# Block Ciphers, Data Encryption Standard (DES), and TwoFish

(CS-452)

Week 3

# Block Ciphers

- Block ciphers: 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
  - Many current ciphers are block ciphers
  - Broader range of applications
  - DES (Data Encryption Standard): a commonly used cryptographic algorithms, especially in financial applications.

# Block Ciphers: Confusion and Diffusion

- Properties of a secure block cipher
- Developed by Claude Shannon in 1945
- Confusion:
  - Basic idea: obscures the relationship between the ***key*** and the resulting ***ciphertext***
  - Achieved by having each bit of the ciphertext depend on multiple bits of the key
  - If one bit of encryption key changes → many/all bits of the ciphertext will change (in a seemingly random manner), thus making the key difficult to derive from the ciphertext
  - Achieved through ***substitution***
    - Example: a → b substitution performed with Caesar Cipher and key of 1
  - Used by both block and stream ciphers

# Block Ciphers: Confusion and Diffusion

- Properties of a secure block cipher
- Developed by Claude Shannon in 1945
- Diffusion:
  - **Basic idea:** obscuring the relationship between the ***plaintext*** and the resulting ***ciphertext***
  - Dissipates statistics of plaintext into long-range statistics of the ciphertext
  - If one bit of the plaintext changes ➔ then ciphertext will change significantly and in unpredictably (pseudo-randomly)
  - Diffusion is achieved through ***permutation***
  - **Example:** the permutation performed by Row Transposition Cipher
  - Only ***block ciphers*** use diffusion

# Block Cipher Principles

- A block cipher operates on a plaintext block of n bits to produce a ciphertext block of n bits.
- There are possible different plaintext blocks.

# Block Cipher Principles

- 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
- For the encryption to be reversible (for decryption to be possible), each must produce a unique ciphertext block
- Reversible:

| Plaintext | Ciphertext |
|---|---|
| 00 | 11 |
| 01 | 10 |
| 10 | 00 |
| 11 | 01 |

# Block Cipher Principles

- 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,
- for the encryption to be reversible (for decryption to be possible), each must produce a unique ciphertext black
- Irreversible:

| Plaintext | Ciphertext |
|---|---|
| 00 | 11 |
| 01 | 10 |
| 10 | 00 |
| 11 | 00 |

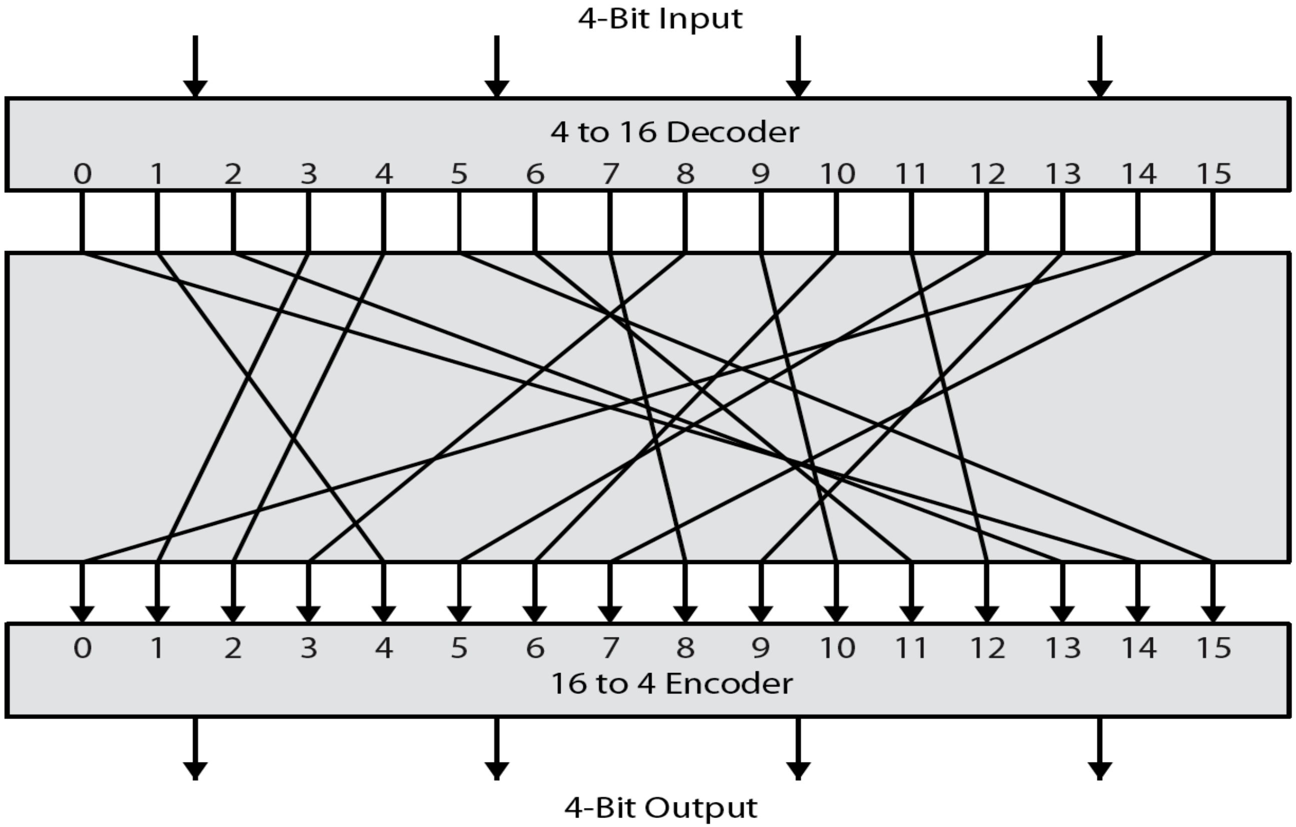
# Block Cipher Principles

- 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,
- for the encryption to be reversible (for decryption to be possible), each must produce a unique ciphertext black
- Irreversible:

| Plaintext | Ciphertext |
|---|---|
| 00 | 11 |
| 01 | 10 |
| 10 | 00 |
| 11 | 00 |

# Ideal Block Cipher

- The logic of a general substitution cipher for n = 4
  - A 4-bit input produces one of 16 possible input states, which is mapped into a unique one of 16 possible output states, each of which is represented by 4 ciphertext bits.



# Ideal Block Cipher

- Ideal block cipher: allows for maximum number of possible encryption mappings from the plaintext block.
  - n bits → possible mappings

# Ideal Block Cipher

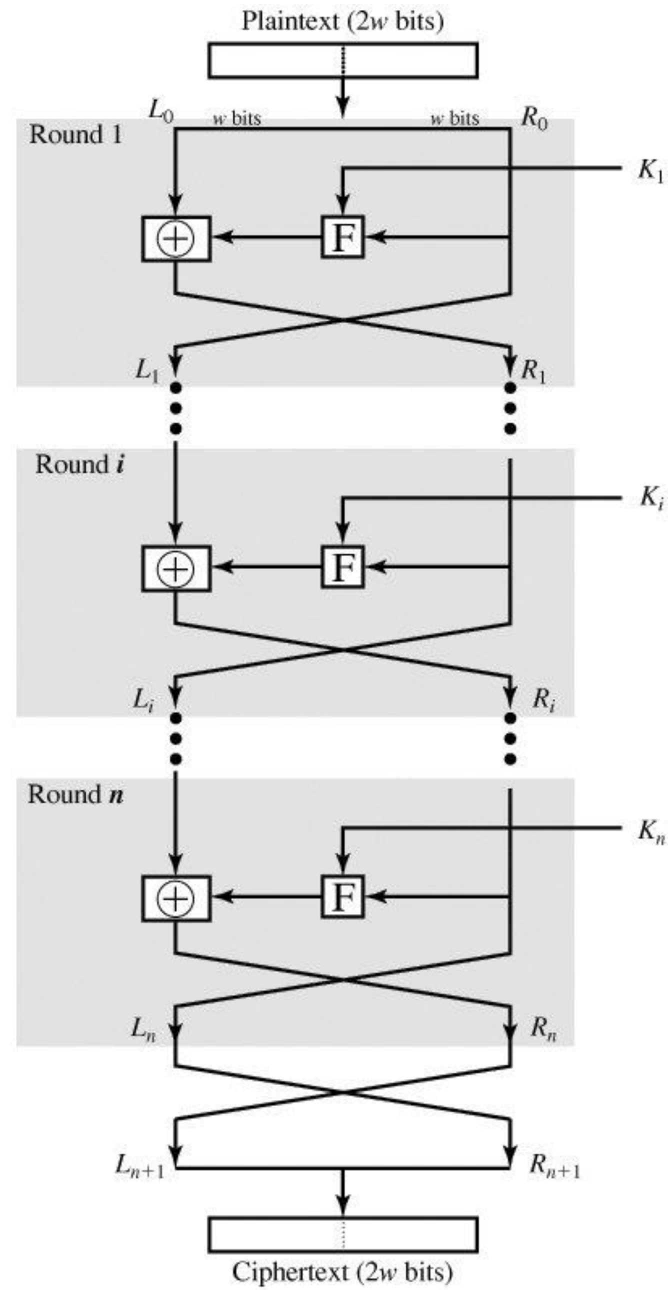
- Ideal block cipher: allows for maximum number of possible encryption mappings from the plaintext block.
  - n bits ➔ $2^n!$ possible mappings
  - Impractical when n is large
    - Each mapping constitutes a key
    - n=64 ➔ the key size is > $63*2^{63}$ -- not practical.

