1. Introduction

1.1 COMPUTER SECURITY

The protection afforded to an automated information system in order to attain the applicable objectives of preserving the integrity, availability, and confidentiality of information system resources (includes hardware, software, firmware, information/ data, and telecommunications)**[1].**

1.2 Cryptography

Cryptography is closely related to the disciplines of cryptology and cryptanalysis. Cryptography includes techniques such as microdots, merging words with images, and other ways to hide information in storage or transit. However, in today's computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as cleartext) into ciphertext (a process called encryption), then back again (known as decryption). Individuals who practice this field are known as cryptographers **[1].**

Objectives of modern cryptography

1) **Confidentiality** (the information cannot be understood by anyone for whom it was unintended)

2) **Integrity** (the information cannot be altered in storage or transit between sender and intended receiver without the alteration being detected)

3) **Non-repudiation** (the creator/sender of the information cannot deny at a later stage his or her intentions in the creation or transmission of the information)

4) **Authentication** (the sender and receiver can confirm each other's identity and the origin/destination of the information)

Procedures and protocols that meet some or all of the above criteria are known as cryptosystems. Cryptosystems are often thought to refer

only to mathematical procedures and computer programs; however, they also include the regulation of human behavior, such as choosing hard-to-guess passwords, logging off unused systems, and not discussing sensitive procedures with outsiders **[3]**.

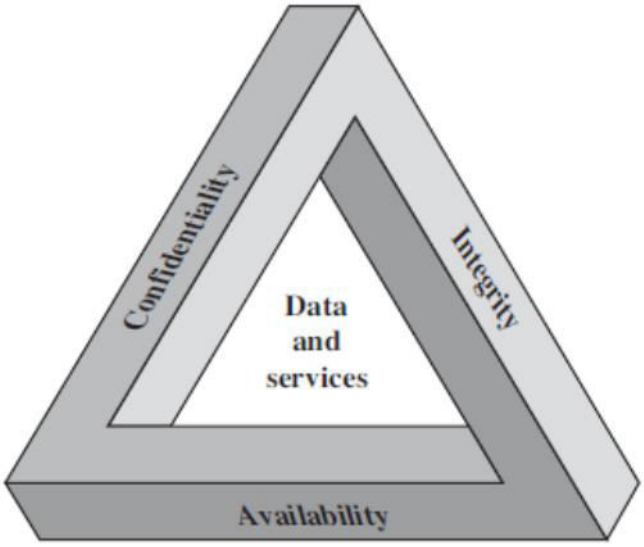

Figure (1.1) the security requirements

1.3 Security Mechanisms

security mechanisms defined in X.800. The mechanisms are divided into those that are implemented in a specific protocol layer, such as TCP or an application-layer protocol, and those that are not specific to any particular protocol layer or security service **[6]**.

1.3.1 specific security mechanisms

may be incorporated into the appropriate protocol layer in order to provide some of the OSI security services**[4]**.

A – Encipherment

The use of mathematical algorithms to transform data into a form that is not readily intelligible . the transformation and subsquent recovery of the data depend on an algorithm and zero or more encryption keys.

B – Digital signature

Data appended to,or a cryptographic transformation of,a data use that allows a recipient of the data unit to prove the source and integrity of the data unit and protect aginst forgery (e.g,by the recipient).

C – Access control

A variety of mechanisms that enforce access rights to resources .

D – Data integrity

A variety of mechanisms used to assure the integrity of a data unit or stream of data unit .

E – A uthentication Exchange

A mechanism intended to ensure the identity of an entity by means of information exchange .

F –Traffic padding

The insertion of bits into gaps in a data stream to frustrate traffic analysis attempts .

G –Routing control

Enables selection of particular physically secure routes for certain data and allows routing changes,especially when a breach of security is suspected .

H – Notarization

The use of a trusted thrid parity to assure certain properties of a data exchanges .

1.3.2 pervasive security mechanisms

Mechanisms that are not specific to any particular OSI security service

or protocol layer **[4]**.

A – Trusted Functionality

That which is perceived to be correct with respect to some criteria (e.g,as established by a security policy).

B – securiy Label

The marking bound to a resource(which may be a data unit)that names or designates the security attributes of that resource .

C – Event Detection

Detection of security relevant events.

