MSC-CS

CRYPTOGRAPHY & NETWORK SECURITY

UNIT - 2

Unit-2: Classical Encryption Techniques :

Symmetric Cipher Model – Substitution Techniques – Transposition Techniques – Rotor Machines – Steganography – Block Ciphers and the Data Encryption Standard: Traditional Block Cipher Structure – The Data Encryption Standard – A DES Example – The Strength of DES – Block Cipher Design Principles

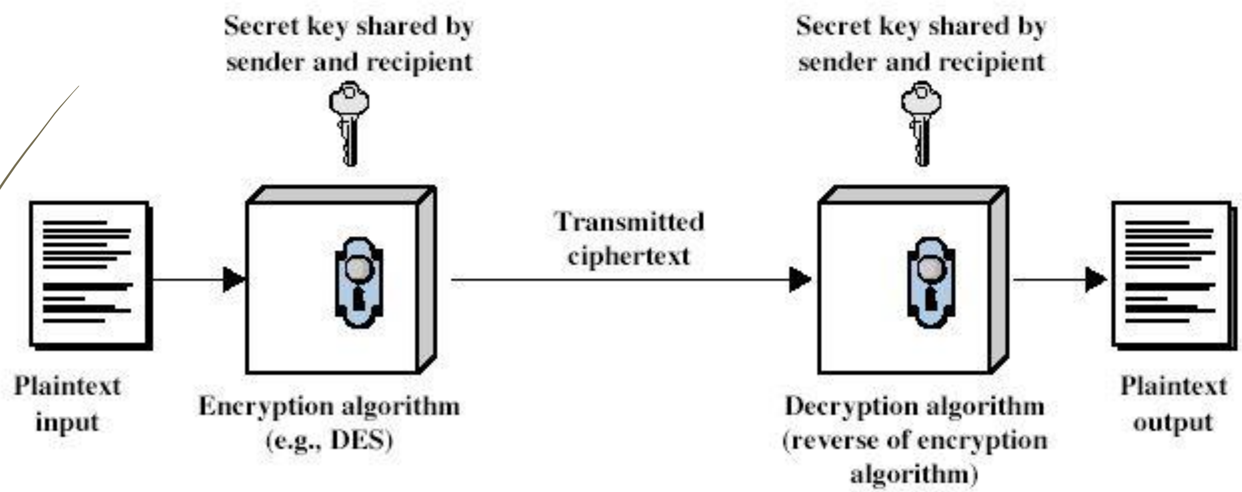
Symmetric Encryption

- or conventional / private-key / single-key
- sender and recipient share a common key
- all classical encryption algorithms are private-key
- was only type prior to invention of public-key in 1970's
- and by far most widely used

Some Basic Terminology

- **plaintext** - original message
- **ciphertext** - coded message
- **cipher** - algorithm for transforming plaintext to ciphertext
- **key** - info used in cipher known only to sender/receiver
- **encipher (encrypt)** - converting plaintext to ciphertext
- **decipher (decrypt)** - recovering ciphertext from plaintext
- **cryptography** - study of encryption principles/methods
- **cryptanalysis (codebreaking)** - study of principles/ methods of deciphering ciphertext *without* knowing key
- **cryptology** - field of both cryptography and cryptanalysis

Symmetric Cipher Model



Requirements

- two requirements for secure use of symmetric encryption:
  - a strong encryption algorithm
  - a secret key known only to sender / receiver
- mathematically have:

  $Y = E_K(X)$

  $X = D_K(Y)$
- assume encryption algorithm is known
- implies a secure channel to distribute key

Cryptography

- characterize cryptographic system by:
  - type of encryption operations used
    - substitution / transposition / product
  - number of keys used
    - single-key or private / two-key or public
  - way in which plaintext is processed
    - block / stream

Cryptanalysis

- objective to recover key not just message
- general approaches:
  - cryptanalytic attack
  - brute-force attack

Cryptanalytic Attacks

- **ciphertext only**
  - only know algorithm & ciphertext, is statistical, know or can identify plaintext
- **known plaintext**
  - know/suspect plaintext & ciphertext
- **chosen plaintext**
  - select plaintext and obtain ciphertext
- **chosen ciphertext**
  - select ciphertext and obtain plaintext
- **chosen text**
  - select plaintext or ciphertext to en/decrypt

More Definitions

- **unconditional security**
  - no matter how much computer power or time is available, the cipher cannot be broken since the ciphertext provides insufficient information to uniquely determine the corresponding plaintext
- **computational security**
  - given limited computing resources (eg time needed for calculations is greater than age of universe), the cipher cannot be broken

Brute Force Search

- always possible to simply try every key
- most basic attack, proportional to key size
- assume either know / recognise plaintext

Key Size (bits)	Number of Alternative Keys	Time required at 1 decryption/µs	Time required at 10^6 decryptions/µs
32	$2^{32} = 4.3 \times 10^9$	2^{31} µs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} µs = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} µs = 5.4×10^{24} years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} µs = 5.9×10^{36} years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} µs = 6.4×10^{12} years	6.4×10^6 years

Classical Substitution Ciphers

- where letters of plaintext are replaced by other letters or by numbers or symbols
- or if plaintext is viewed as a sequence of bits, then substitution involves replacing plaintext bit patterns with ciphertext bit patterns

Caesar Cipher

- earliest known substitution cipher
- by Julius Caesar
- first attested use in military affairs
- replaces each letter by 3rd letter on
- example:

  meet me after the toga party
  PHHW PH DIWHU WKH WRJD SDUWB

Caesar Cipher

- can define transformation as:

  a b c d e f g h i j k l m n o p q r s t u v w x y z
  D E F G H I J K L M N O P Q R S T U V W X Y Z A B C

- mathematically give each letter a number

a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25

- then have Caesar cipher as:

$c = \mathrm{E}(p) = (p + k) \bmod (26)$

$p = \mathrm{D}(c) = (c - k) \bmod (26)$

Cryptanalysis of Caesar Cipher

- only have 26 possible ciphers
  - A maps to A,B,..Z
- could simply try each in turn
- a **brute force search**
- given ciphertext, just try all shifts of letters
- do need to recognize when have plaintext
- eg. break ciphertext "GCUA VQ DTGCM"

Monoalphabetic Cipher

- rather than just shifting the alphabet
- could shuffle (jumble) the letters arbitrarily
- each plaintext letter maps to a different random ciphertext letter
- hence key is 26 letters long

Plain: abcdefghijklmnopqrstuvwxyz
Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

Plaintext: ifwewishtoreplaceletters
Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA

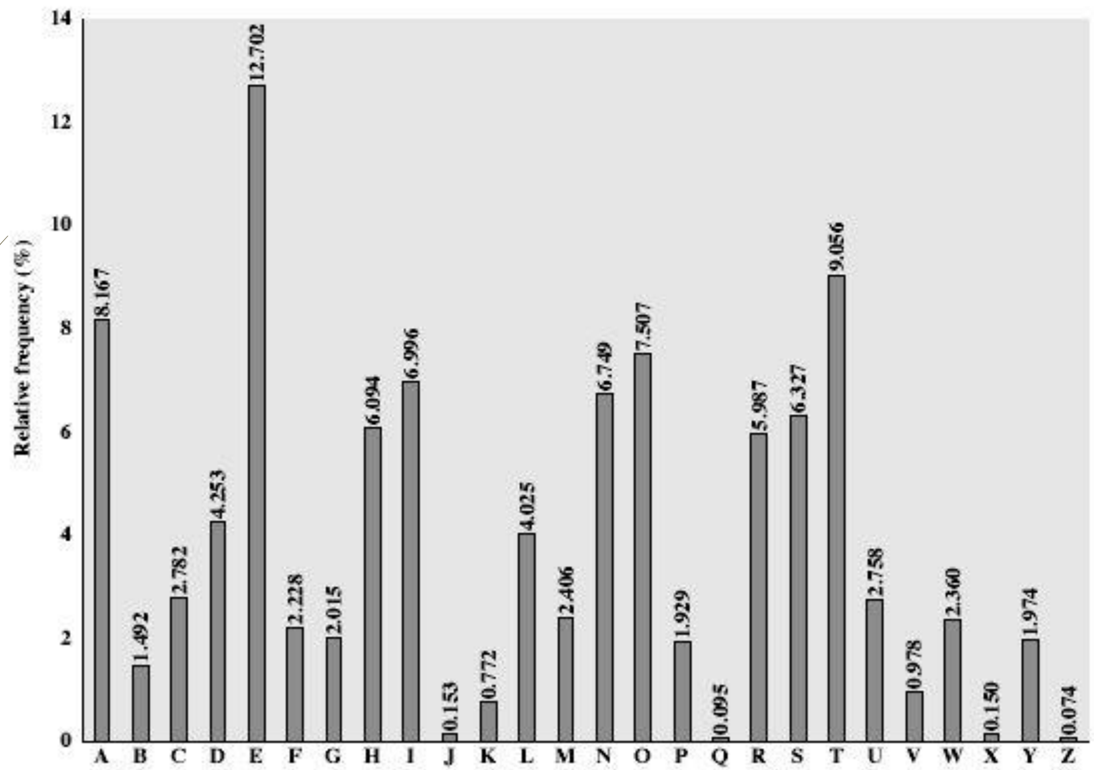
Monoalphabetic Cipher Security

- now have a total of 26! = 4 x 1026 keys
- with so many keys, might think is secure
- but would be **!!!WRONG!!!**
- problem is language characteristics

Language Redundancy and Cryptanalysis

- human languages are **redundant**
- eg "th lrd s m shphrd shll nt wnt"
- letters are not equally commonly used
- in English E is by far the most common letter
  - followed by T,R,N,I,O,A,S
- other letters like Z,J,K,Q,X are fairly rare
- have tables of single, double & triple letter frequencies for various languages

English Letter Frequencies



Use in Cryptanalysis

- key concept - monoalphabetic substitution ciphers do not change relative letter frequencies
- discovered by Arabian scientists in 9th century
- calculate letter frequencies for ciphertext
- compare counts/plots against known values
- if caesar cipher look for common peaks/troughs
  - peaks at: A-E-I triple, NO pair, RST triple
  - troughs at: JK, X-Z
- for monoalphabetic must identify each letter
  - tables of common double/triple letters help

Example Cryptanalysis

- given ciphertext:

```
UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZ
VUEPHZHMDZSHZOWSFPAPPDTSVPQUZWYMXUZUHSX
EPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ
```

- count relative letter frequencies (see text)
- guess P & Z are e and t
- guess ZW is th and hence ZWP is the
- proceeding with trial and error finally get:

```
it was disclosed yesterday that several informal but
direct contacts have been made with political
representatives of the viet cong in moscow
```

Playfair Cipher

- not even the large number of keys in a monoalphabetic cipher provides security
- one approach to improving security was to encrypt multiple letters
- the **Playfair Cipher** is an example
- invented by Charles Wheatstone in 1854, but named after his friend Baron Playfair

Playfair Key Matrix

- a 5X5 matrix of letters based on a keyword
- fill in letters of keyword (sans duplicates)
- fill rest of matrix with other letters
- eg. using the keyword MONARCHY

M	O	N	A	R
C	H	Y	B	D
E	F	G	I/J	K
L	P	Q	S	T
U	V	W	X	Z

Encrypting and Decrypting

- plaintext is encrypted two letters at a time
  1. if a pair is a repeated letter, insert filler like 'X’
  2. if both letters fall in the same row, replace each with letter to right (wrapping back to start from end)
  3. if both letters fall in the same column, replace each with the letter below it (again wrapping to top from bottom)
  4. otherwise each letter is replaced by the letter in the same row and in the column of the other letter of the pair

Security of Playfair Cipher

- security much improved over monoalphabetic
- since have 26 x 26 = 676 digrams
- would need a 676 entry frequency table to analyse (verses 26 for a monoalphabetic)
- and correspondingly more ciphertext
- was widely used for many years
  - eg. by US & British military in WW1
- it **can** be broken, given a few hundred letters
- since still has much of plaintext structure

Polyalphabetic Ciphers

- **polyalphabetic substitution ciphers**
- improve security using multiple cipher alphabets
- make cryptanalysis harder with more alphabets to guess and flatter frequency distribution
- use a key to select which alphabet is used for each letter of the message
- use each alphabet in turn
- repeat from start after end of key is reached

Vigenère Cipher

- simplest polyalphabetic substitution cipher
- effectively multiple caesar ciphers
- key is multiple letters long $K = k_1 \ k_2 \ldots k_d$
- i^{th} letter specifies i^{th} alphabet to use
- use each alphabet in turn
- repeat from start after d letters in message
- decryption simply works in reverse

Example of Vigenère Cipher

- write the plaintext out
- write the keyword repeated above it
- use each key letter as a caesar cipher key
- encrypt the corresponding plaintext letter
- eg using keyword *deceptive*

key: deceptivedeceptivedeceptive

plaintext: wearediscoveredsaveyourself

ciphertext:ZICVTWQNGRZGVTWAVZHCQYGLMGJ

Aids

- simple aids can assist with en/decryption
- a **Saint-Cyr Slide** is a simple manual aid
  - a slide with repeated alphabet
  - line up plaintext 'A' with key letter, eg 'C'
  - then read off any mapping for key letter
- can bend round into a **cipher disk**
- or expand into a **Vigenère Tableau**

Security of Vigenère Ciphers

- have multiple ciphertext letters for each plaintext letter
- hence letter frequencies are obscured
- but not totally lost
- start with letter frequencies
  - see if look monoalphabetic or not
- if not, then need to determine number of alphabets, since then can attach each

Kasiski Method

- method developed by Babbage / Kasiski
- repetitions in ciphertext give clues to period
- so find same plaintext an exact period apart
- which results in the same ciphertext
- of course, could also be random fluke
- eg repeated “VTW” in previous example
- suggests size of 3 or 9
- then attack each monoalphabetic cipher individually using same techniques as before

Autokey Cipher

- ideally want a key as long as the message
- Vigenère proposed the **autokey** cipher
- with keyword is prefixed to message as key
- knowing keyword can recover the first few letters
- use these in turn on the rest of the message
- but still have frequency characteristics to attack
- eg. given key *deceptive*

key: deceptivewearediscoveredsav
plaintext: wearediscoveredsaveyourself
ciphertext:ZICVTWQNGKZEIIGASXSTSLVVWLA

One-Time Pad

- if a truly random key as long as the message is used, the cipher will be secure
- called a One-Time pad
- is unbreakable since ciphertext bears no statistical relationship to the plaintext
- since for **any plaintext** & **any ciphertext** there exists a key mapping one to other
- can only use the key **once** though
- problems in generation & safe distribution of key