# The Feistel Cipher

- The Feistel cipher: approximate the ideal block cipher by utilizing the concept of a product cipher
  - 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 mappings (rather than $2^n!$ mappings)
  - Alternates substitution and permutation

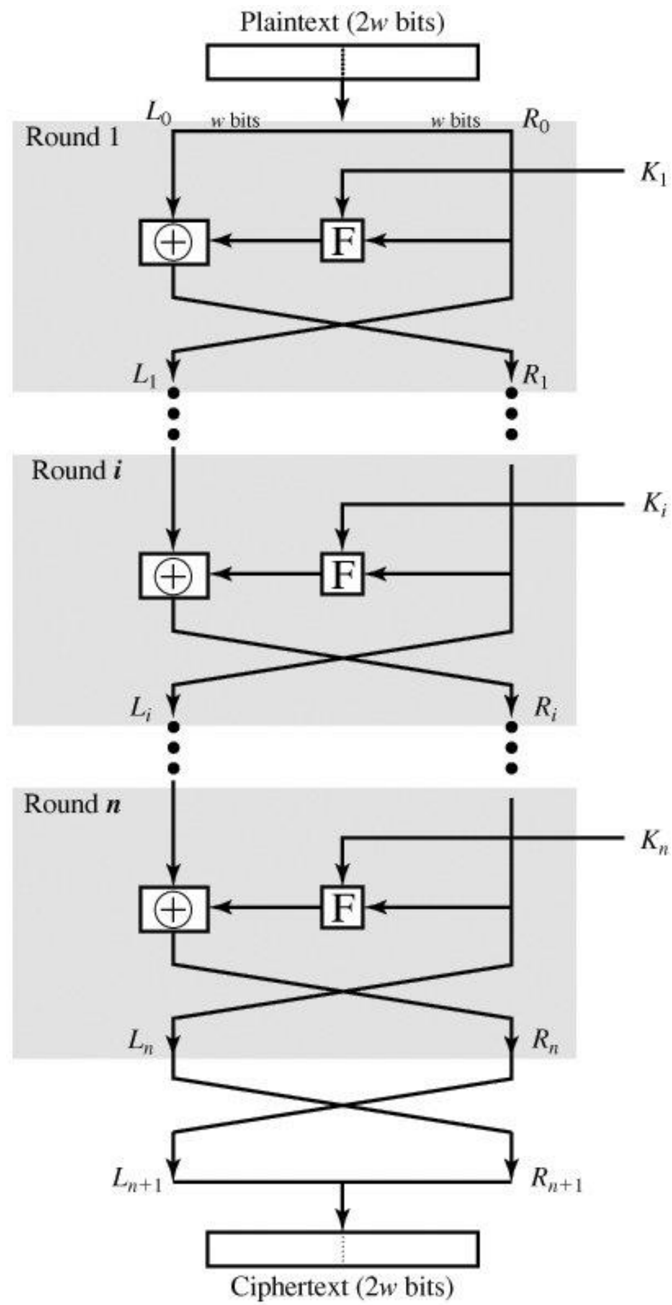
# Feistel Cipher Structure

- Most symmetric block ciphers are based on a Feistel Cipher Structure.
- Inputs: a plaintext block of length 2w bits and a key K.
- The plaintext block is divided into two halves $L_0$, $R_0$
  - Pass through n rounds of processing and then combined to produce the ciphertext block.
  - Each round i has inputs $L_{i-1}$ and $R_{i-1}$ derived from the previous round, as well as a subkey $K_i$ derived from K.
  - In general, $K_i$ are different from K and from each other.



# Feistel Cipher Structure

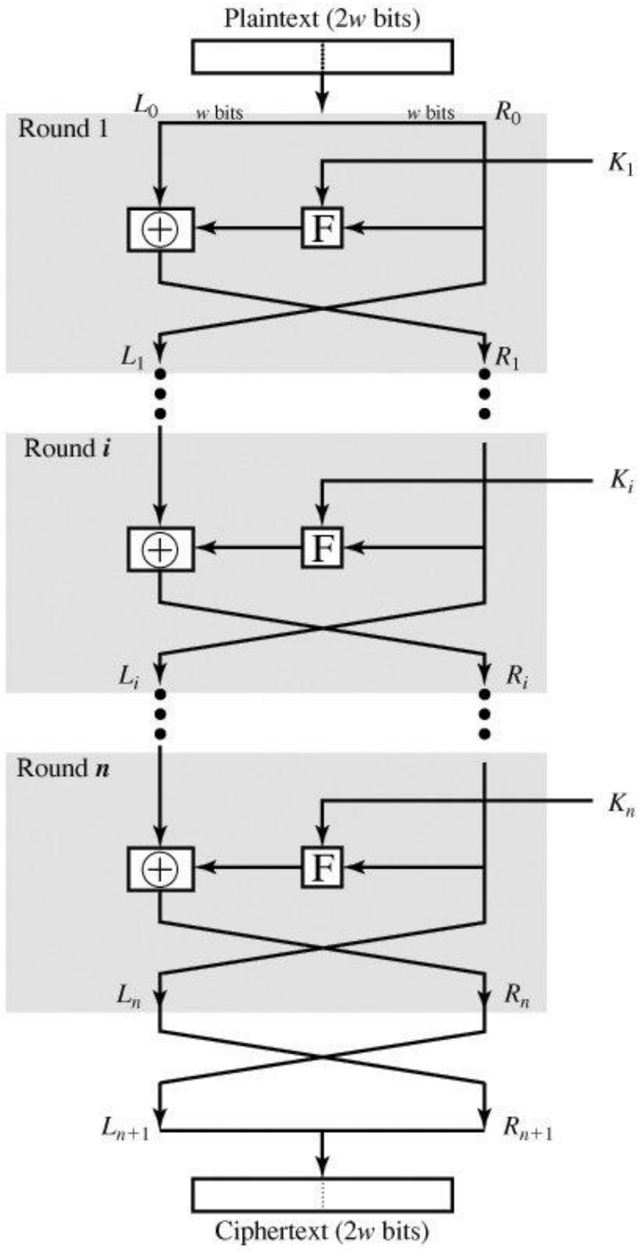
- A substitution is applied to the left half
  - Applying a round function F to the right half of the data. F is parameterized by the round subkey $K_i$
  - Then take the exclusive-OR (XOR) of the output of F and the left half of the data.
- A permutation is then performed that consists of the interchange of the two halves of the data.



