Assignment # 1

Generell Systemteori Med Tonvikt på styr- och kontrollfunktioner
(SÄK1a/2I1501del1/2I4118)

Data Encryption Standard
A Symmetric Cryptographic Algorithm

Batch: 2005-2006

Group Members

Sadeeq Jan
Charu Gupta
Tariq Saeed

Table of Contents

List of Figures

INTRODUCTION TO CRYPTOGRAPHY

Cryptography *is the* art of achieving security by encoding messages to make them non-readable. Cryptography enables one to store sensitive information or transmit it across insecure networks (like the Internet) so that it cannot be read by anyone accept the intended recipient.

A *Cryptographic Algorithm,* or *Cipher*, is a mathematical function used in the encryption and decryption process. A cryptographic algorithm works in combination with a *key*—a word, number, or phrase—to encrypt the plaintext.
The same plaintext encrypts to different cipher text with different keys. The security of encrypted data is entirely dependent on two things: the strength of the cryptographic algorithm and the secrecy of the key.
A cryptographic algorithm, plus all possible keys and all the protocols that make it work comprise a ***cryptosystem*.** [1]

1.1 Types of Cryptography

1.1.1 Symmetric cryptography
1.1.2 Asymmetric cryptography

1.1.1 Symmetric/conventional/Private Key Cryptography

In conventional cryptography, also called *secret-key* or *symmetric-key* encryption, one key is used both for encryption and decryption. The Data Encryption Standard (DES) is an example of a conventional cryptosystem that is widely employed by the Federal Government.

1.1.2 Asymmetric Cryptography/Public Key Cryptography

In traditional **private/secret/single key** cryptography one key is used which is shared by both sender and receiver. If this key is disclosed communications are compromised. Public key cryptography is an asymmetric scheme that uses a *pair* of keys for encryption: a *public key*, which encrypts data, and a corresponding *private*, or *secret key* for decryption. You publish your public key to the world while keeping your private key secret. Anyone with a copy of your public key can then encrypt information that only you can read. Even people you have never met. It is computationally infeasible to deduce the private key from the public key. Anyone who has a public key can encrypt information but cannot decrypt it. Only the person who has the corresponding private key can decrypt the Information.

1.2 Types of Transformation

Substitution
Transposition

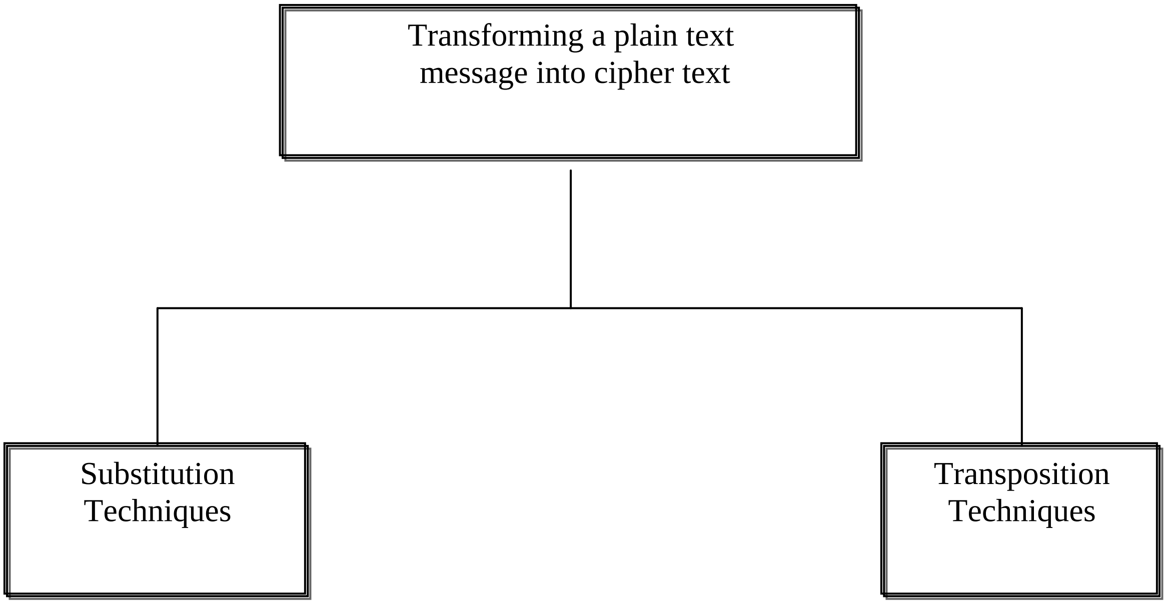


Figure 1.1 Transforming plaintext into ciphertext

1.2.1 Substitution

In Substitution, letters of plaintext are replaced by other letters or by numbers or symbols. For example “Hello” can be transformed to “KHOOR”, where every character is replaced by the next 3rd character.