D – security Audit Trail

Data collected and potentially used to facilitate a security audit, which is an independent review and examination of system records and activites .

E –Security Recovery

Deals with requests form mechanisms,such as event handling and management functions,and takes recovery actions.

2.1 Introduction

There are two main types of cryptography in use today – **symmetric** or **secret key** cryptography and **asymmetric** or **public key** cryptography. Symmetric key cryptography is the oldest type whereas asymmetric cryptography is only being used publicly since the late 1970's1. Asymmetric cryptography was a major milestone in the search for a perfect encryption scheme **[1]**.

Secret key cryptography goes back to at least Egyptian times and is of concern here. It involves the use of only one key which is used for both encryption and decryption (hence the use of the term symmetric). Figure 2.1 depicts this idea. It is necessary for security purposes that the secret key never be revealed **[5]**.

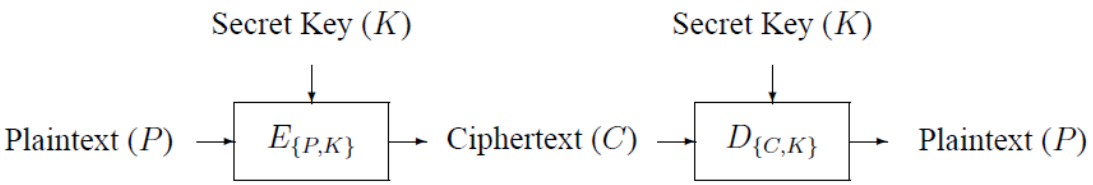

Figure 2.1: Secret key encryption.

2.2 Brief history of DES

Up until recently, the main standard for encrypting data was a symmetric algorithm known as the Data Encryption Standard (DES). However, this has now been replaced by a new standard known as the Advanced Encryption Standard (AES). DES is a 64 bit block cipher which means that it encrypts data 64 bits at a time. This is contrasted to a stream cipher in which only one bit at a time (or sometimes small groups of bits such as a byte) is encrypted.

DES was the result of a research project set up by International Business Machines (IBM) corporation in the late 1960's which resulted in a cipher known as LUCIFER. In the early 1970's it was decided to commercialise LUCIFER and a number of significant changes were introduced. IBM was not the only one involved in these changes as they sought technical advice from the National Security Agency (NSA) (other outside consultants were involved but it is likely that the NSA were the major contributors from a technical point of view). The altered version of LUCIFER was put forward as a proposal for the new national encryption standard requested by the National Bureau of Standards (NBS)3. It was finally adopted in 1977 as the Data Encryption Standard - DES (FIPS PUB 46) **[6]**.

There are many types of symmetric algorithms, each with varying levels of complexity. Such ciphers include: IDEA, RC4, RC5, RC6 and the new Advanced Encryption Standard (AES). AES is an important algorithm and was originally meant to replace DES (and its more secure variant triple DES) as the standard algorithm for *non-classified* material. However as of 2003, AES with key sizes of 192 and 256 bits has been found to be secure enough to protect information up to *top secret*.

2.3 Work of DES algorithm

DES (and most of the other major symmetric ciphers) is based on a cipher known as the **Feistel block cipher**. This was a block cipher developed by the IBM cryptography researcher Horst Feistel in the early 70's. It consists of a number of rounds where each round contains bit-shuffling, non-linear substitutions (S-boxes) and exclusive OR operations. Most symmetric encryption schemes today are based on this structure (known as a **feistel network**)**[1].**

As with most encryption schemes, DES expects two inputs - the plaintext to be encrypted and the secret key. The manner in which the plaintext is accepted, and the key arrangement used for encryption and decryption, both determine the type of cipher it is. DES is therefore a symmetric, 64 bit **block cipher** as it uses the same key for both encryption and decryption and only operates on 64 bit blocks of data at a time5 (be they plaintext or ciphertext). The key size used is 56 bits, however a 64 bit (or eight-byte) key is actually input. The least significant bit of each byte is either used for parity (odd for DES) or set arbitrarily and does not increase the security in any way. All blocks are numbered from left to right which makes the eight bit of each byte the parity bit **[8].**