# Feistel Cipher Structure

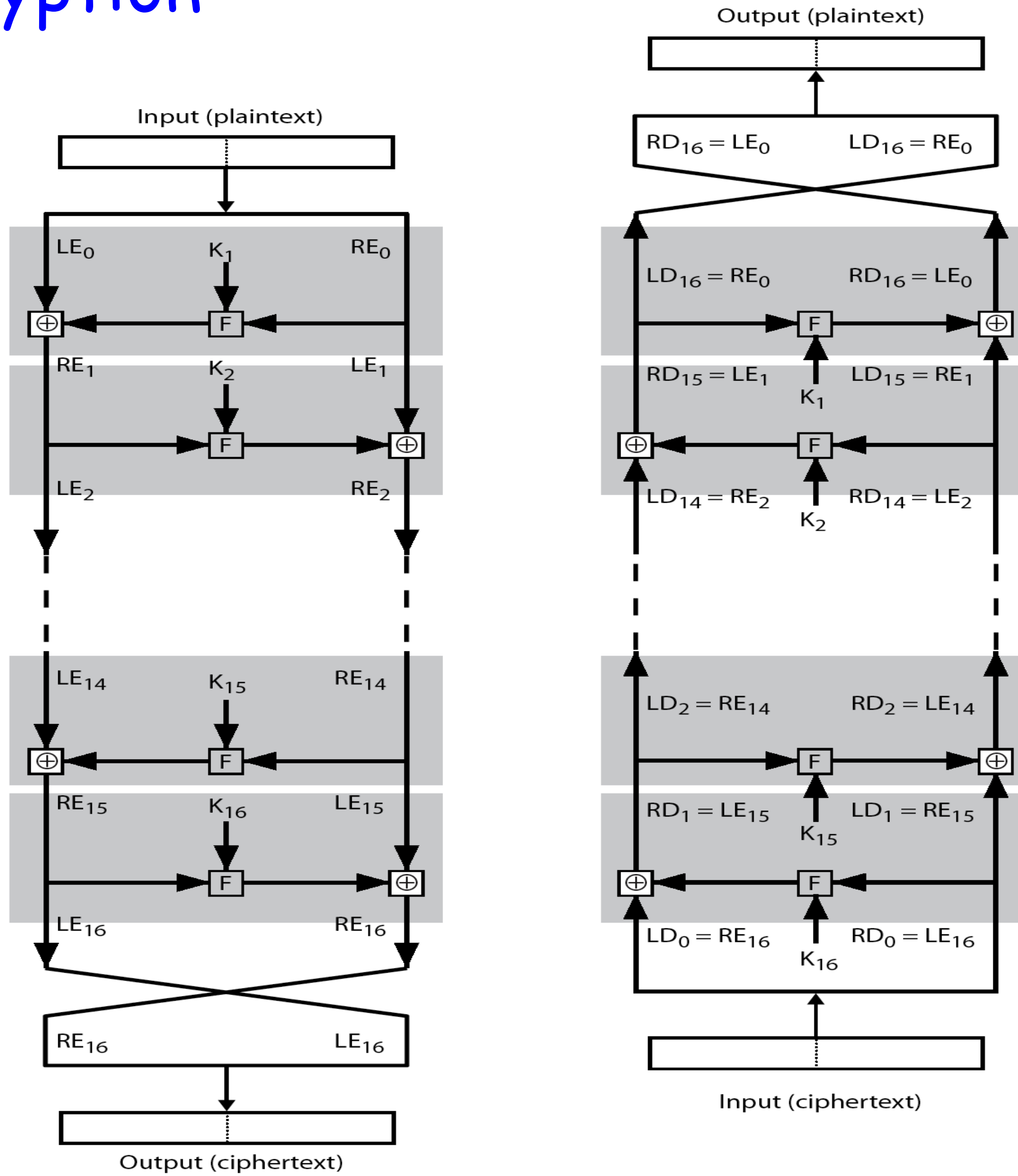
$$L_i = R_{i-1}$$

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



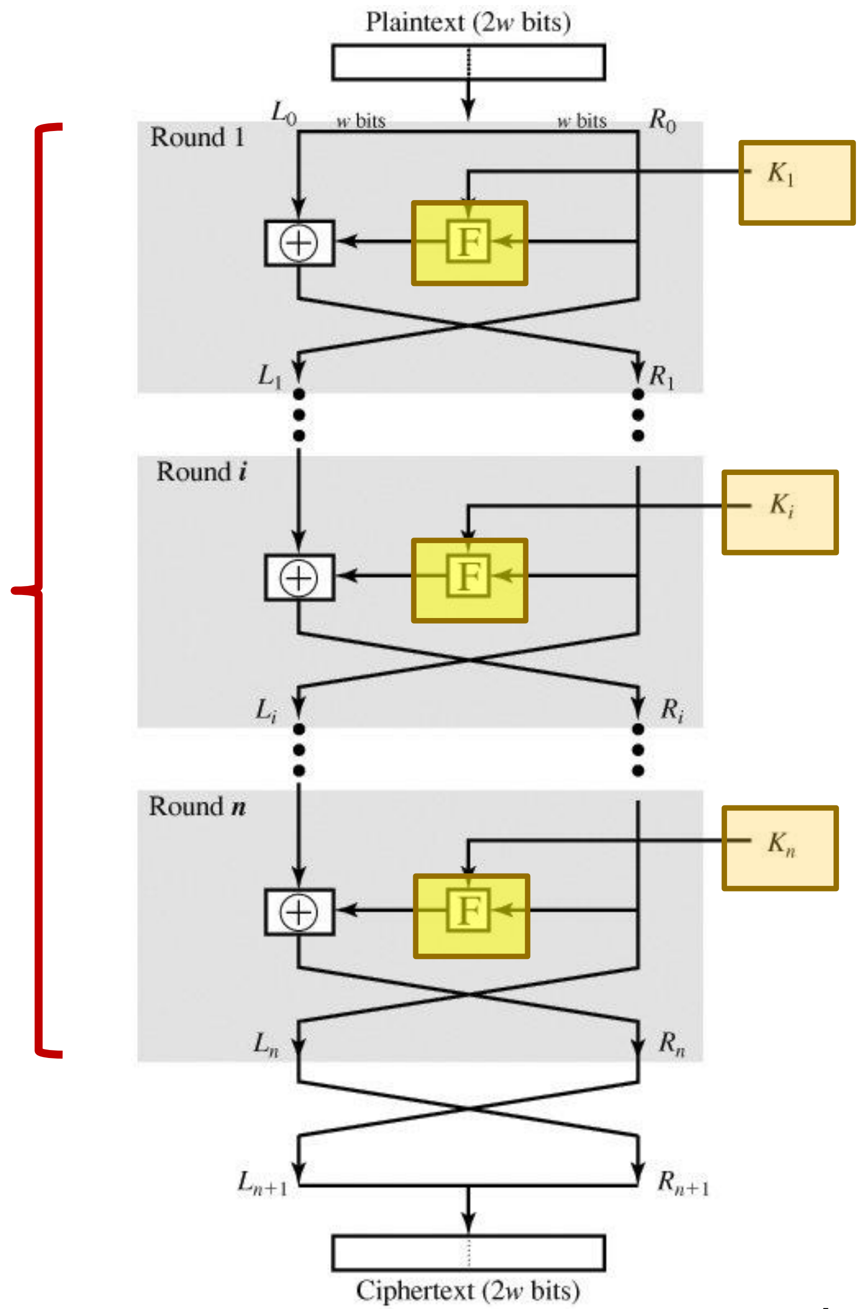
# Feistel Cipher Decryption

- Same as the encryption process
- Except that the subkeys $K_i$ are used in reverse order: $K_n$ in the first round,…, $k_1$ in the last round.
- Need not implement two different algorithms.



# Iterated Ciphers

- Feistal Cipher belongs to more general class of ciphers known as iterated ciphers that:
  - Process plaintext through multiple rounds
  - Apply the same round function at every round
  - At each round use a different sub-key derived from the main key



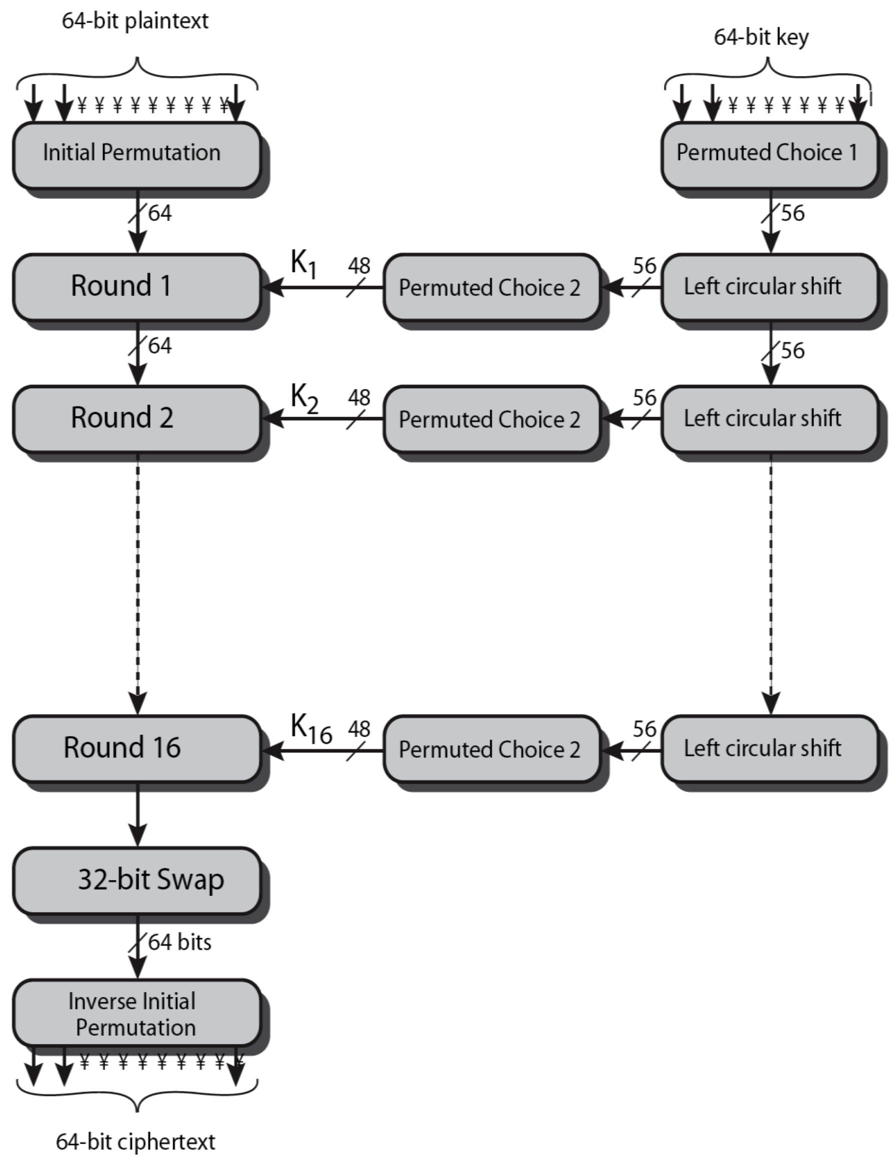
# Data Encryption Standard (DES)

# Data Encryption Standard (DES)

- A widely used block cipher
- Based on the Feistel cipher
- Developed in 1974 by IBM and the U.S. government to set a standard that everyone could use to securely communicate with each other
- 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.
- Use of DES has flourished, especially in financial applications