1.2.2 Transposition

In Transposition, letters of plaintext are rearranged to form ciphertext. For example “NetworkSecurity” can be transformed to “ytisecurkwroetn”. In this example there is no replacement but only rearrangement of characters.

1.3 Symmetric Cipher Model

Symmetric encryption was widely used before the development of Asymmetric encryption.

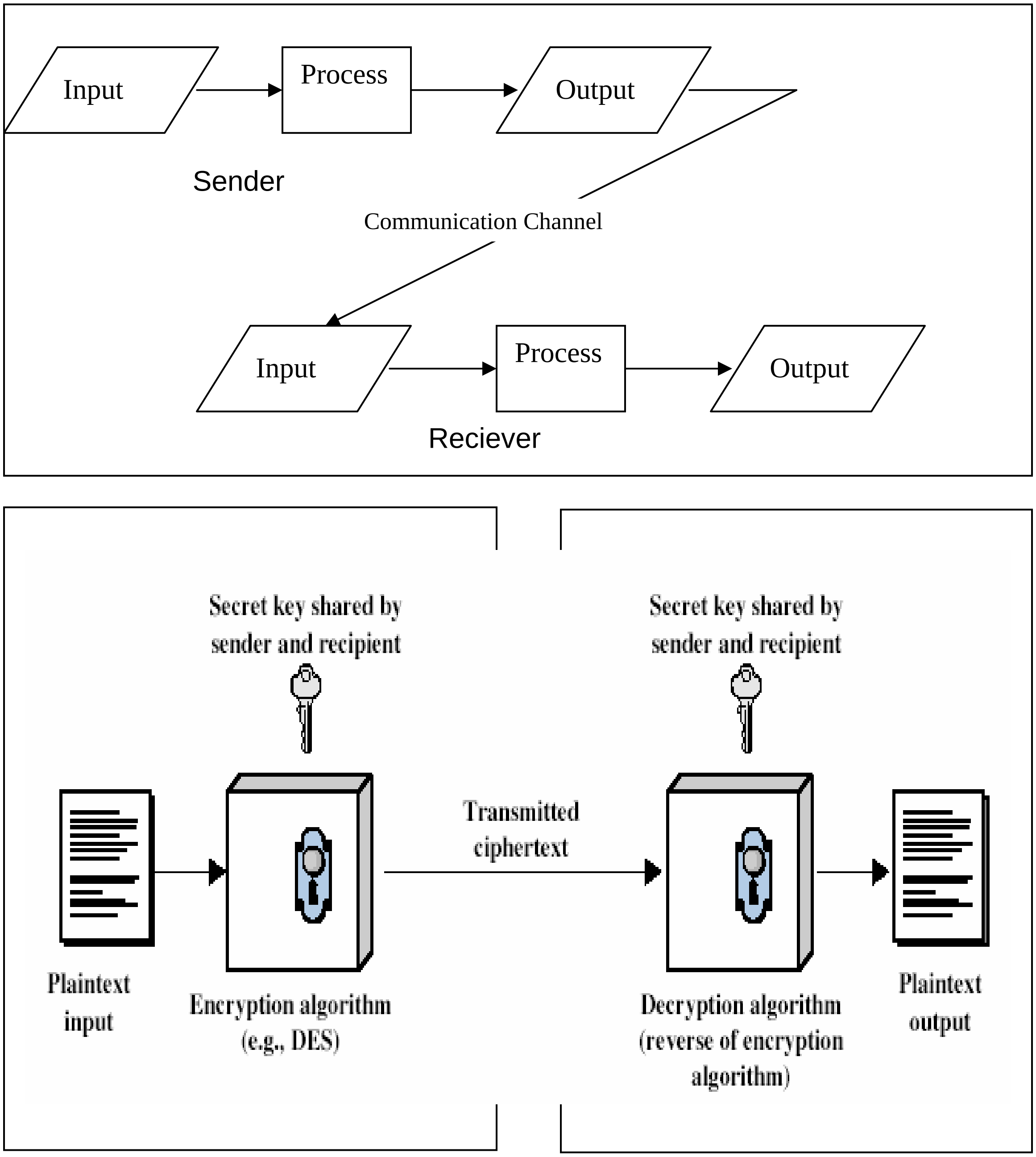
1.3.1 Ingredients of Symmetric Cipher Model

There are five basic ingredients of symmetric cipher model.