Once a plain-text message is received to be encrypted, it is arranged into 64 bit blocks required for input. If the number of bits in the message is not evenly divisible by 64, then the last block will be padded. Multiple permutations and substitutions are incorporated throughout in order to increase the difficulty of performing a cryptanalysis on the cipher. However, it is generally accepted that the initial

and final permutations offer little or no contribution to the security of DES and in fact some software implementations omit them (although strictly speaking these are not DES as they do not adhere to the standard) **[8].**

2.3.1 Structure of DES

Figure 2.1 shows the sequence of events that occur during an encryption operation. DES performs an initial permutation on the entire 64 bit block of data. It is then split into 2, 32 bit sub-blocks, Li and Ri which are then passed into what is known as a **round** (see figure 2.2), of which there are 16 (the subscript i in Li and Ri indicates the current round). Each of the rounds are identical and the effects of increasing their number is twofold - the algorithms security is increased and its temporal efficiency decreased. Clearly these are two conflicting outcomes and a compromise must be made. For DES the number chosen was 16, probably to guarantee the elimination of any correlation between the cipher text and either the plaintext or key6. At the end of the 16th round, the 32 bit Li and Ri output quantities are swapped to create what is known as the **pre-output**. This [R16, L16] concatenation is permuted using a function which

is the exact inverse of the initial permutation. The output of this final permutation is the 64 bit ciphertext**[8].**

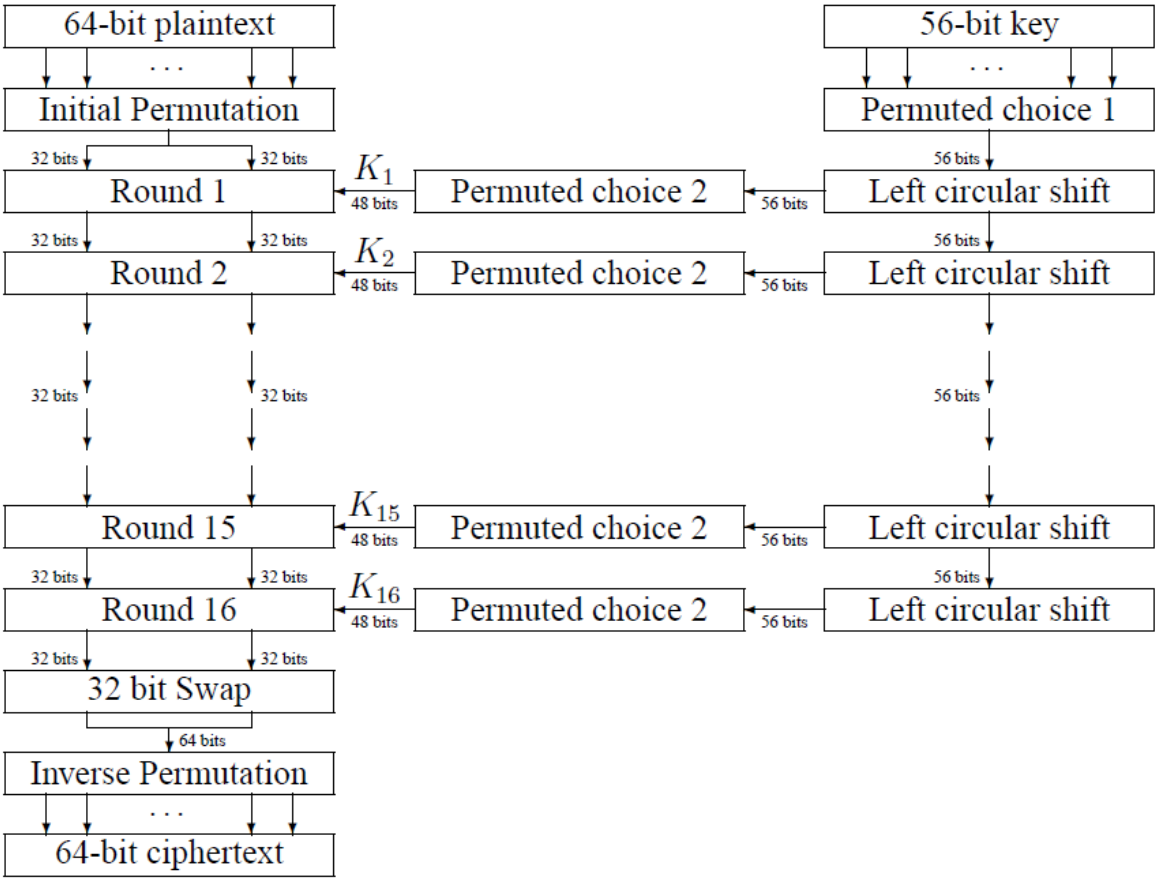

Figure (2.2) flow diagram of DES algorithm for encrypting data

(a) Initial permutation (IP)