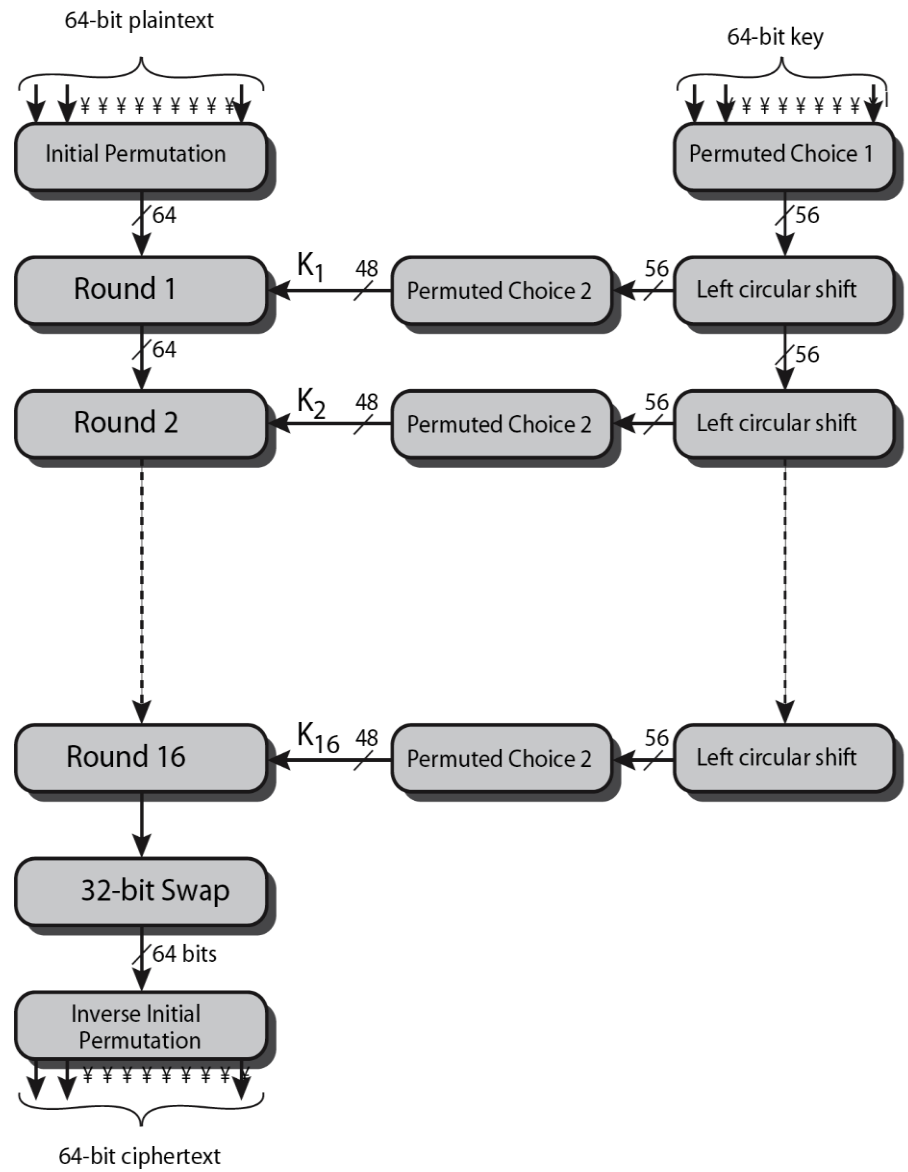
# DES Encryption Overview

- The plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input
- Then passes through 16 rounds of the same function, which involves both permutation and substitution function
- The left and right halves of the output of the last round are swapped, which is then passed through a permutation that is the inverse of IP to produce the 64-bit ciphertext
  - $IP^{-1}(IP(M)) = M$



# DES Encryption Overview

- The 64-bit key is passed through a permutation function.
- For each 16 round, a subkey $K_i$ is produced by the combination of a 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 shifts of the key bits



# Initial Permutation (IP)

Input:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ | $M_6$ | $M_7$ | $M_8$ |
| $M_9$ | $M_{10}$ | $M_{11}$ | $M_{12}$ | $M_{13}$ | $M_{14}$ | $M_{15}$ | $M_{16}$ |
| $M_{17}$ | $M_{18}$ | $M_{19}$ | $M_{20}$ | $M_{21}$ | $M_{22}$ | $M_{23}$ | $M_{24}$ |
| $M_{25}$ | $M_{26}$ | $M_{27}$ | $M_{28}$ | $M_{29}$ | $M_{30}$ | $M_{31}$ | $M_{32}$ |
| $M_{33}$ | $M_{34}$ | $M_{35}$ | $M_{34}$ | $M_{35}$ | $M_{36}$ | $M_{37}$ | $M_{38}$ |
| $M_{41}$ | $M_{42}$ | $M_{43}$ | $M_{44}$ | $M_{45}$ | $M_{46}$ | $M_{47}$ | $M_{48}$ |
| $M_{49}$ | $M_{50}$ | $M_{51}$ | $M_{52}$ | $M_{53}$ | $M_{54}$ | $M_{55}$ | $M_{56}$ |
| $M_{57}$ | $M_{58}$ | $M_{59}$ | $M_{60}$ | $M_{61}$ | $M_{62}$ | $M_{63}$ | $M_{64}$ |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

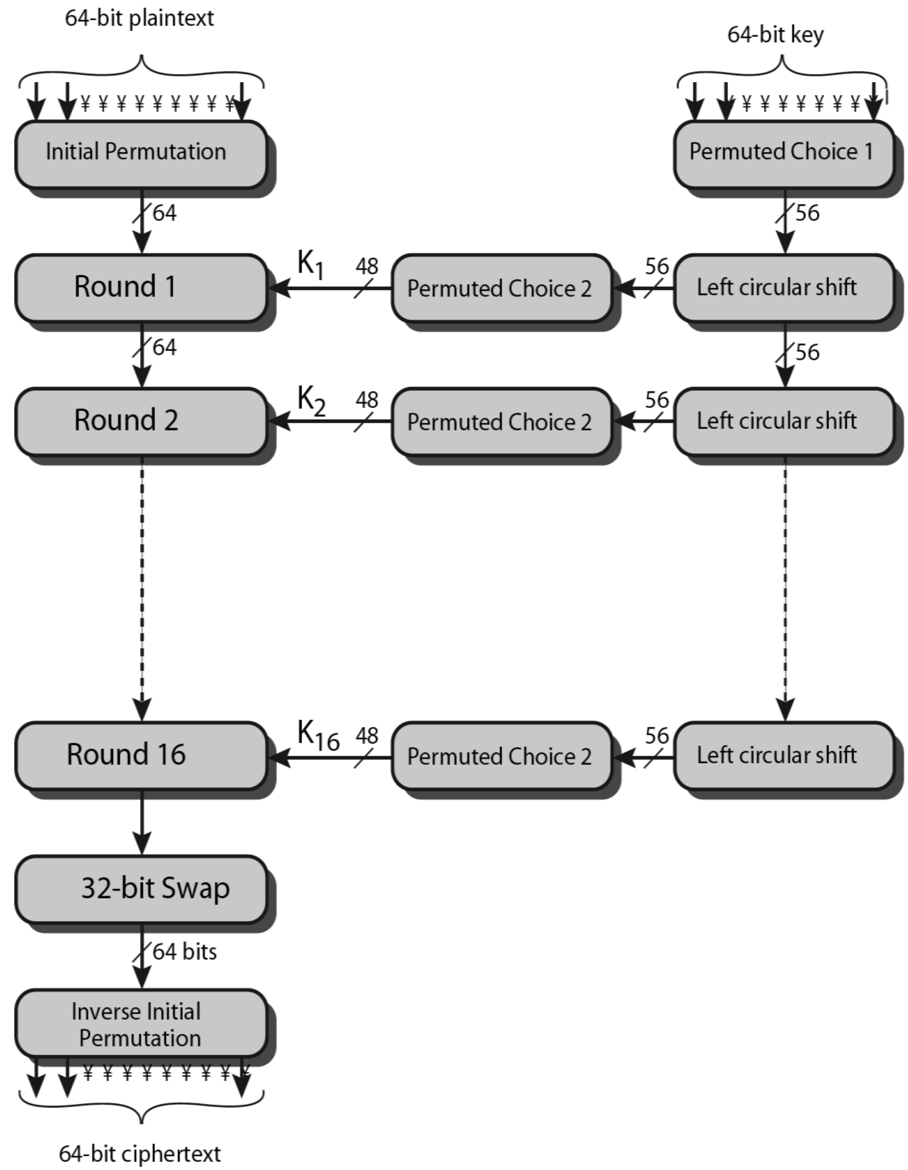
# Initial Permutation (IP)

Output:

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

# DES Encryption Overview

- The 64-bit key is passed through a permutation function.
- For each 16 round, a subkey $K_i$ is produced by the combination of a 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 shifts of the key bits