- **Plaintext** is the original intelligible message or date which is given to the algorithm as input.
- **Ciphertext** is the scrambled message produced as output.
- **key** is the number/text used in cipher and it is known only to sender/receiver
- **Encryption Algorithm** is the process of converting plaintext to ciphertext.
- **Decryption Algorithm** is the process of recovering ciphertext from plaintext.

1.4 Symmetric Cipher as a System

A process which takes some input and performs transformation on the input to produce some output can be regarded as a system. Therefore we can consider a Symmetric Cipher Model as a System because it takes plaintext as input, process this input and produces Ciphertext as output as shown in figure 1.2.


Input
Process
Output
Sender
Communication Channel
Input
Process
Output
Reciever
Secret key shared by sender and recipient
Secret key shared by sender and recipient
Transmitted ciphertext
Plaintext input
Encryption algorithm (e.g., DES)
Decryption algorithm (reverse of encryption algorithm)
Plaintext output

Sender System (Subsystem I) **Receiver System (Subsystem II)**
Input: Plaintext *Input: Ciphertext*
Output: Ciphertext *Output: Plaintext*

Figure 1.2 Symmetric Cipher Model [2]

The entire model can be considered as a combination of two subsystems. i.e. a system at the sender which is known as the Encryption System and another system at the receiver which is known as Decryption System.

1.5 Symmetric Cipher Algorithms

The following are a few algorithms used for symmetric ciphers

- Ceaser Cipher
- Mono-Alphabatic Cipher
- Playfair Cipher
- Hill Cipher
- Polyalphabatic Cipher
- One Time Pad

All of these algorithms were broken by hackers except the One Time Pad but One Time Pad has also some limitations due to which it was not used widely. Therefore a new algorithm was needed and IBM developed DES (Data Encryption Standard). The next section describes DES in detail.

DATA ENCRYPTION STANDARD

2.1 DES History

In 1971 the IBM team led by Horst Feistel developed an Algorithm called LUCIFER. This algorithm was sold to Lloyd’s of London for use in cash dispensing system. LUCIFER is a feistel block cipher that operates on blocks of 64 bits using a key size of 128 bits. Because of the promising results produced by the LUCIFER project, IBM embarked on an effort to develop a marketable commercial encryption product that ideally could be implemented on a single chip. So they refined LUCIFER with a key size of 56 bits which was implemented on a single chip and this algorithm was adopted in 1977 as the Data Encryption Standard by National Bureau of Standards (NBS).

Before its adoption as a standard, the proposed DES was subjected to intense criticism. The first area of concern was that the key length in IBM’s original LUCIFER algorithm was 128 bits, but that of the proposed system was only 56 bits, an enormous reduction in key size of 72 bits. Critics feared that that this key length was too short to withstand brute force attack.

The second area of concern was that the design criteria for the internal structure of DES, the S-boxes, were classified. Thus, users could not be sure that the internal structure of DES was free of any hidden weak points that would enable NSA to decipher messages without benefit of the key. Subsequent events, particularly the recent work on differential cryptanalysis, seem to indicate that DES has a very strong internal structure.
According to IBM participants, the only changes that were made to the proposal were changes to the S-boxes, suggested by NSA, that removed vulnerabilities identified in the course of the evaluation process.

Whatever the merits of the case, DES has flourished and is widely used, especially in financial applications. In 1994, NIST reaffirmed DES for federal use for another five years; NIST recommended the use of DES for applications other than the protection of classified information. In 1999, NIST issued a new version of its standard (FIPS PUB 46-3) that indicated that DES should only be used for legacy systems and that triple DES (which in essence involves repeating the DES algorithm three times on the plaintext using two or three different keys to produce the ciphertext) be used.

2.2 DES Encryption

The overall scheme for DES encryption is illustrated in Figure 2.1. As with any encryption scheme there are two inputs to the encryption function; the plaintext to be encrypted and the key. In this case, the plaintext must be 64 bits in length and the key is 56 bits in length.