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(b)Inverse Initial **permutation (IP)**

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

(c) Expansion Permutation (E)

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

(d) Permutation Function (P)

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

As figure 2.2 shows, the inputs to each round consist of the Li,Ri pair and a 48 bit **subkey** which is a shifted and contracted version of the original 56 bit key. The use of the key can be seen in the right hand portion of figure 2.2:

• Initially the key is passed through a permutation function (**PC**1 - defined in table 2.2)

• For each of the 16 iterations, a subkey (**K**i) is produced by a combination of a left circular shift and a permutation (**PC**2 - defined in table 2.2) which is the same for each iteration. However, the resulting subkey is different for each iteration because of repeated shifts**[4].**

(a) Input Key

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

(b) Permuted Choice One (PC-1)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

(d) Schedule of Left Shifts

Round Number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits Rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Table 2.2: DES key schedule

2 . 4 The proposed algorithm

2.4.1 the proposed algorithm for encryption process.

Algorithm

input: plaintext and key generation (48-bits) different key for each round**.**
output: cipher text.
step1: convert plaintext to blocks(64-bit).
step2: work initial permutation for plaintext block(64bit).
Step3: each block divide into left and right block.
Step4: for I = 0 to 16 do

1- part of right block enter the function with the key (F(key,Ri)).
the process of function
a- expansion permutation of 32-bit (right block)to 48-bit.
b- XOR between (the right block and the key).
c- S_BOX(8s_box) input 6-bit and the output 4-bit and combine the 4-bit for each s_box to be a 32-bit.
d- Permutation function (p_ box)of 32-bit.

2- XOR between (the left block and the result of function).
3- Swap process (between the left block and the right block).

Next
Step5: combine the two blocks and work the inverse initial permutation it to make cipher text.

2.4.2 the proposed algorithm for decrption process.

Algorithm

input: cipher text and key generation (48-bits) different key for each round**.**
output: plaintext
step1: convert plaintext to blocks(64-bit).
step2: work initial permutation for plaintext block(64bit).
Step3: each block divide into left and right block.
Step4: for I = 16 to 0 do
4- part of right block enter the function with the key (F(key,Ri)).
the process of function
e- expansion permutation of 32bit (right block) to 48-bit.
f- XOR between (the right block and the key).
g- S_BOX(8s_box) input 6bit and the output 4bit and combine of each s_box to be a 32-bit.
h- Permutation function (p_ box) of 32-bit.
5- XOR between (the left block and the result of function).
6- Swap process (between the left block and the right block).
Next
Step5: combine the two blocks and work the inverse initial permutation it to make plaintext.

2.4.3 the proposed algorithm for encryption process of RC6:

Input: plaintext and key generation (S [0, …..,2r+3])
Output: cipher text.
Step1: convert plaintext to block (128-bit).
Step2: each block divide into (A,B,C,D).
Step3: B=B+S [0].
D=D+S [1].

Step4: for I = 0 to r do
{
- T= ((B x (2B + 1)) <<< LG (w)
- u = ((D x (2D + 1)) <<< LG(w)
- A = ((A ⊕ t) <<< u) + S[2i]
- C = ((C ⊕ u) <<< t) + S[2i + 1]
- Swap process (A, B, C, D) = (B, C, D, A)

}
Step5: A = A + S [2r + 2].
C = C + S [2r + 3] .
Step6: combine the 4-block(A,B,C,D) to make the cipher text .