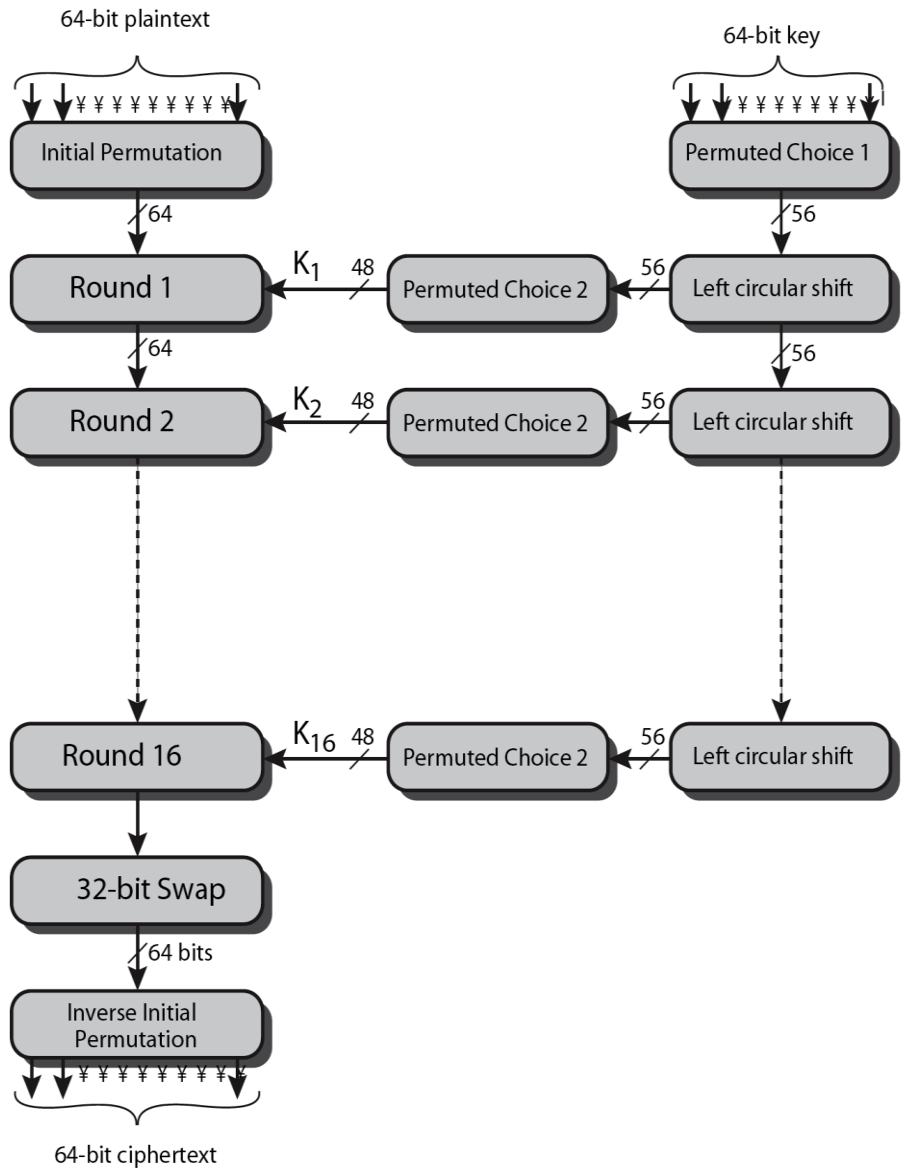


# Permuted Choice One (PC-1)

| | | | | | | |
|---|---|---|---|---|---|---|
| 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 |

# DES Encryption Overview

- The 64-bit key is passed through a permutation function.
- For each 16 round, a subkey $K_i$ is produced by the combination of a 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 shifts of the key bits



# Schedule of Left Circular Shifts

- The output of PC-1 is then treated as two 28 bits quantities, labeled $C_1$ and $D_1$.
- At each round, $C_i$ and $D_i$ are separately subjected to a circular left shift, of 1 or 2 bits.

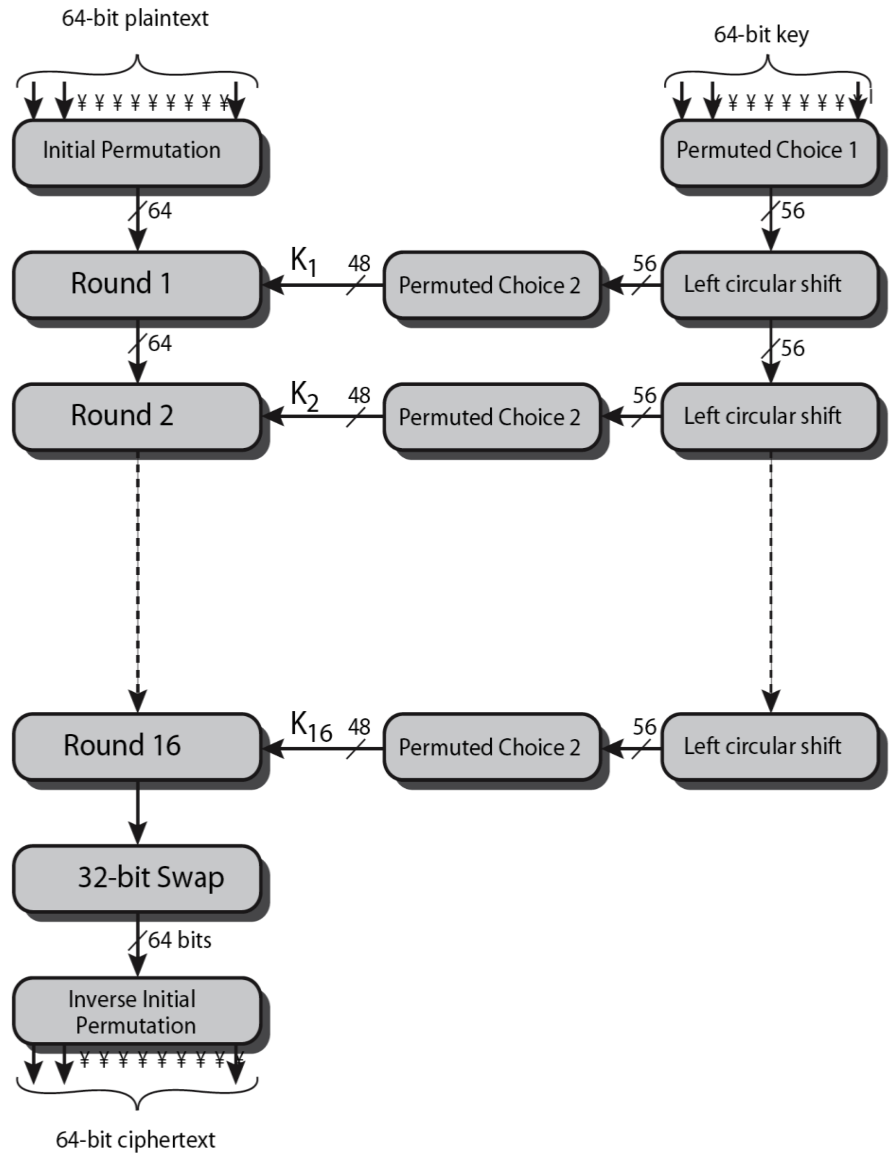
| Round number | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Bits rotated | 1 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 |

# 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 |

# DES Encryption Overview

- The plaintext passes through an initial permutation (IP) that rearranges the bits to produce the permuted input
- Then passes through 16 rounds of the same function, which involves both permutation and substitution function
- The left and right halves of the output of the last round are swapped, which is then passed through a permutation that is the inverse of IP to produce the 64-bit ciphertext
  - $IP^{-1}(IP(M)) = M$



# Inverse Initial Permutation ($IP^{-1}$)

Input:

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

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 |

# Inverse Initial Permutation ($IP^{-1}$)

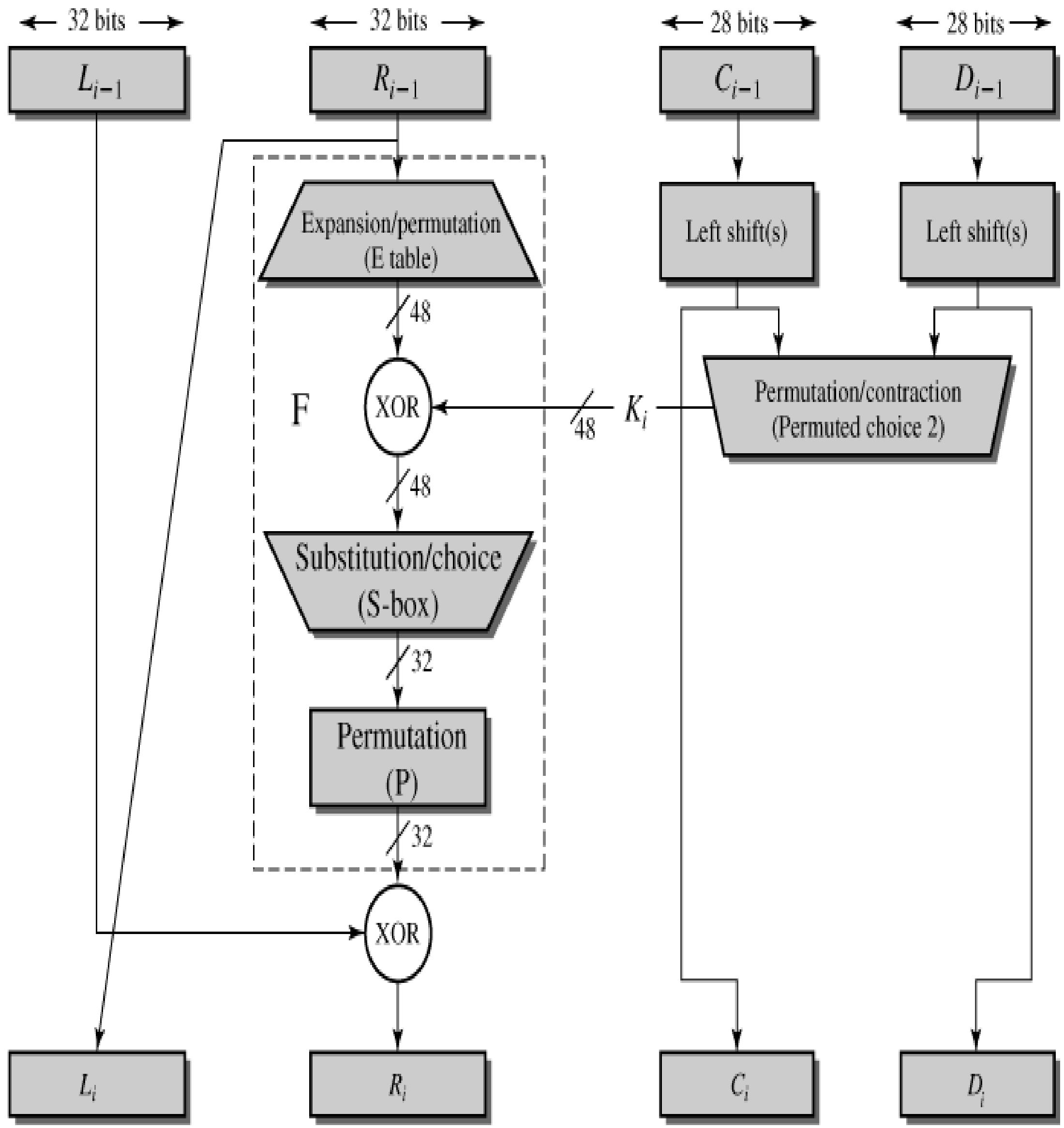
Output:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $M_1$ | $M_2$ | $M_3$ | $M_4$ | $M_5$ | $M_6$ | $M_7$ | $M_8$ |
| $M_9$ | $M_{10}$ | $M_{11}$ | $M_{12}$ | $M_{13}$ | $M_{14}$ | $M_{15}$ | $M_{16}$ |
| $M_{17}$ | $M_{18}$ | $M_{19}$ | $M_{20}$ | $M_{21}$ | $M_{22}$ | $M_{23}$ | $M_{24}$ |
| $M_{25}$ | $M_{26}$ | $M_{27}$ | $M_{28}$ | $M_{29}$ | $M_{30}$ | $M_{31}$ | $M_{32}$ |
| $M_{33}$ | $M_{34}$ | $M_{35}$ | $M_{34}$ | $M_{35}$ | $M_{36}$ | $M_{37}$ | $M_{38}$ |
| $M_{41}$ | $M_{42}$ | $M_{43}$ | $M_{44}$ | $M_{45}$ | $M_{46}$ | $M_{47}$ | $M_{48}$ |
| $M_{49}$ | $M_{50}$ | $M_{51}$ | $M_{52}$ | $M_{53}$ | $M_{54}$ | $M_{55}$ | $M_{56}$ |
| $M_{57}$ | $M_{58}$ | $M_{59}$ | $M_{60}$ | $M_{61}$ | $M_{62}$ | $M_{63}$ | $M_{64}$ |