The processing of the plaintext proceeds in three phases. First the 64 bit plaintext passes through an initial permutation that rearranges the bits to produce the permuted input. This is followed by a phase consisting of 16 rounds of the same function, which involves both permutation and substitution functions. The output of the last (16th) round consists of 64 bits that are a function of the input plaintext and the key. The left and right halves of the output are swapped to produce the pre-output. Finally, the pre-output is passed through a permutation that is the inverse of the initial permutation function, to produce the 64 bit ciphertext.

The right hand portion of figure shows the way in which the 56 bit key is used. Initially, the key is passed through a permutation function, then for each of the 16 rounds, a subkey(K_i) is produced by the combination of the left circular shift and a permutation. The permutation function is the same for each round, but a different subkey is produced because of the repeated iterations of the key bits.

A key consists of 64 binary digits ("O"s or "1"s) of which 56 bits are randomly generated and used directly by the algorithm. The other 8 bits, which are not used by the algorithm, are used for error detection. The unique key chosen for use in a particular application makes the results of encrypting data using the algorithm unique. Selection of a different key causes the cipher that is produced for any given set of inputs to be different. The cryptographic security of the data depends on the security provided for the key used to encipher and decipher the data.Data can be recovered from cipher only by using exactly the same key used to encipher it. Unauthorized recipients of the cipher who know the algorithm but do not have the correct key cannot derive the original data algorithmically. However, anyone who does have the key and the algorithm can easily decipher the cipher and obtain the original data. A standard algorithm based on a secure key thus provides a basis for exchanging encrypted computer data by issuing the key used to encipher it to those authorized to have the data.