3.1 Introduction

Rapid growth of internet applications fueled the need for securing information and computers. Encryption algorithms play vital role to secure information. This project provides new secure method of most common encryption algorithms namely: DES, RC6**[8].**

RC6 is a block cipher based on RC5 and designed by Rivest, Sidney, and Yin for RSA Security. Like RC5, RC6 is a parameterized algorithm where the block size, the key size, and the number of rounds are variable; again, the upper limit on the key size is 2040 bits.RC6 was designed to meet the requirements of the Advanced Encryption Standard (AES) competition. RC6 proper has a block size of 128 bits and supports key sizes of 128, 192 and 256 bits, but, like RC5. RC6 can be viewed as interweaving two parallel RC5 encryption processes. It uses an extra multiplication operation not present in RC5 in order to make the rotation dependent on every bit in a word **[6]**.

3.2 Implementation

1-main interface

The windows below show the implementation of our project:

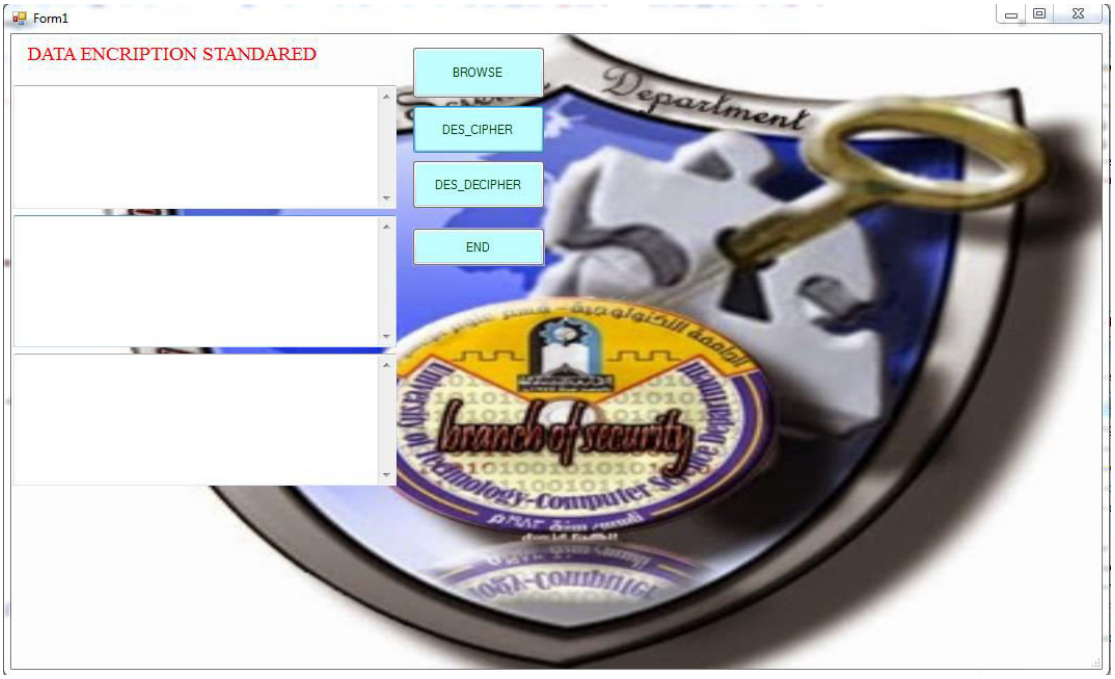

Figure (3.1) main interface

This the first widow of program that ask the user to brows the file that wants to ecrypted it.