Transposition Ciphers

- now consider classical **transposition** or **permutation** ciphers
- these hide the message by rearranging the letter order
- without altering the actual letters used
- can recognise these since have the same frequency distribution as the original text

Rail Fence cipher

- write message letters out diagonally over a number of rows
- then read off cipher row by row
- eg. write message out as:

```
m e m a t r h t g p r y
 e t e f e t e o a a t
```

- giving ciphertext

MEMATRHTGPRYETEFETEOAAT

Row Transposition Ciphers

- a more complex transposition
- write letters of message out in rows over a specified number of columns
- then reorder the columns according to some key before reading off the rows

Key: 3 4 2 1 5 6 7
Plaintext: a t t a c k p
o s t p o n e
d u n t i l t
w o a m x y z
Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

Product Ciphers

- ciphers using substitutions or transpositions are not secure because of language characteristics
- hence consider using several ciphers in succession to make harder, but:
  - two substitutions make a more complex substitution
  - two transpositions make more complex transposition
  - but a substitution followed by a transposition makes a new much harder cipher
- this is bridge from classical to modern ciphers

Rotor Machines

- before modern ciphers, rotor machines were most common complex ciphers in use
- widely used in WW2
  - German Enigma, Allied Hagelin, Japanese Purple
- implemented a very complex, varying substitution cipher
- used a series of cylinders, each giving one substitution, which rotated and changed after each letter was encrypted
- with 3 cylinders have 26^3=17576 alphabets

Hagelin Rotor Machine

Steganography

- an alternative to encryption
- hides existence of message
  - using only a subset of letters/words in a longer message marked in some way
  - using invisible ink
  - hiding in LSB in graphic image or sound file
- has drawbacks
  - high overhead to hide relatively few info bits

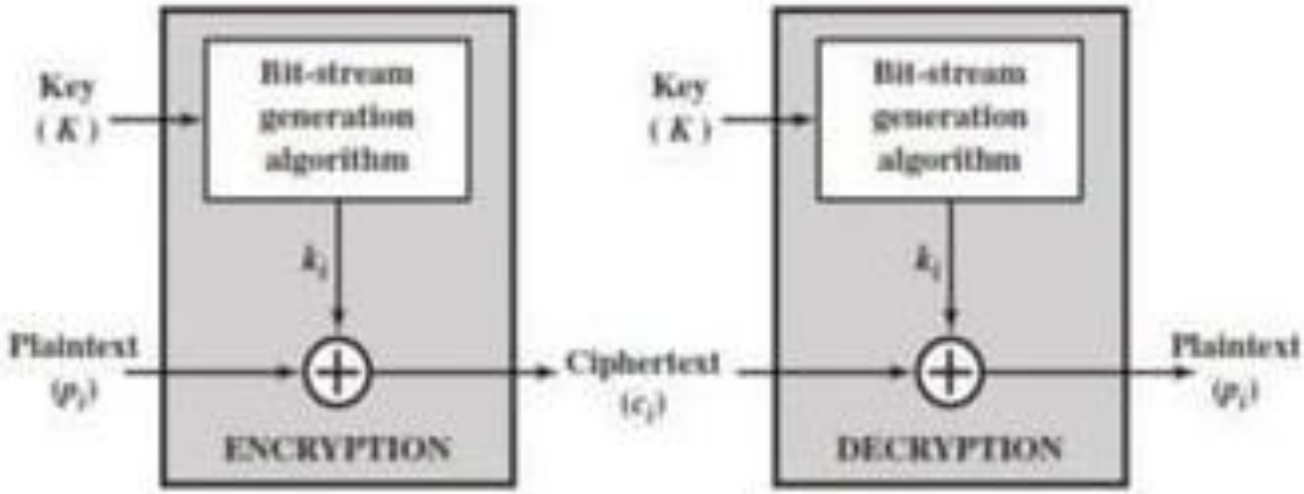
Block Ciphers and
Data Encryption Standard

Traditional Block Cipher Structure

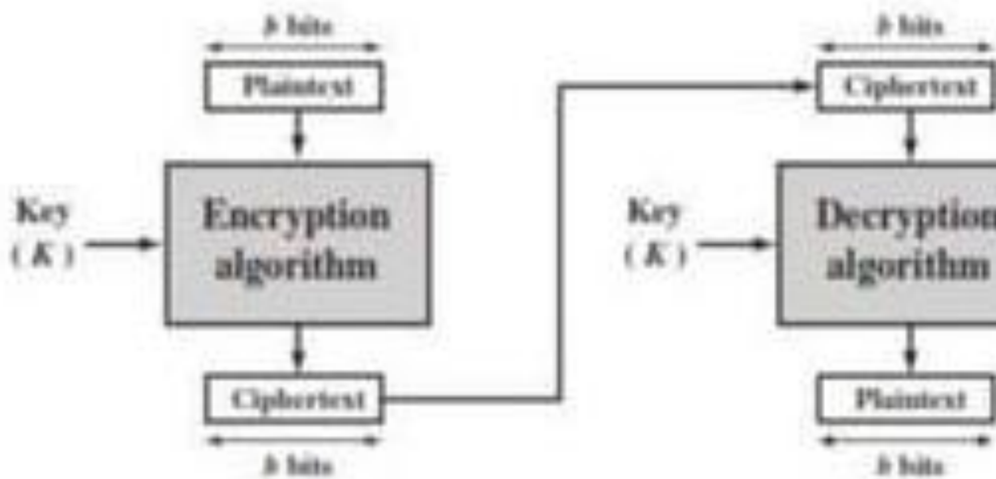
- A block cipher is an encryption/decryption scheme in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length.
- Many block ciphers have a Feistel structure. Such a structure consists of a number of identical rounds of processing. In each round, a substitution is performed on one half of the data being processed, followed by a permutation that interchanges the two halves. The original key is expanded so that a different key is used for each round.
- The Data Encryption Standard (DES) has been the most widely used encryption algorithm until recently. It exhibits the classic Feistel structure. DES uses a 64-bit block and a 56-bit key.

Stream Ciphers and Block Ciphers

- A **stream cipher** is one that encrypts a digital data stream one bit or one byte at a time. Examples of classical stream ciphers are the autokeyed Vigenère cipher and the Vernam cipher.
- A **block cipher** is one in which a block of plaintext is treated as a whole and used to produce a ciphertext block of equal length.
- Typically, a block size of 64 or 128 bits is used. As with a stream cipher, the two users share a symmetric encryption key



(a) Stream cipher using algorithmic bit-stream generator



(b) Block cipher

Motivation for the Feistel Cipher Structure

- A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits.
  - Each plaintext must produce a unique ciphertext block (for decryption to be possible).
  - Such transformation is called reversible or nonsingular.