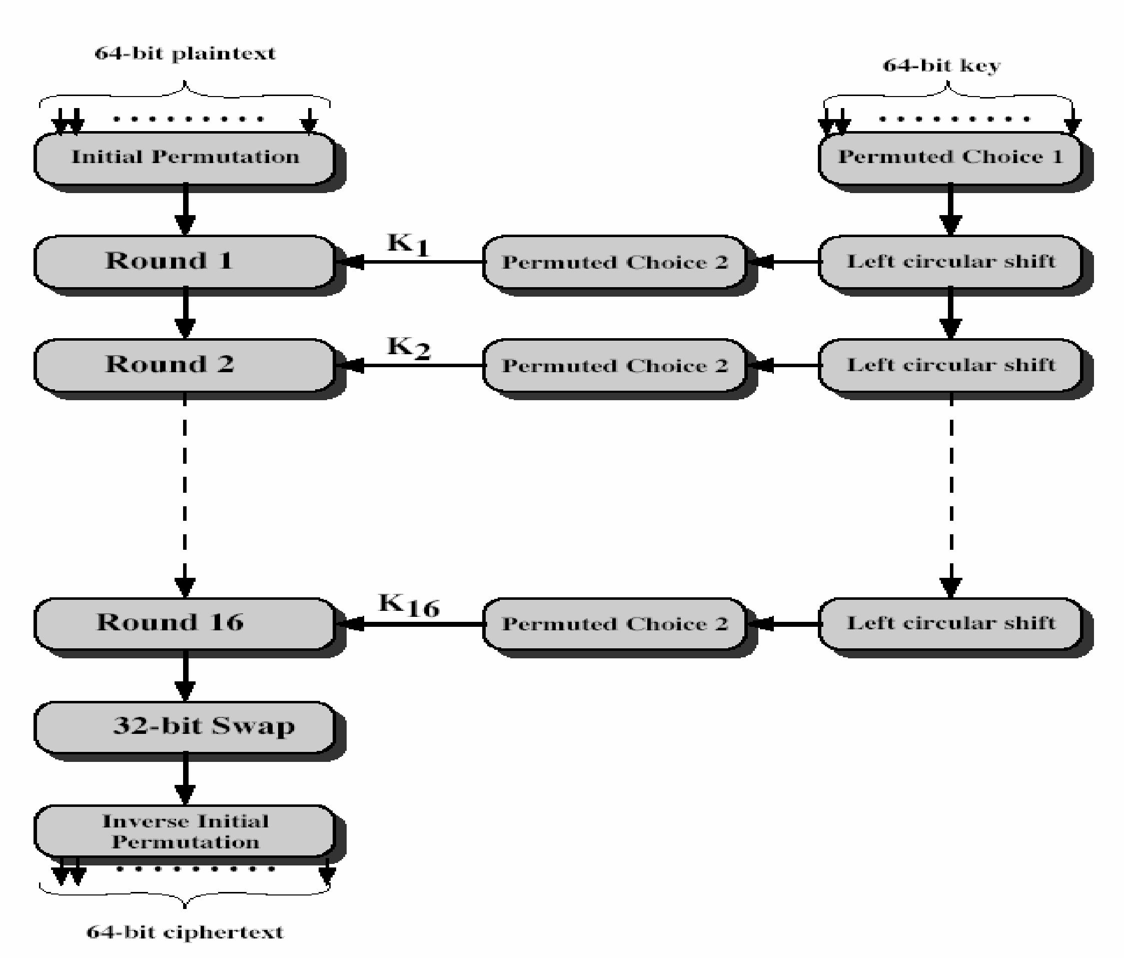


Figure 2.1 DES Algorithm [2]

2.2.1 Initial permutation

The first step of DES Algorithm is the initial permutation in which the 64 bits of the input block are permuted according to the following table.

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

The 58th bit of the input becomes the first bit of the permuted input, 50th bit becomes the second bit, the bit 42 becomes the 3rd bit and so on. The 7th bit of the input becomes the last bit of the permuted input.

The permuted input block is then passed through 16 rounds. The details of a single round are explained in section 2.3.2. The output of that computation, called the preoutput, is then subjected to the following permutation which is the inverse of the initial permutation:

IP^{-1}

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

That is, the output of the algorithm has bit 40 of the preoutput block as its first bit, bit 8 as its second bit, and so on, until bit 25 of the preoutput block is the last bit of the output.

2.2.2 Details of a Single Round

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

$\boldsymbol{Li = Ri-1}$
$\boldsymbol{Ri = Li-1 \text{ xor } F(Ri-1, Ki)}$

The round key Ki 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 Ki. This 48 bit result passes through a substitution function that produces a 32 bit output, which is permuted as defined in table 2.2.

2.2.3 Key Generation

A 64-bit key used as input to the algorithm. The bits of the key are numbered from 1 through 64, every 8th bit is ignored. This is first subjected to a permutation governed by a table labeled Permuted Choice 1. The resulting 56 bit key is then treated as two 28 bit quantities, labeled as C0 and D0. at each round Ci-1 and Di-1 are separately subjected to a circular left shift, or rotation, of 1 or 2 bits, as governed by Table. These shifted values serve as input to the next round. They also serve as input to permuted choice 2, which produce a 48 bit output that serves as input to the function F (Ri-1,Kj).