# Single Round of DES Algorithm

- Uses two 32-bit L & R halves. Ki: 48 bits

1. The R input is expanded to 48 bits (permutation + duplication of 16 of R bits); the resulting 48 bits are XORed with Ki.



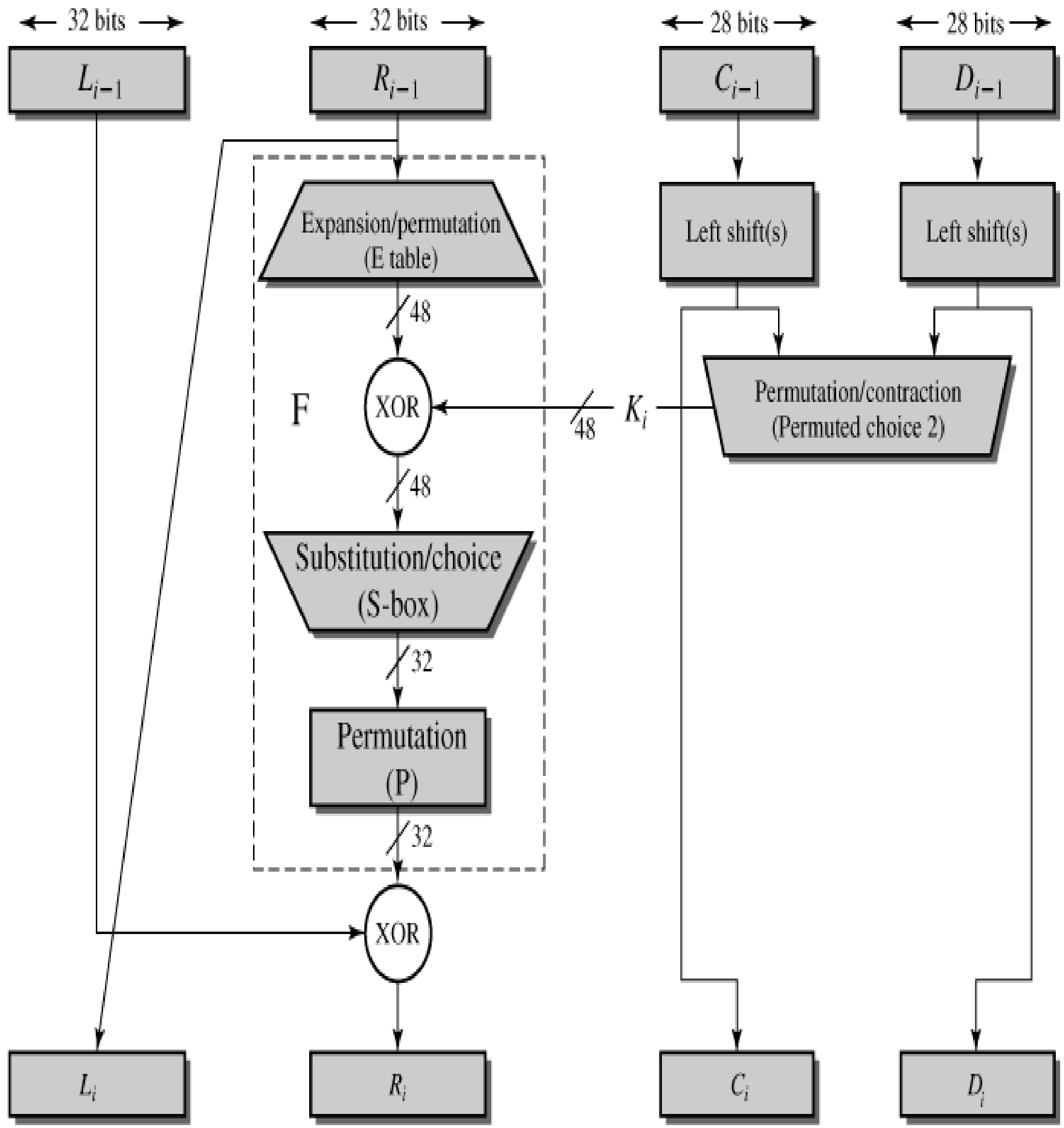
# Expanded Permutation (E)

| | | | | | |
|---|---|---|---|---|---|
| 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 |

# Single Round of DES Algorithm

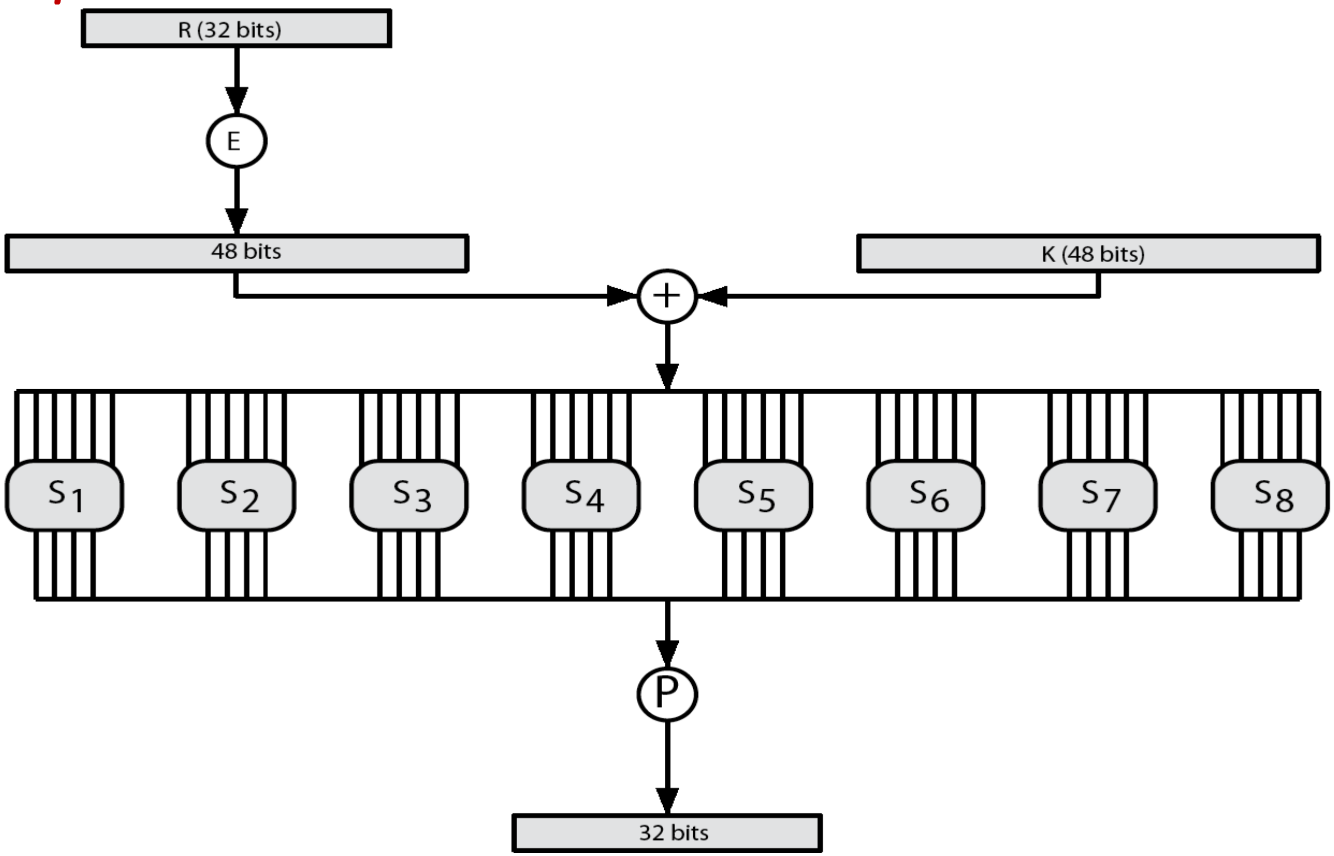
- Uses two 32-bit L & R halves. $K_i$: 48 bits

1. The R input is expanded to 48 bits (permutation + duplication of 16 of R bits); the resulting 48 bits are XORed with $K_i$.
2. The 48-bits then pass through a substitution function that produces 32-bit output.