Reversible Mapping	
Plaintext	**Ciphertext**
00	11
01	10
10	00
11	01

Irreversible Mapping	
Plaintext	**Ciphertext**
00	11
01	10
10	01
11	01

Motivation for the Feistel Cipher Structure

- The logic of a general substitution cipher. (for $n = 4$)

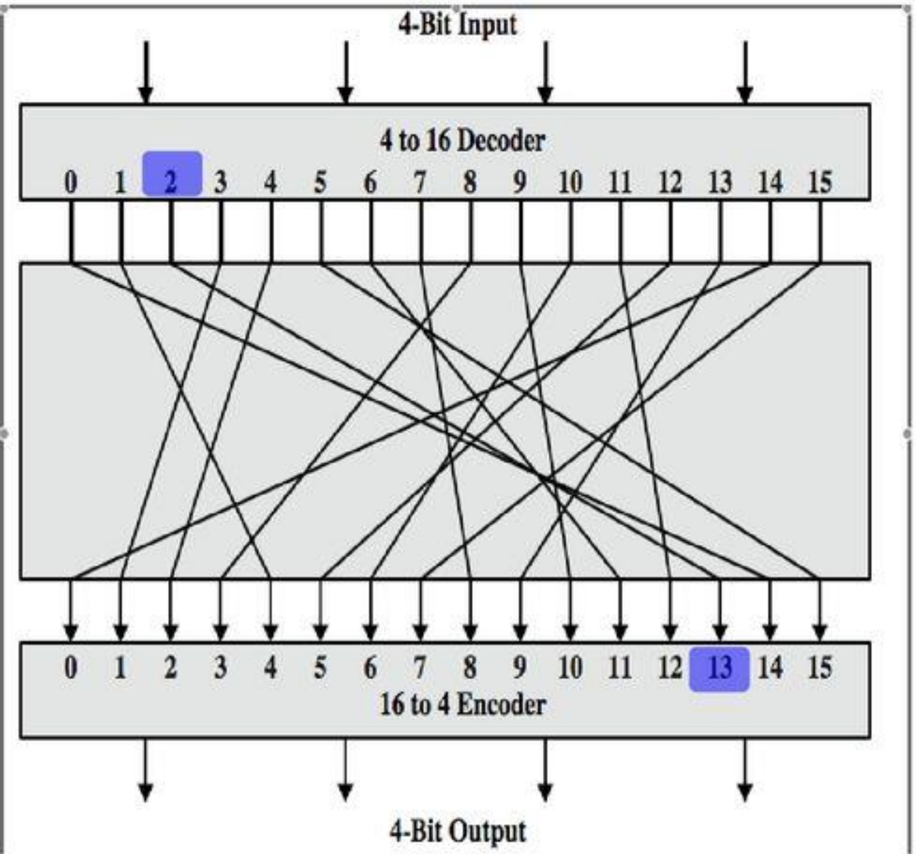


Table 3.1 Encryption and Decryption Tables for Substitution Cipher of Figure 3.4

Plaintext	Ciphertext
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Ciphertext	Plaintext
0000	1110
0001	0011
0010	0100
0011	1000
0100	0001
0101	1100
0110	1010
0111	1111
1000	0111
1001	1101
1010	1001
1011	0110
1100	1011
1101	0010
1110	0000
1111	0101

Motivation for the Feistel Cipher Structure

- A practical problem with the general substitution cipher
  - If a small block size is used, then the system is equivalent to a classical substitution cipher.
    - Such systems are vulnerable to a statistical analysis of the plaintext.
  - If block size is sufficiently large and an arbitrary reversible substitution is allowed, then statistical analysis is infeasible.
    - This is not practical from a performance point of view.
    - For n-bit block cipher, the key size is n X 2^n bits.
    - For n = 4, the key size is 4 x 16 = 64 bits.
    - For n = 64, the key size is 64 x 2^{64} bits

The Feistel Cipher

- Feistel proposed the ideal block cipher by utilizing the concept of a product cipher, which is the execution of two or more simple cipher in sequence in such a way that the final result or product is cryptographically stronger.

- Feistel proposed the use of a cipher that alternates substitution and permutations.

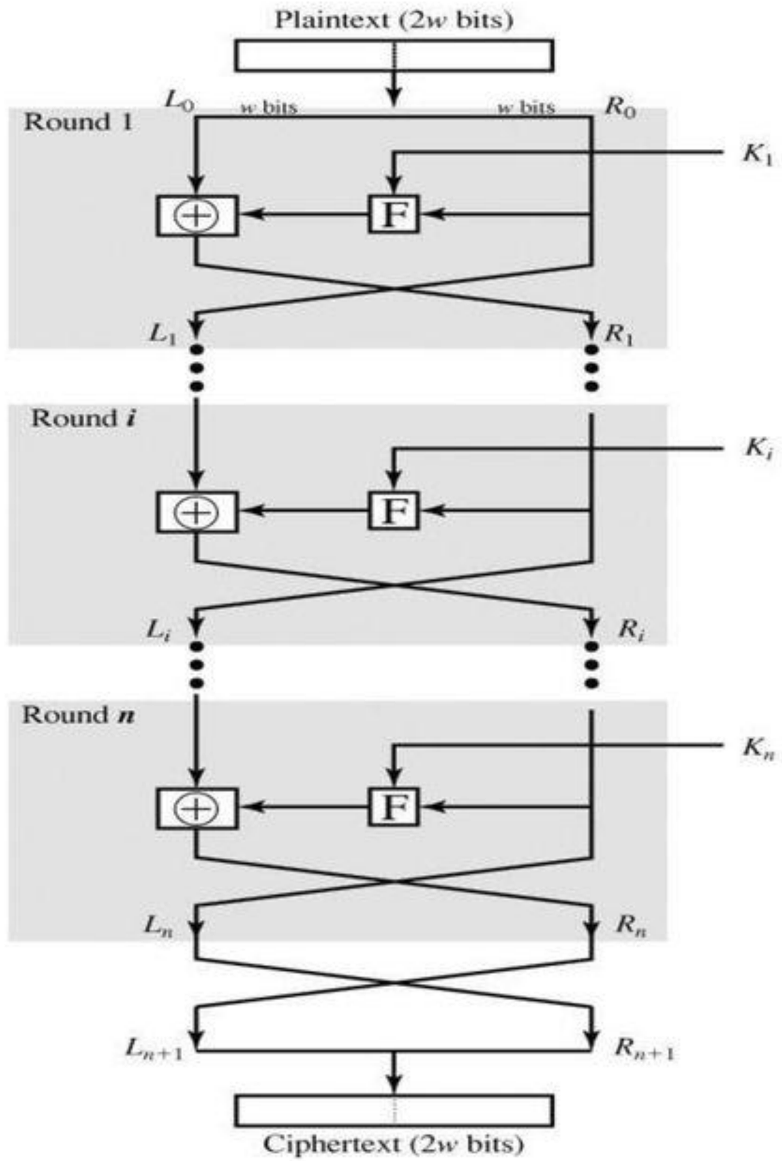
- In fact, this is a practical application of a proposal by Claude Shannon to develop a product cipher that alternates *confusion* and *diffusion* functions.

Diffusion and Confusion

- Diffusion (achieved by transposition technique)
  - To make the statistical relationship between the plaintext and ciphertext as complex as possible in order to thwart attempts to discover the key.
  - Block cipher relies on stream and block cipher.
- Confusion (achieved by substitution technique)
  - To make the relationship between the statistics of the ciphertext and the value of the encryption key as complex as possible to thwart attempts to discover the key.
  - Stream cipher relies only on confusion.

