-1} \pmod{\phi(n)}$. Fortunately, there is a single algorithm that will, at the same time, calculate the greatest common divisor of two integers and, if the gcd is 1, determine the inverse of one of the integers modulo the other. The algorithm is referred to as the extended Euclid's algorithm. Thus, the procedure is to generate a series of random numbers, testing each against $\phi(n)$ until a number relatively prime to $\phi(n)$ is found.

Chapter 5

IMPLEMENTATION RESULTS

5.1 DES implementation in C++



```
C:\ F:\Proj stuff\Crypto\DES.exe
Enter a string : 3578456
Input Text: 3578456
Encoded Text: n[°∞DO?i
Output Text: 3578456
```

Sample Input:

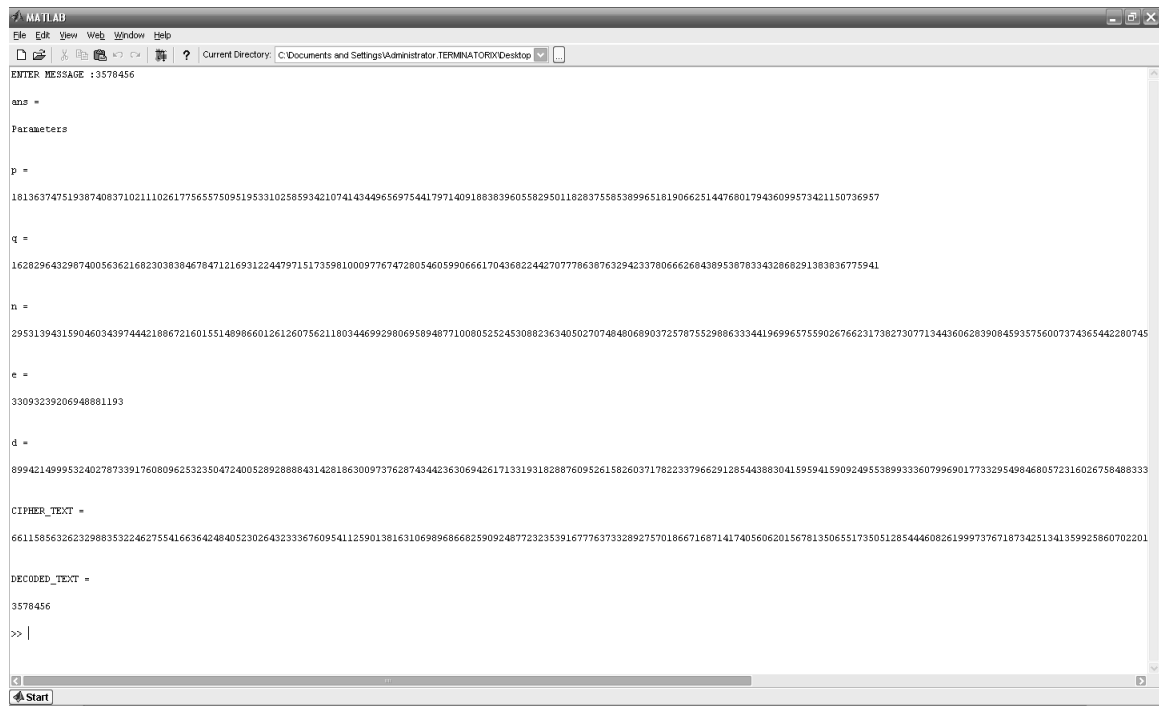
MESSAGE STRING: 3578456

Sample Output:

ENCRYPTED TEXT: n[°∞DO?i

DECRYPTED TEXT: 3578456

5.2 RSA Implementation in MATLAB

A screenshot of the MATLAB command window. The window title is 'MATLAB'. The menu bar includes 'File', 'Edit', 'View', 'Web', 'Window', and 'Help'. The current directory is 'C:\Documents and Settings\Administrator\TERMINATOR\Desktop'. The command prompt shows the following sequence: 'ENTER MESSAGE : 3578456', 'ans =', 'Parameters', 'p =', a long prime number, 'q =', another long prime number, 'n =', the product of p and q, 'e =', a small integer, 'd =', the modular inverse of e modulo n-1, 'CIPHER_TEXT =', the encrypted message, 'DECODED_TEXT =', and the decrypted message '3578456'. The prompt '>> |' is at the bottom.

```
MATLAB
File Edit View Web Window Help
Current Directory: C:\Documents and Settings\Administrator\TERMINATOR\Desktop

ENTER MESSAGE : 3578456

ans =

Parameters

p =

18136374751938740837102111026177565575095195331025859342107414344965697544179714091883839605582950118283755853899651819066251447680179436099573421150736957

q =

1628296432987400563621682303836467847121693122447971517359810009776747280546059906617043682244270777863876329423378066626843895387833432868291383836775941

n =

29531394315904603439744421886721601551489866012612607562118034469929806958948771008052524530882363405027074848068903725787552988633344196996575590267662317382730771344360628390845935756007374365442280745

e =

33093239206948881193

d =

89942149955324027873391760809625323504724005289288884314281863009737628743442363069426171331931828876095261582603717822337966291285443883041595941590924955389933360799690177332954984680572316026758488333

CIPHER_TEXT =

66115856326232988353224627554166364248405230264323336760954112590138163106989686682590924877232353916777637332892757018667168714174056062015678135065517350512854446082619997376718734251341359925860702201

DECODED_TEXT =

3578456

>> |
```

Sample Input:

MESSAGE STRING: 3578456

Sample Output:

ENCRYPTED TEXT:

661158563262329883532246275541663642484052302643233367609541125901381631
069896866825909248772323539167776373328927570186671687141740560620156781
350655173505128544460826199973767187342513413599258607022016921428012442
701974470570649450979274900523901664537086266094485628438393044081266464
59691999953052623390

DECRYPTED TEXT: 3578456

REFERENCES:

- [1] W.Stallings; “Cryptography and Network Security” 2nd Edition, Prentice Hall,1999
- [2] Bruce Schneir: Applied Cryptography, 2nd edition, John Wiley & Sons, 1996
- [3] Piper,F “Encryption”. Security and Detection, Ecos 97. European Conference;
- [4] Abrams, M., and Podell, H. “Cryptography” Potentials, IEEE Page No 36-38. Issue:1,Volume: 20, Feb-Mar,2001
- [5] Eskiciogiu,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.
- [7] W.Diffie; M.E.Hell man, “ New Directions in Cryptography” IEEE Transactions Information Theory, Nov, pp 644-654
- [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” , Compcon Spring '91. Digest of Papers, 25 Feb-1 March 1991