# S-Boxes

- The substitution consists of a set of 8 S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.
- S-Box substitution allows DES to achieve the *confusion principle.*



# S-Boxes

- The substitution consists of a set of 8 S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.
  - The first and last bits of the input to $S_i$ form a 2-bit binary number to select one of 4 substitutions defined by the four rows (0, 1, 2, 3) in the table for $S_i$.
  - The middle 4 bits select one of 16 columns (0-15).
  - E.g. in $S_1$, input 011001

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

# S-Boxes

- The substitution consists of a set of 8 S-boxes, each of which accepts 6 bits as input and produces 4 bits as output.
  - The first and last bits of the input to $S_i$ form a 2-bit binary number to select one of 4 substitutions defined by the four rows (0, 1, 2, 3) in the table for $S_i$.
  - The middle 4 bits select one of 16 columns (0-15).
  - E.g. in $S_1$, input 011001
    - The row is 01 (row 1)
    - The column is 1100 (column 12)
    - The value is 9 – output is 1001.

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

# S-Box: Design Criteria (1)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis
  - Any change to the input to an S-box should result in random-looking changes to the output
  - 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

# S-Box: Design Criteria (1)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 1:** Any change to the input to an S-box should result in ***random-looking*** changes to the output:

**Example:** $\text{S-Box}_1(000000) = 14 = 1110$
$\text{S-Box}_1(000001) = 0 = 0000$
$\text{S-Box}_1(000010) = 4 = 0100$

(these changes look random…statistical tests exist to verify this)

# S-Box: Design Criteria (2)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 2:** No output bit of any S-box should be too close a ***linear function*** of the input bits.

Another words, it should be ***impossible to use a linear function*** in order to reliably predict any bit of the output based on the input bits.

# S-Box: Design Criteria (3)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 3:** Each row of an S-box should include all 16 possible output bit combinations.

**Example:** Notice each row contains all possible 4-bit values (0-15)

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

# S-Box: Design Criteria (4)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 4:** If two inputs to an S-Box differ in exactly one bit, the outputs must differ by at least two bits

**Example:** $\text{S-Box}_1(000000) = 14 = 1110$
$\text{S-Box}_1(000001) = 0 = 0000$

(The inputs differ by one bit. The outputs differ by 3 bits)

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

# S-Box: Design Criteria (5)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 5:** If two inputs to an S-Box differ in two middle bits, the outputs must differ by at least two bits

**Example:** $S\text{-}Box_1(000000) = 14 = 1110$
$S\text{-}Box_1(001100) = 11 = 1011$

(The inputs differ by 2 middle bits. The outputs differ by 2 bits)

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

# S-Box: Design Criteria (6)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 6:** If two inputs to an S-Box are identical in the first two bits and the last two bits, the resulting outputs must be different:

**Example:** $\text{S-Box}_1(000000) = 14 = 1110$

$\text{S-Box}_1(001100) = 11 = 1011$

The outputs are different!

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

# S-Box: Design Criteria (6)

- The design of the round function focuses on the design of S-boxes and on the permutation P.
- The design was primarily aimed at thwarting differential cryptanalysis

**Criteria 6:** If two inputs to an S-Box are identical in the first two bits and the last two bits, the resulting outputs must be different:

**Example:** $\text{S-Box}_1(000000) = 14 = 1110$
$\text{S-Box}_1(001100) = 11 = 1011$

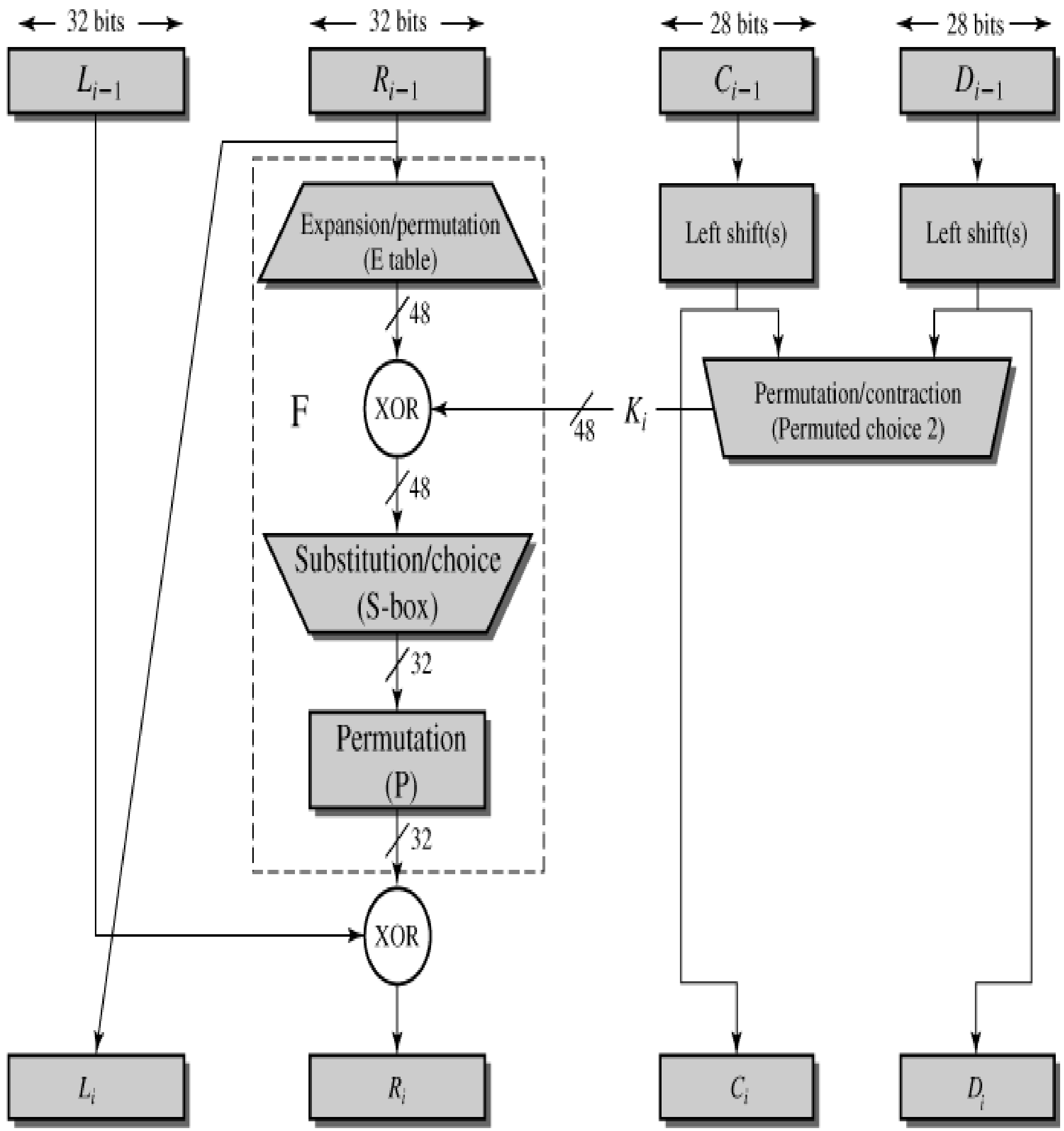
The outputs are different!

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

# Single Round of DES Algorithm

- Uses two 32-bit L & R halves. $K_i$: 48 bits

1. The R input is expanded to 48 bits (permutation + duplication of 16 of R bits); the resulting 48 bits are XORed with $K_i$.
2. The 48-bits then pass through a substitution function that produces 32-bit output
3. The 32-bit pass through a permutation function.



# 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 |

- Rearrangement of bits by the P-table allows DES to achieve the *diffusion principle*.
- Two output bits from each S-box affect middle bits of the next round and the other two affect end bits.

# 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 |

- The four output bits from each S-box affect six different S-boxes on the next round.

Expanded Permutation (E)