Feistel Cipher Structure

- Feistel structure
  - Input
    - Plaintext : $2w$ bits
    - A Key K
  - Output
    - Ciphertext : $2w$ bits



Feistel Cipher Structure

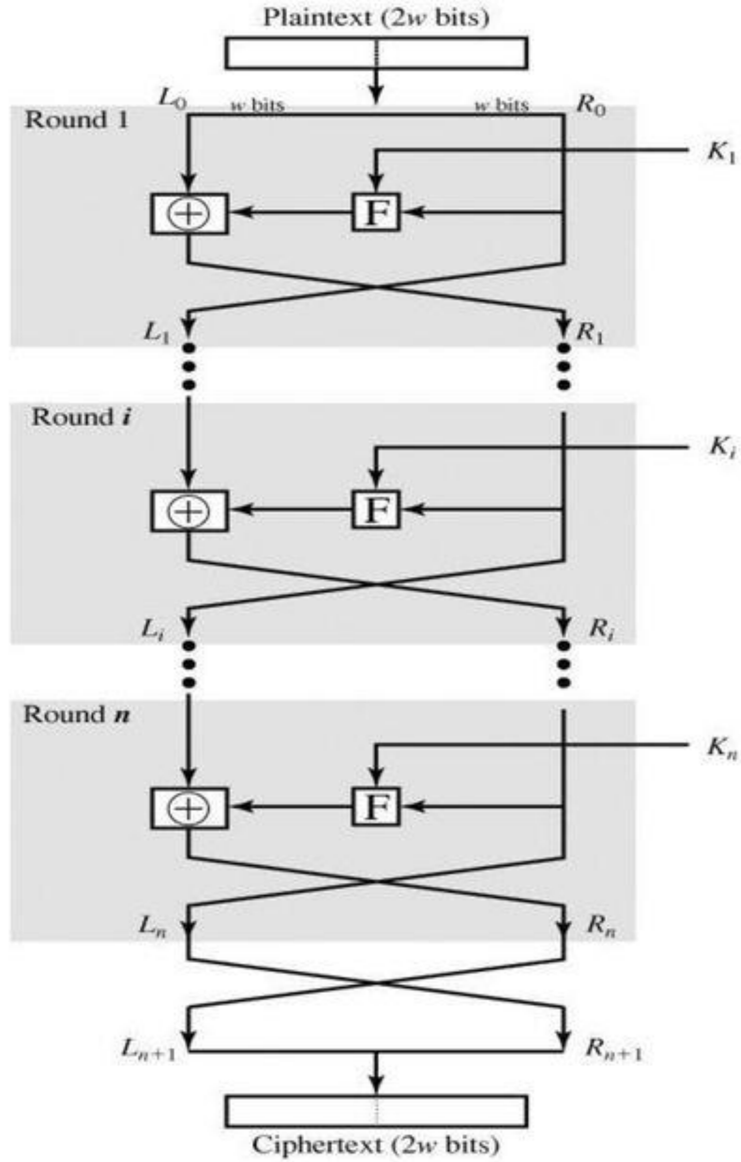
- The input is divided into two halves L_0 and R_0 and they pass through n rounds.

- Round i
  - Input: $\boldsymbol{L_{i-1}}, \boldsymbol{R_{i-1}}$, and $\boldsymbol{K_i}$ (round key)
  - Output: $\boldsymbol{L_i}$ and $\boldsymbol{R_i}$
  - A substitution is performed on the left half L_{i-1}.

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

  - A permutation is performed by swapping the two halves.

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



Feistel Cipher Structure

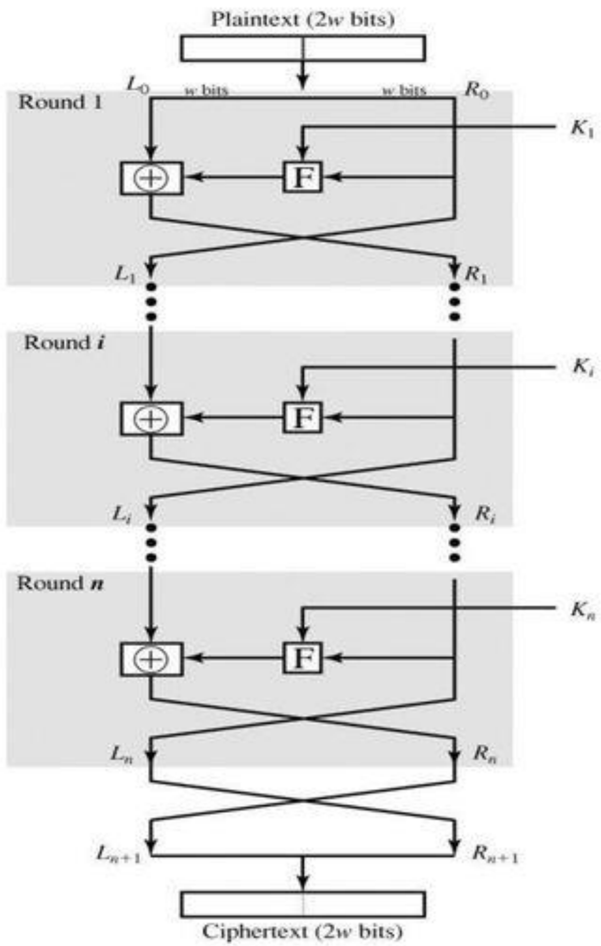
- Design features
  - **Block size**
    - The larger it is, the securer the cipher is but the slower the cipher is.
    - 64 or 128 bits
  - **Key size**
    - The larger it is, the securer the cipher is but the slower the cipher is.
    - 64 or 128 bits
  - **Number of rounds**
    - The larger it is, the securer the cipher is but the slower the cipher is.
    - 16 rounds is typical.

Feistel Cipher Structure

- Design features
  - **Subkey generation**
    - The more complex it is, the securer the cipher is but the slower…
  - **Round function**
    - The more complex it is, the securer the cipher is but the slower…
  - **Fast software encryption/decryption**

Feistel Decryption Algorithm

- Decryption is the same as the encryption except that the subkeys are used in reverse order.