2- Brose file

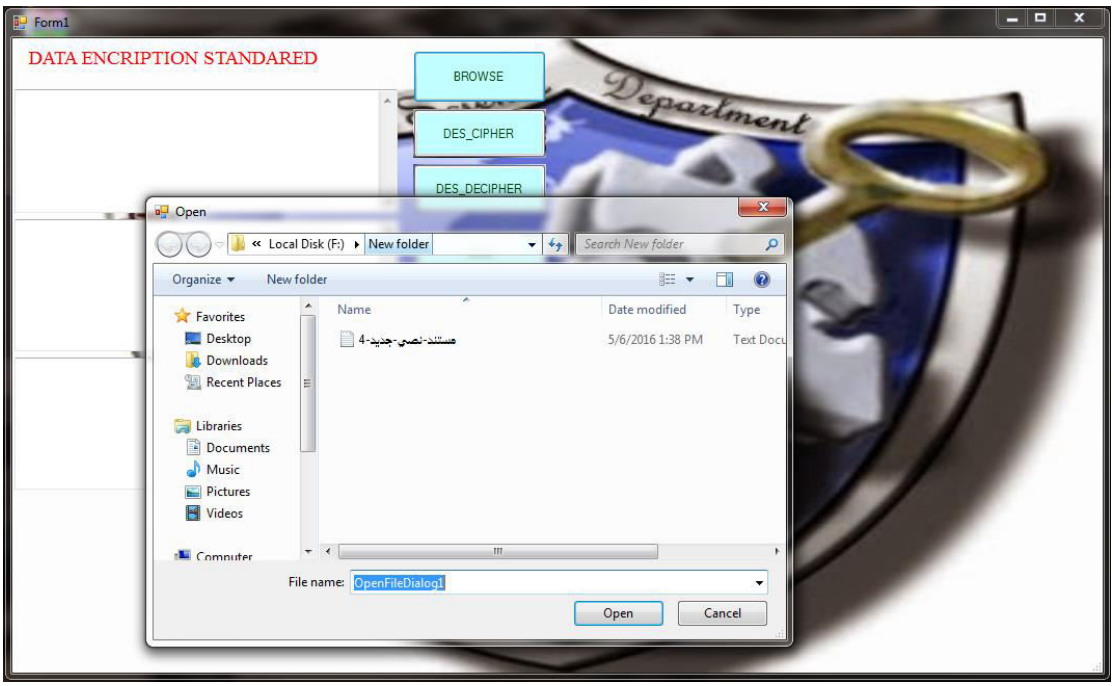

Figure (3.2) Select file for encryption

the user select a file and press open then next widow is appear when the user press on the DES_CIPHE button:

3-Enter 16 character

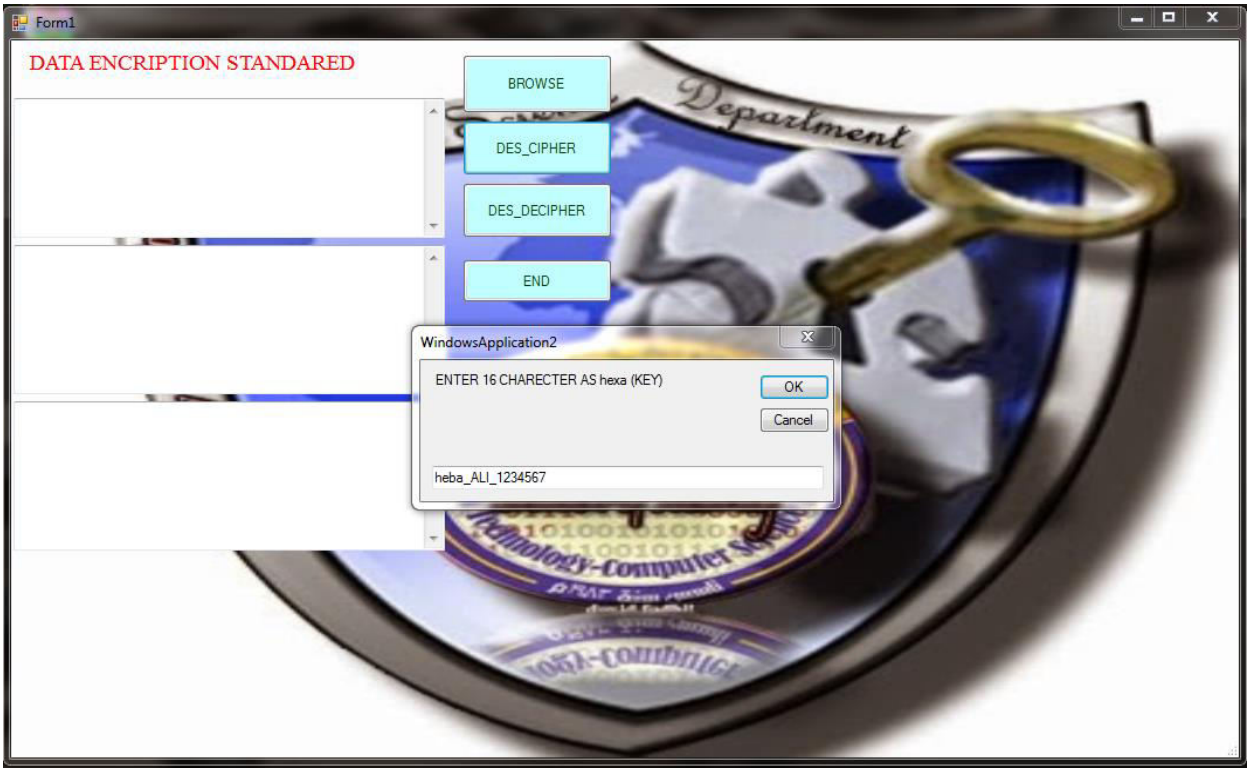

Figure (3.3) explain how the key is entered

The program ask the user to enter 16 characters as key and press ok.