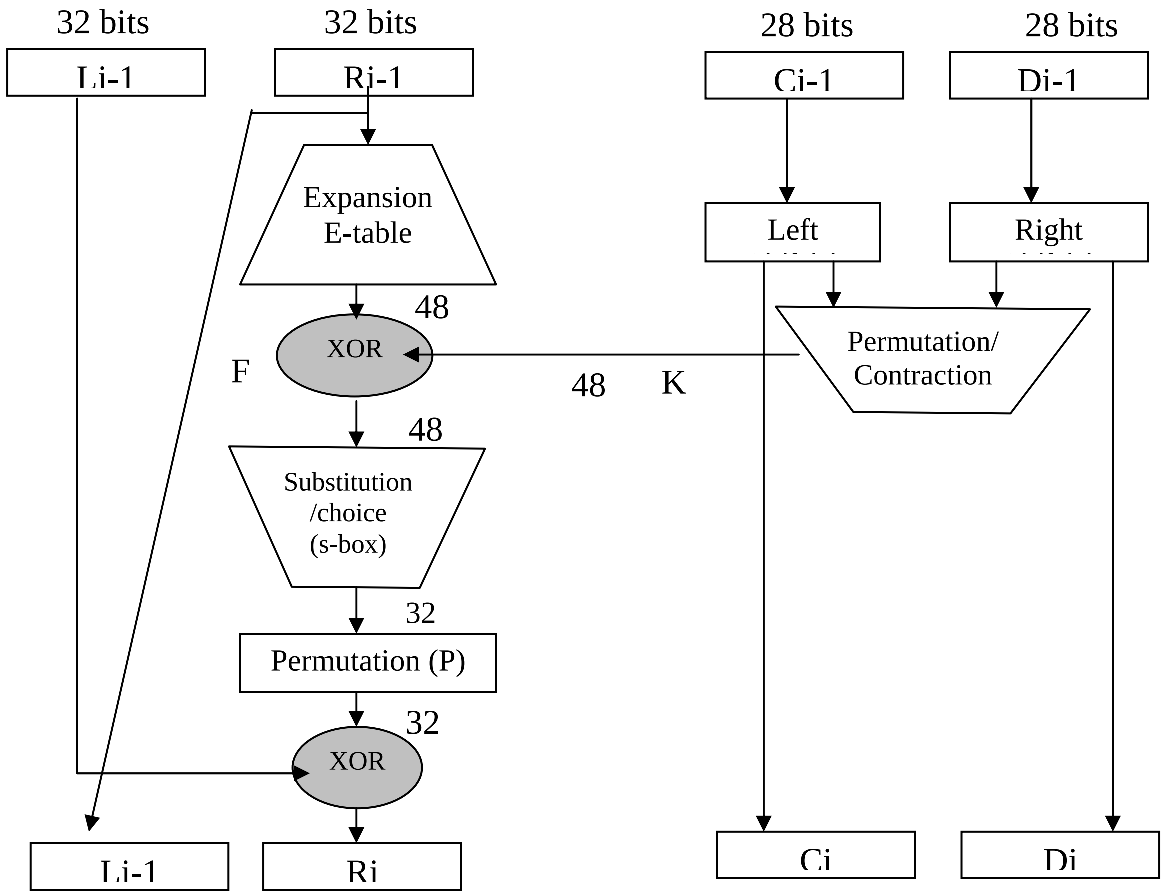


Figure 2.2 Single Round of DES

Table 2.1 **Expansion Table *E* BIT-SELECTION TABLE**

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

Table 2.2 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

2.2.4 S-Boxes

There are 8 selection boxes S1, S2, …..S8. each S box takes 6-bit block as input and produces 4-bit block as output.

S_1

Column Number

Row No.	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

The first and last bits of the input box Si for a 2-bit binary number which is used for the selection of Row defined in the above table. The middle four bits are used to select one of the sixteen columns. The decimal value at the intersection of row and column is then converted to 4-bit binary number to produce the output.

For example if the input is 110001, then the first and last bits of this number are 11 which selects row number 3. the middle four bits are 1000 which is equivalent of 8 in decimal, that means the column to be selected is 8^{th}. In row 3 and column 8 the number is 5 which in binary is equivalent 0101 which is the intended output.

All of the four S-Boxes are shown below:

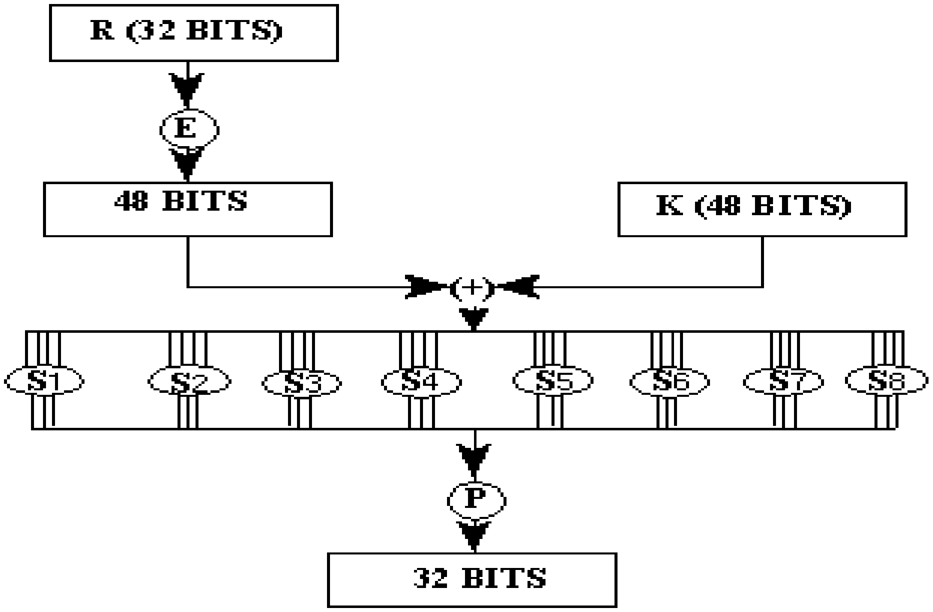


Figure 2.3 S-Boxes [2]