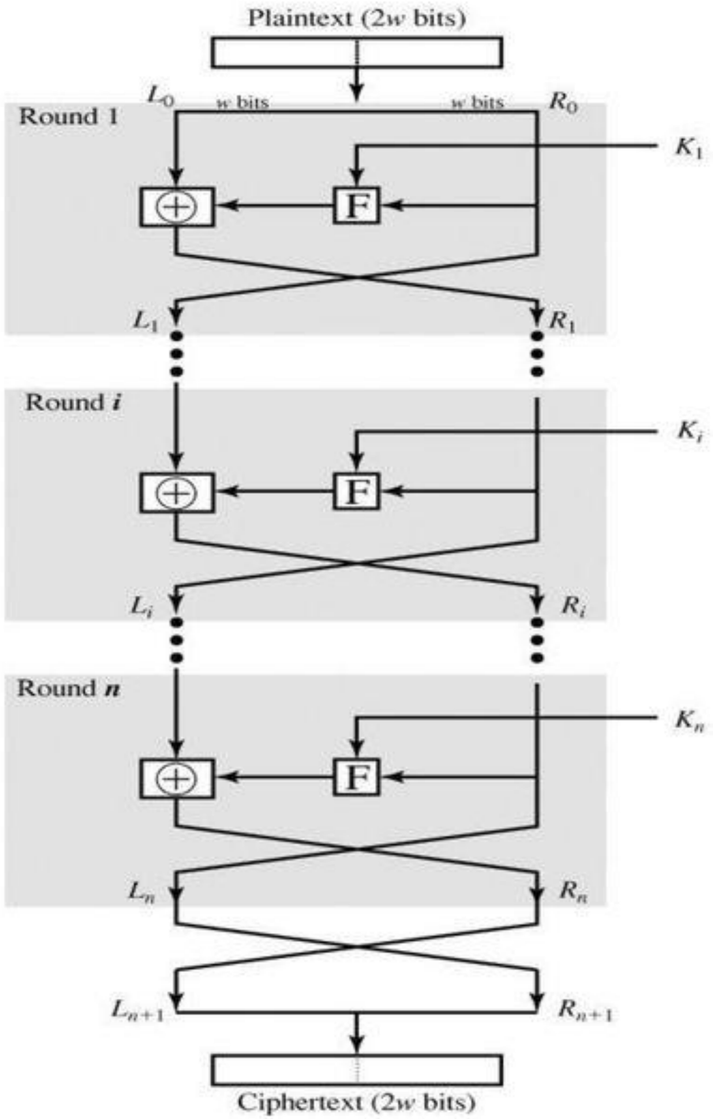
Feistel Cipher Structure

- Round i

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

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

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



The Data Encryption Standard

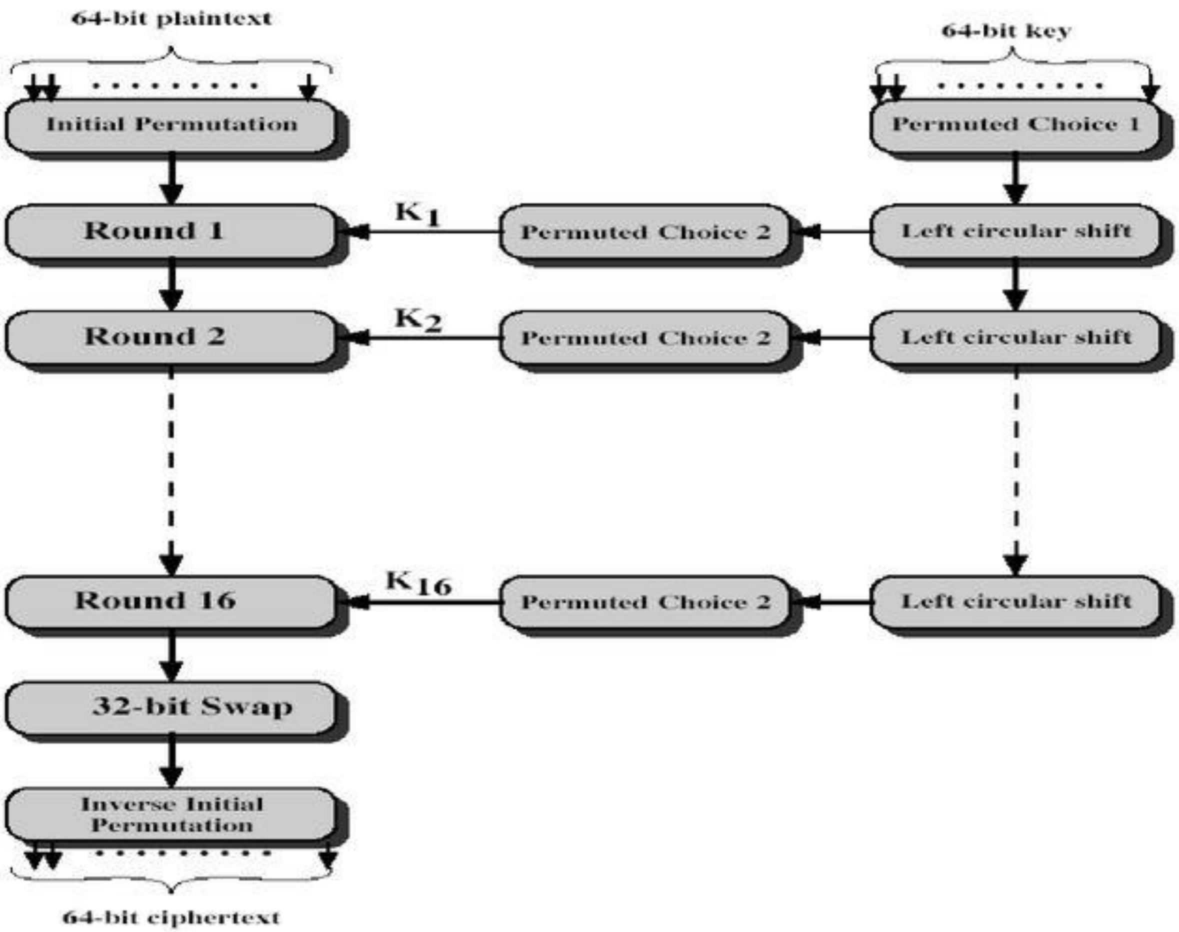
- DES Encryption
  - Initial Permutation
  - Details of Single Round
  - Key Generation
- The Avalanche Effect

The Data Encryption Standard

- The most widely used encryption.
  - Adopted in 1977 by NIST
  - FIPS PUB 46

- Data are encrypted in 64-bit blocks using a 56-bit key.

DES Encryption



- **DES is a Feistel cipher with the exception of IP and IP^{-1}.**

Initial Permutation

- The permutation
  - X = IP(M)

- The inverse permutation
  - Y = $IP^{-1}(X) = IP^{-1}(IP(M))$
  - The original ordering is restored

(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

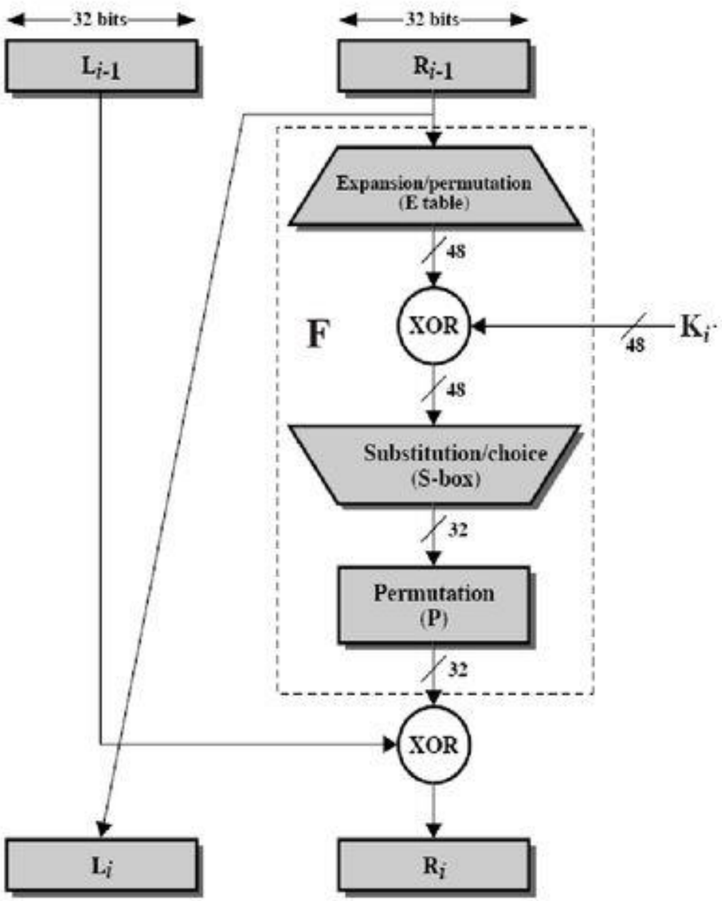
(b) Inverse Initial Permutation (IP^{-1})

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

Single Round

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

- **F** function
  - $\mathbf{R}_{i\text{-}1}$ is expanded to 48-bits using **E**.
  - The result is XORed with the 48-bit round key.
  - The 48-bit is substituted by a 32-bit.
  - The 32-bit is permuted by **P**.