4-Enter key of RC6

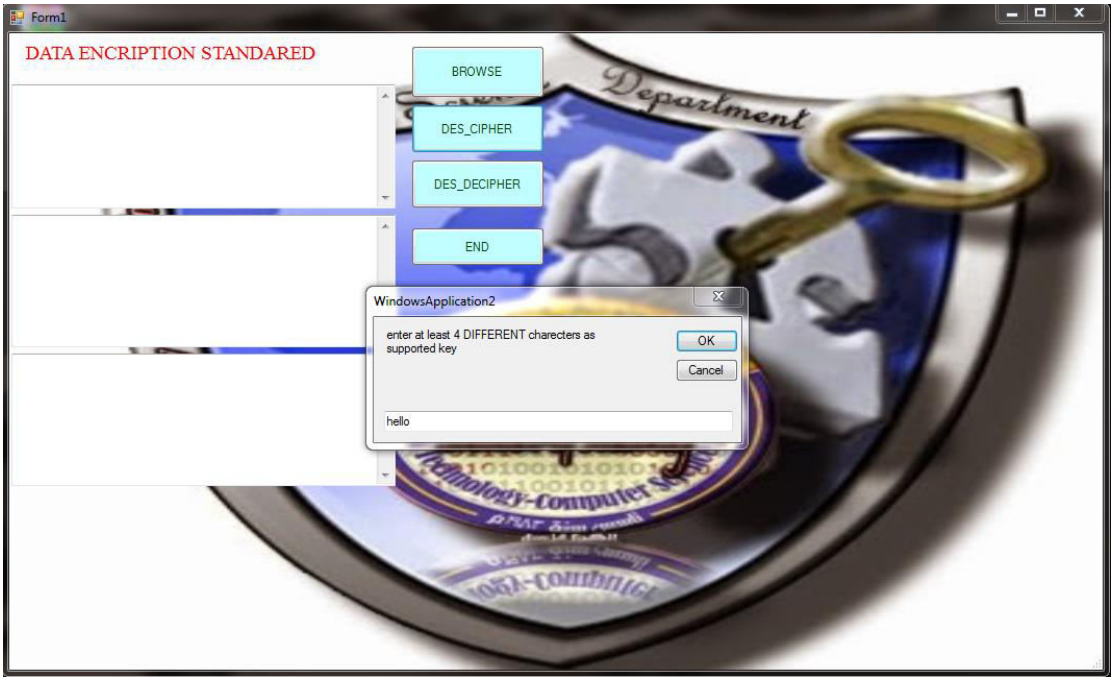

Figure (3.4) Rc6 key

In this widow the program ask the user to enter 4 different as supported key then press ok to encryption be execute.