$\mathbf{S}_1$

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

$\mathbf{S}_2$

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

$\mathbf{S}_3$

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

$\mathbf{S}_4$

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

$\mathbf{S}_5$

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

S$_6$

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

S$_7$

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

S$_8$

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

2.3 DES Decryption

The decryption must unwind steps of data computation. As with any feistel cipher, decryption uses the same algorithm as encryption, except that the application of subkeys is reversed. i.e. using subkeys in reverse order (SK16 … SK1). In this case IP undoes final FP step of encryption. 1st round with SK16 undoes 16th encrypt round, 16th round with SK1 undoes 1st encrypt round, then final FP undoes initial encryption IP thus recovering original data value.

2.4 Vulnerabilities in DES System

When DES was adopted as a federal standard, people started to explore vulnerabilities in it, their concerns, by and large, fell into two areas, key size and the nature of the algorithm.

2.4.1 Concerns for Key Size

When DES was first introduced, it used a Key size of 56 bits which had 2^{56} possible keys. Due to which people thought that it would be impractical to break it using brute force attack but Electronic Frontier Foundation (EEF) broke DES encryption in July 1998 using a special purpose “DES cracker” machine that was built for less than $250,000. The attack took less than three days. The EFF has published a detailed description of the machine, enabling others to build their own cracker. EFF has published all of the research in a book by O'Reilly and Associates, entitled "Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design."

However there is more to a key search attack than simply running through all possible keys. Unless known plaintext is provided, the analyst must be able to recognize plaintext as plaintext. If the message is just plain text in English, then the result pops out easily, although the task of recognizing English would have to be automated. If the text message has been compressed, the problem becomes even more difficult to automate. Thus, to supplement the brute force approach, some degree of knowledge abort the expected plaintext is needed, and some means of automatically distinguishing plaintext from garbage is also needed.

2.4.2 The Nature of the DES Algorithm

Another concern is the possibility that cryptanalysis is possible by exploiting the characteristics of the DES algorithm. The focus of concern has been on the eight substitution tables, or S-boxes, that are used in each iteration. Because the design criteria for these boxes, and indeed for the entire algorithm, were not made public, there is a suspicion that the boxes were constructed in such a way that cryptanalysis is possible for an opponent who knows the weaknesses in the S-boxes. This assertion is tantalizing, and over the years a number of regularities and unexpected behaviors of the S-boxes have been discovered. Despite this, no one has so far succeeded in discovering the supposed fatal weaknesses in the S-boxes.

2.4.3 Timing Attacks

Timing attacks is related to public key algorithms. However, the issue may also be relevant for symmetric ciphers. In essence, a timing attack is one in which information about the key or the plaintext is obtained by observing how long it takes a given implementation to perform decryptions on various ciphertexts. A timing attack exploits the fact that an encryption or decryption algorithm often takes slightly different amounts of time on different inputs. A report [5] on an approach that yields the Hamming weight (number of bits equal to one) of the secret key. This is a long way from knowing the actual key, but it is an intriguing first step. The authors conclude that DES appears to be fairly resistant to a successful timing attack; it so far appears unlikely that this technique will ever be successful against DES or more powerful symmetric ciphers such as triple DES and AES.

CONCLUSION

Data Encryption Standard is a very powerful algorithm. It is much faster than the public key systems like RSA. It is also easy to implement in both hardware and software. It was tested for about 25 years and no logic flaws were detected. Despite these advantages DES also has some limitations like the key transmission of private key over public channel, slower than AES, RC6 and its smaller key length.

However if the key size of DES is made larger which is done in Triple DES, then DES can be a very powerful and unbreakable algorithm.

REFERENCES & BIBLIOGRAPHY

[1] An Introduction to Cryptography
http://www.pgpi.org/doc/guide/6.5/en/intro/ (Date: 2005/09/08)

[2] Cryptography and Network Security by William Stallings

[3] Federal Information Processing Standards Publication 46-2, 1993 December 30 Announcing the Standard for DATA ENCRYPTION STANDARD (DES)

[4] Cracking DES: Secrets of Encryption Research, Wiretap Politics, and Chip Design by O'Reilly and Associates

[5] Hevia, A and Kiwi ,M. “Strength of Two Data Encryption Standard Implementations under Timing Attacks” ACM Transactions on Information and System Security, November 1999.