Single Round

- Expansion *E*
  - 32 bits → 48 bits
  - 16 bits are reused.

(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

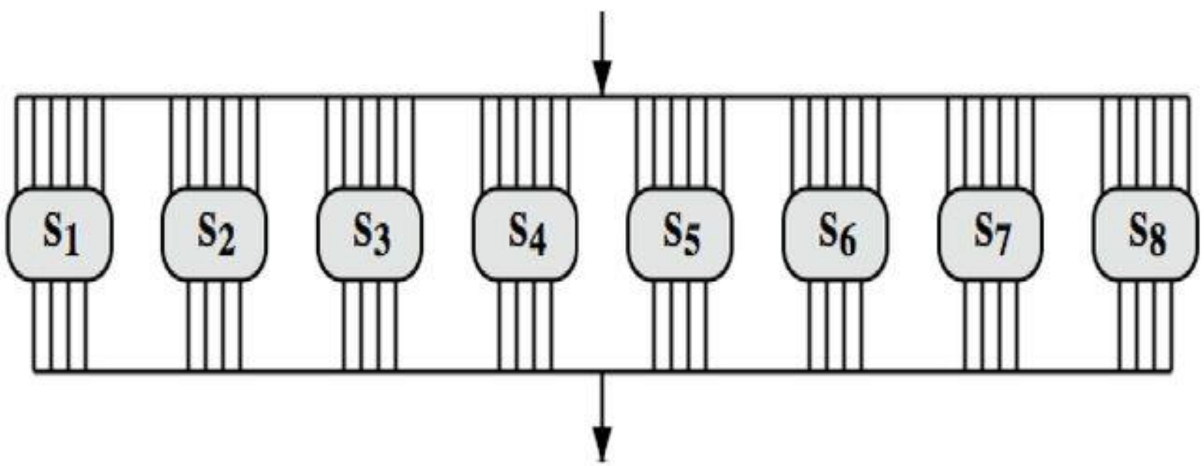
- Permutation *P*

(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

Single Round

- Substitution
  - 48 bits → 32 bits
  - 8 S-boxes
  - Each S-box gets 6 bits and outputs 4 bits.



Single Round

- Each S-box is given in page 79.
  - Outer bits 1 & 6 (**row** bits) select one rows
  - Inner bits 2-5 (**col** bits) are substituted
    - Example : Input : 011001
      - the row is 01 (row 1)
      - the column is 1100 (column 12)
      - Output is 1001

S_1

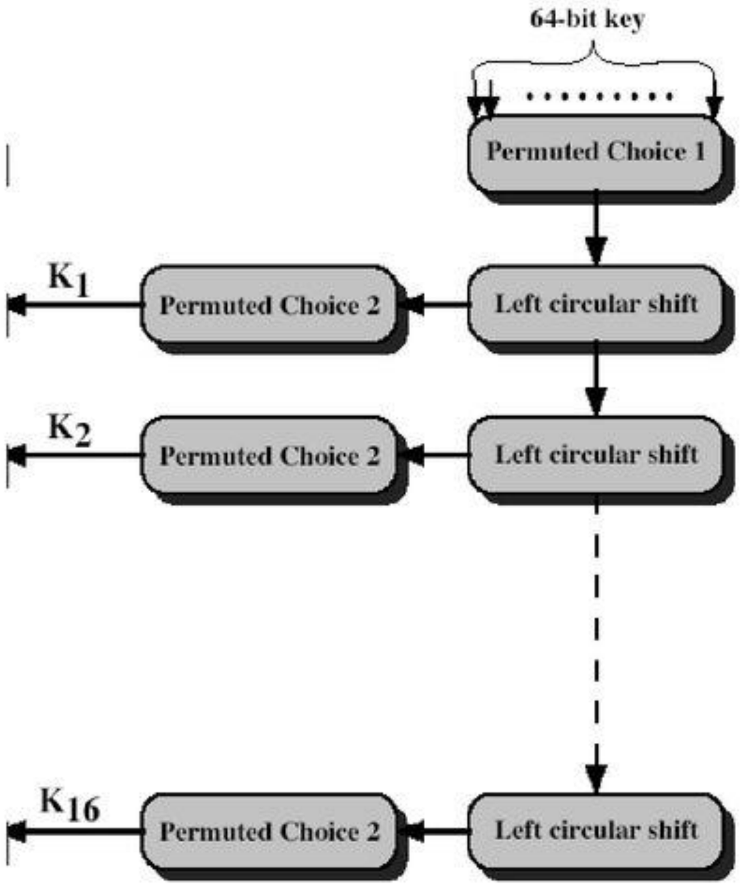
Column Number

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

Key Generation

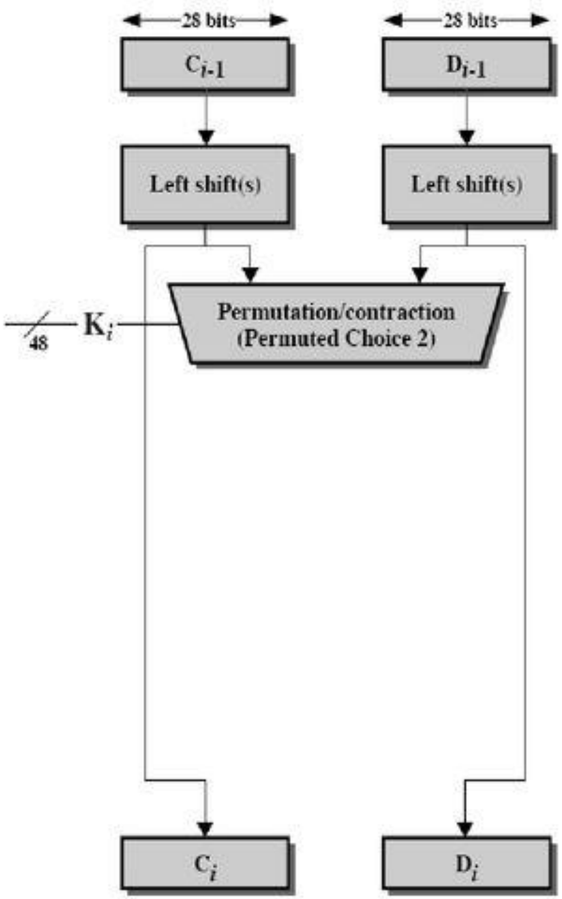
- A 64-bit key used as input
  - Every 8th bit is ignored.
  - Thus, the key is 56 bits.

- **PC1** permute 56 bits into two 28-bit halves.



Key Generation

- In each round,
  - each 28 bits are rotated left and
  - 24 bits are selected from each half.



l of DES Algorithm

Key Generation

Table 3.4 DES Key Schedule Calculation

(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

Key Generation

(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

DES Decryption

- Decryption uses the same algorithm as encryption.
  - Feistel cipher
  - Roundkey schedule is reversed.

The Avalanche Effect

- A small change of plaintext or key produces a significant change in the ciphertext.

- DES exhibits a strong avalanche effect.