5- Encryption process

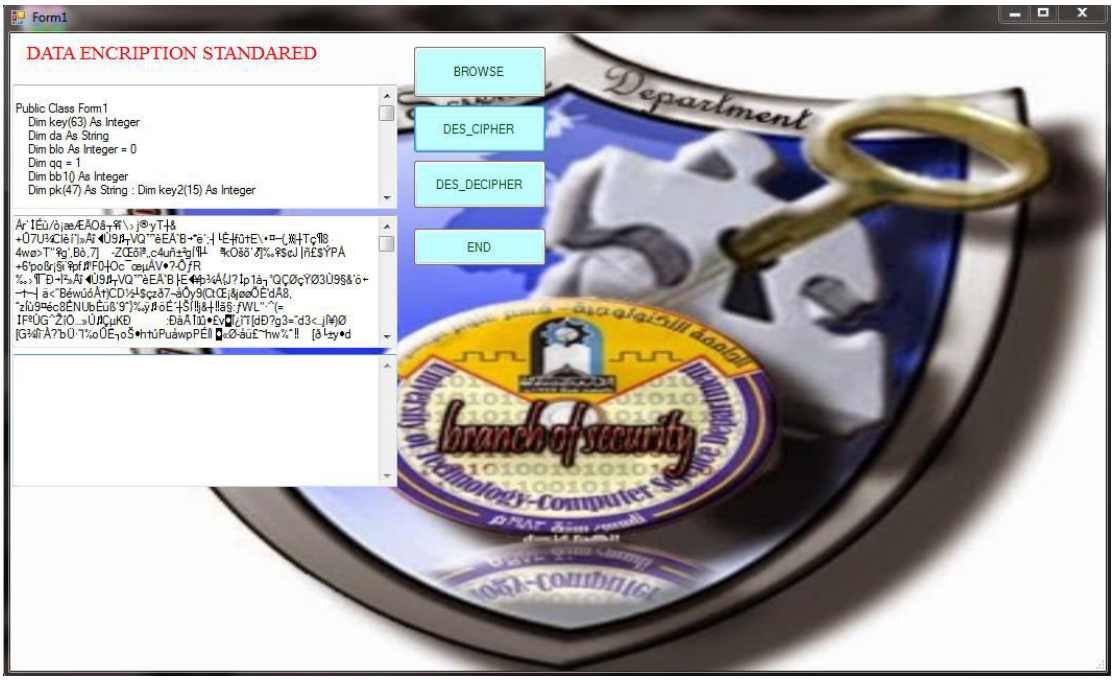

Figure (3.5) show the des _encryption

The below widow will appear when the user press on DES_DECIPHER button:

6- Decryption process

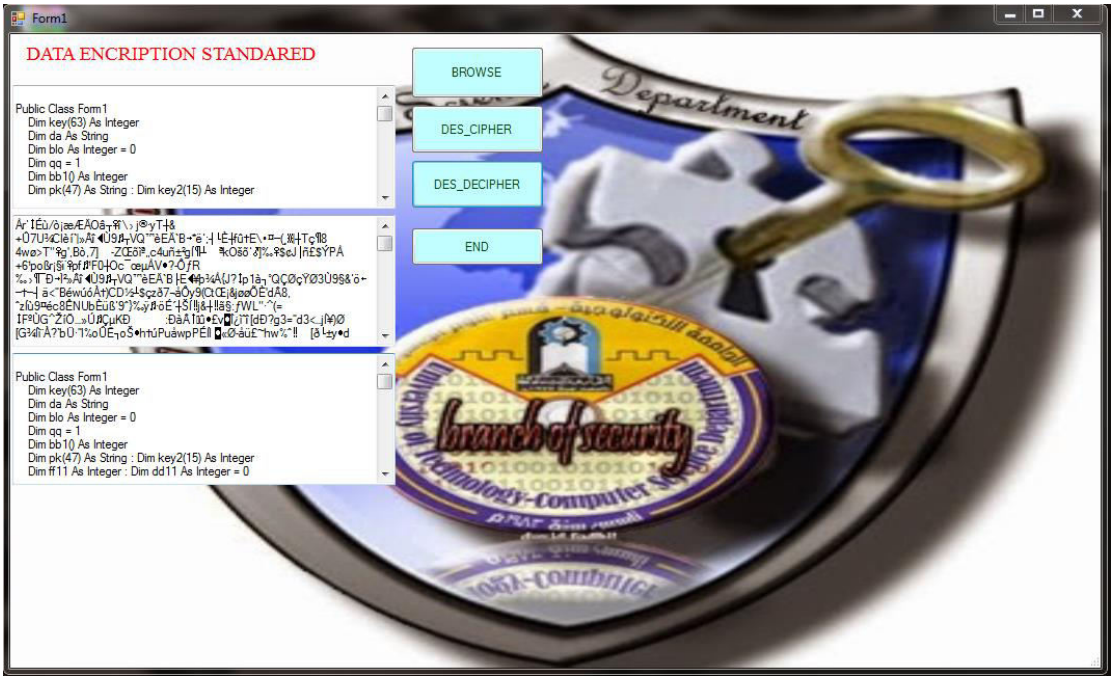

Figure (3.6) show des _decryption

To end the program the user press on END button .