The Avalanche Effect

- Example

Plaintext 1	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
Plaintext 2	10000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
Key	00000001	1001011	0100100	1100010	0011100	0011000	0011100	0110010

(a) Change in Plaintext

Round	Number of bits that differ
0	1
1	6
2	21
3	35
4	39
5	34
6	32
7	31
8	29
9	42
10	44
11	32
12	30
13	30
14	26
15	29
16	34

The Avalanche Effect

- Example

plaintext	01101000	10000101	00101111	01111010	00010011	01110110	11101011	10100100
Key 1	1110010	1111011	1101111	0011000	0011101	0000100	0110001	11011100
Key 2	0110010	1111011	1101111	0011000	0011101	0000100	0110001	11011100

(b) Change in Key

Round	Number of bits that differ
0	0
1	2
2	14
3	28
4	32
5	30
6	32
7	35
8	34
9	40
10	38
11	31
12	33
13	28
14	26
15	34
16	35

The Strength of DES

- The Use of 56-bit keys
- The Nature of the DES Algorithm
- Timing Attacks

The Use of 56-bit Keys

- If the key length is 56-bit, we have $2^{56} = 7.2 \times 10^{16}$ keys.
- In 1998, Electronic Frontier Foundation (EFF) announced 'DES cracker' which can attack DES in 3 days.
  - It was built for less than $250,000.
- Alternatives to DES
  - AES (key size is 128 ~ 256 bit) and triple DES (112 ~ 168 bit)

Differential and Linear Cryptanalysis

- Differential Cryptanalysis
  - History
  - Differential Cryptanalysis Attack

- Linear Cryptanalysis

Differential Cryptanalysis

- One of the most significant advances in cryptanalysis in recent years is differential cryptanalysis.

History

- Murphy, Biham & Shamir published 1990.

- The first published attack that is capable of breaking DES in less than 2^{55} complexity.
  - As reported, can successfully cryptanalyze DES with an effort on the order of 2^{47}, requiring chosen plaintexts.

- This is a powerful tool, but it does not do very well against DES
  - Differential cryptanalysis was known to IBM as early as 1974

Differential Cryptanalysis Attack

- The differential cryptanalysis attack is complex.

- Change in notation for DES
  - Original plaintext block : m
    - Two halves : m_0, m_1

- At each round for DES, only one new 32-bit block is created.
  - The intermediate message halves are related.

$$m_{i+1} = m_{i-1} \oplus \mathrm{f}(m_i, K_i) \qquad i = 1,2,\ldots,16$$

Differential Cryptanalysis Attack

- Start with two messages *m* and *m'*, and consider the difference between the intermediate message halves :
  - With a known XOR difference

$$\Delta m_i = m_i \oplus m'_i$$

- Then

$$\Delta m = m \oplus m'$$

$$\begin{aligned}\Delta m_{i+1} &= m_{i+1} \oplus m'_{i+1} \\ &= [m_{i-1} \oplus f(m_i, K_i)] \oplus [m'_{i-1} \oplus f(m'_i, K_i)] \\ &= m_{i-1} \oplus m'_{i-1} \oplus f(m_i, K_i) \oplus f(m'_i, K_i) \\ &= \Delta m_{i-1} \oplus [f(m_i, K_i) \oplus f(m'_i, K_i)]\end{aligned}$$

Differential Cryptanalysis Attack

- The Overall strategy is based one these considerations for a single round.
  - The procedure is
    - to begin with two plaintext message *m* and *m'* with a given difference.
    - to trace through a probable pattern of differences after each round to yield a probable difference for the ciphertext.

Differential Cryptanalysis Attack

- Actually, there are two probable differences for the two 32-bit halves. $\Delta m_{17} \| \Delta m_{16}$
- Next, submit m and m' for encryption to determine the actual difference under the unknown key.
- And compare the result to the probable difference.
- If there is a match, $E_K(m) \oplus E_K(m') = (\Delta m_{17} \| \Delta m_{16})$
  - Then, suspect that all the probable patterns at all the intermediate rounds are correct.
- With that assumption, can make some deductions about the key bits.

Linear Cryptanalysis

- another recent development
- also a statistical method
- must be iterated over rounds, with decreasing probabilities
- developed by Matsui et al in early 90's
- based on finding linear approximations
- can attack DES with 2^{47} known plaintexts, still in practise infeasible

Linear Cryptanalysis

- find linear approximations with prob p != ½

  `P[i1,i2,...,ia](+)C[j1,j2,...,jb] = K[k1,k2,...,kc]`

  `where ia,jb,kc are bit locations in P,C,K`

- gives linear equation for key bits
- get one key bit using max likelihood alg
- using a large number of trial encryptions
- effectiveness given by: `|p–½|`

Block Cipher Design Principles

- DES Design Criteria
- Number of Rounds
- Design of Function F
  - Design Criteria for F
  - S-Box Design
- Key Schedule Algorithm

Block Cipher Design Principles

- Although much progress has been made that are cryptographically strong, the basic principles have not changed all.

DES Design Criteria

- Focused on the design of the S-boxes and on the P function.
- The criteria for the S-boxes.
  - No output bit of any S-box should be too close a linear function of the input bits.
  - Each row of an S-box should include all 16 possible output bit combinations
  - If two inputs differ in exactly one bit, the outputs must differ in at least two bits.
  - If two inputs differ in the two middle bits exactly, the outputs must differ in at least two bits.

DES Design Criteria

- The criteria for the S-boxes (~ continue)
  - If two inputs 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.

DES Design Criteria

- The criteria for the permutation P
  - The four output bits from each S-box at round i are distributed so that two of them affect "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-boxes.
  - 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 .
- These criteria are intended to increase the diffusion of the algorithm.

Number of Rounds

- The greater the number of rounds, the more difficult it is to perform cryptanalysis, even for a relatively weak F.

- This criterion is attractive because it makes it easy to judge the strength of an algorithm and to compare different algorithms.

Design of Function F

- The heart of a Feistel block cipher is the function F.
- The function F provides the element of confusion.
  - One obvious criterion is that F be nonlinear.
    - The more nonlinear F, the more difficult.
  - Have good avalanche properties.
    - Strict Avalanche Criterion (SAC)
  - The bit independence criterion (BIC)
    - 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.
- 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.
    - Blowfish has 8×32 S-boxes.
  - Larger S-boxes are more resistant to differential and linear cryptanalysis.
    - For practical reasons, a limit of n equal to about 8 to 10 is usually imposed.

S-Box Design

- S-boxes are typically organized in a different manner than used in DES.
  - An $n \times m$ S-box typically consists of 2^n rows of m bits each.
  - Example, in an 8×32 S-box
    - If the input is 00001001, the output consists of the 32 bits in row 9.

S-Box Design

- Mister and Adams proposed for S-box design.
  - S-box should satisfy both SAC and BIC.
  - All linear combinations of S-box columns should be *bent(boolean function)*.
    - Bent functions
      - A special class of Boolean functions that are highly nonlinear according to certain mathematical criteria.

- Increasing interest in designing and analyzing S-boxes using bent functions.

S-Box Design

- Heys, H. and Tavares, S. proposed for S-boxes.
  - Guaranteed avalanche (GA) criterion
  - An S-box satisfies GA of order if, at least output bits change.
  - Conclude that a GA in the range γ of order 2 to γ order 5 provides strong diffusion characteristics for the overall encryption algorithm.

S-Box Design

- Best method of selecting the S-box entries.
  - 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.
    - 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.

Key Schedule Algorithm

- With any Feistel block cipher, the key is used to generate one subkey for each round.
- We would like to select subkeys to maximize the difficulty of deducing individual subkeys and the difficulty of working back to the main key.
- No general principles have not been proposed.
- Hall suggests that the key schedule should guarantee key/ciphertext Strict Avalanche Criterion and Bit Indepence Criterion.