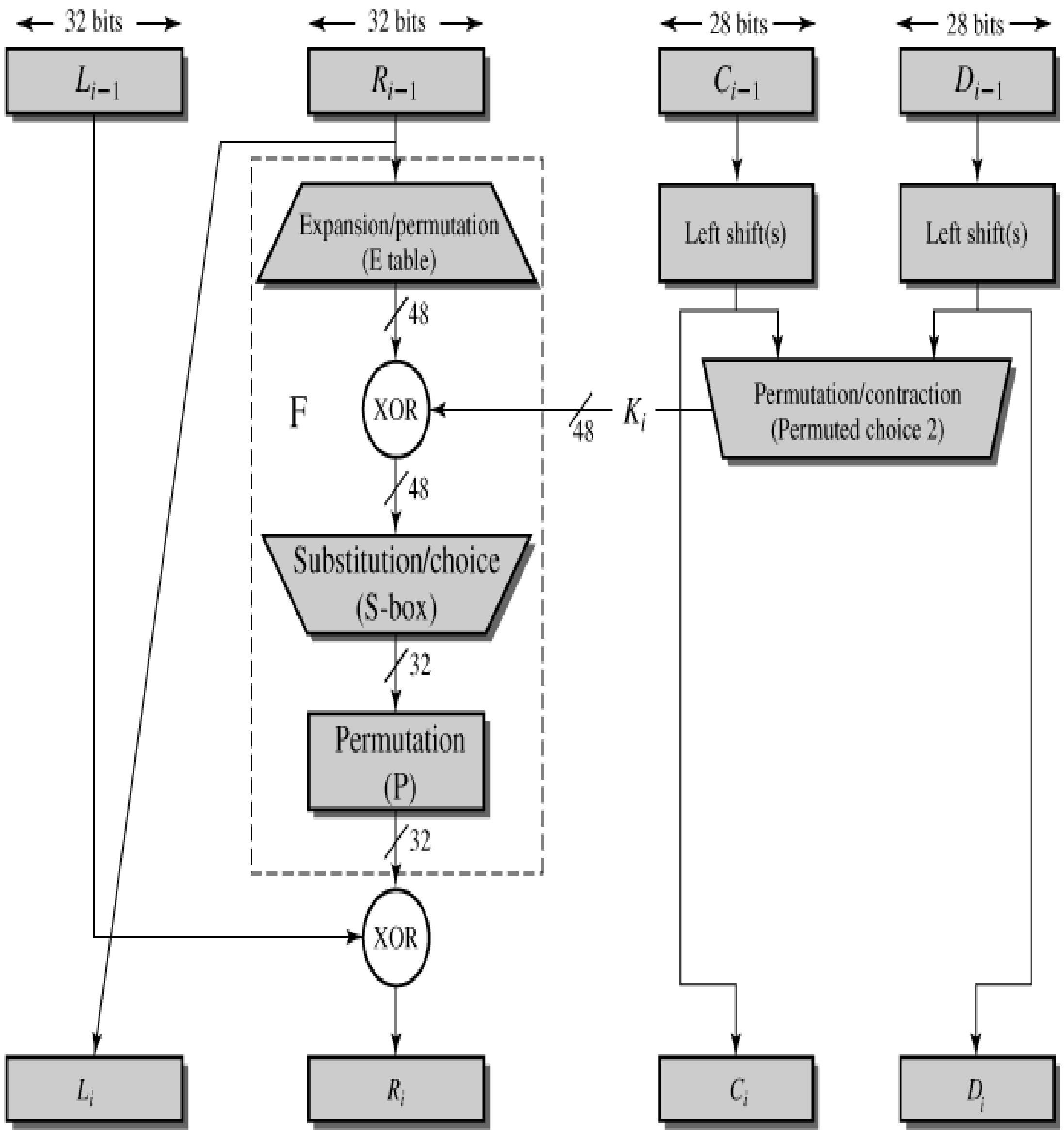
| | | | | | |
|---|---|---|---|---|---|
| 32 | 1 | 2 | 3 | 4 | 5 |
| 4 | 5 | 6 | 7 | 8 | 9 |
| 8 | 9 | 10 | 11 | 12 | 13 |
| 12 | 13 | 14 | 15 | 16 | 17 |
| 16 | 17 | 18 | 19 | 20 | 21 |
| 20 | 21 | 22 | 23 | 24 | 25 |
| 24 | 25 | 26 | 27 | 28 | 29 |
| 28 | 29 | 30 | 31 | 32 | 1 |

# Single Round of DES Algorithm

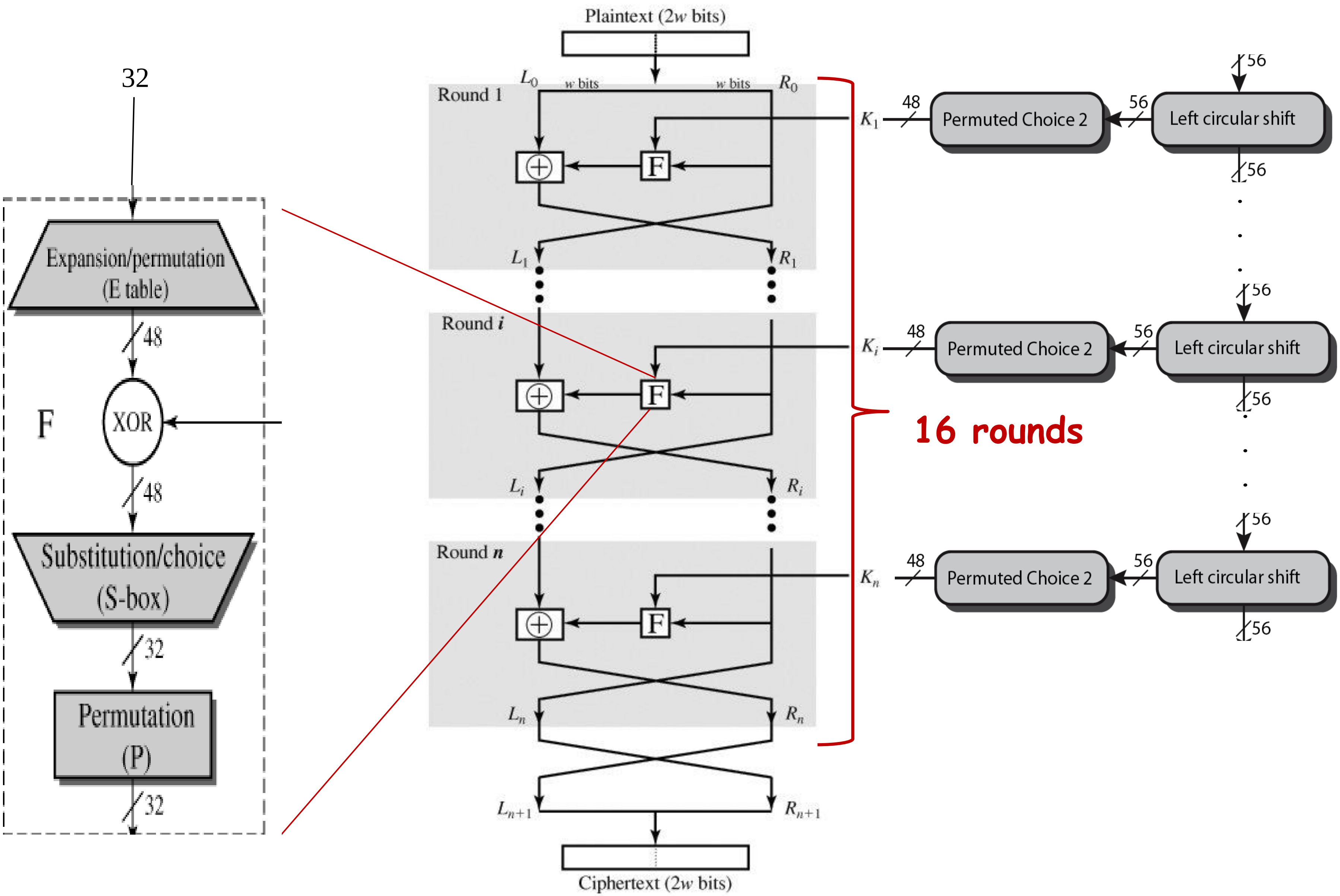
- As for any Feistel cipher,

  $L_i = R_{i-1}$

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



# A Recap: Remember: DES is a Feistel Cipher



# DES Decryption

- As with any Feistel cipher, decryption uses the same algorithm as encryption, except that the application of the subkeys is reversed.

# Avalanche Effect

- Avalanche Effect:
  - Desirable property of any encryption algorithm.
  - Small change in either the plaintext or the key should produce a significant change in the ciphertext.
  - Change in 1 bit of the plaintext or the key should produce a change in many bits of the ciphertext.
- Making attempts of guessing keys impossible
- DES exhibits strong avalanche

# Avalanche Effect

- Two plaintexts:
  - **P1:**

    **0**0000000 00000000
    00000000 00000000
    00000000 00000000
    00000000 00000000
  - **P2:**

    **1**0000000 00000000
    00000000 00000000
    00000000 00000000
    00000000 00000000
- Key K:
  - 0000001 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 |

# Avalanche Effect

- One plaintext P:
  - 01101000 10000101 00101111 01111010 00010011 01110110 11101011 10100100
- Two keys:
  - **K1**

    **1**110010 1111011 1101111 0011000 0011101 0000100 0110001 1101110
  - **K2**

    **0**110010 1111011 1101111 0011000 0011101 0000100 0110001 1101110

| (a) Change in Plaintext | | (b) Change in Key | |
|---|---|---|---|
| Round | Number of bits that differ | Round | Number of bits that differ |
| 0 | 1 | 0 | 0 |
| 1 | 6 | 1 | 2 |
| 2 | 21 | 2 | 14 |
| 3 | 35 | 3 | 28 |
| 4 | 39 | 4 | 32 |
| 5 | 34 | 5 | 30 |
| 6 | 32 | 6 | 32 |
| 7 | 31 | 7 | 35 |
| 8 | 29 | 8 | 34 |
| 9 | 42 | 9 | 40 |
| 10 | 44 | 10 | 38 |
| 11 | 32 | 11 | 31 |
| 12 | 30 | 12 | 33 |
| 13 | 30 | 13 | 28 |
| 14 | 26 | 14 | 26 |
| 15 | 29 | 15 | 34 |
| 16 | 34 | 16 | 35 |

# Strength of DES – Key Size

- 56-bit keys have $2^{56} = 7.2 \times 10^{16}$ values
- Brute force search looks hard
  - Assume: half of the key space has to be searched, a single machine performing one DES encryption per microsecond would take more than a thousand years to break the cipher
- Recent advances have shown is possible
  - In 1977, Diffie and Hellman: theorized technology existed to build a parallel machine with 1 million encryption devices; each performs one encryption per microsecond ➔ 10 hours
  - In 1998, Electronic Frontier Foundation (EFF) had broken a DES encryption using a computer ("Deep Crack") built for less than $250K. The attack took less than 3 days.

# Strength of DES – Key Size

- EFF "Deep Crack" Consisted of 1,186 custom chips designed to try different DES keys:

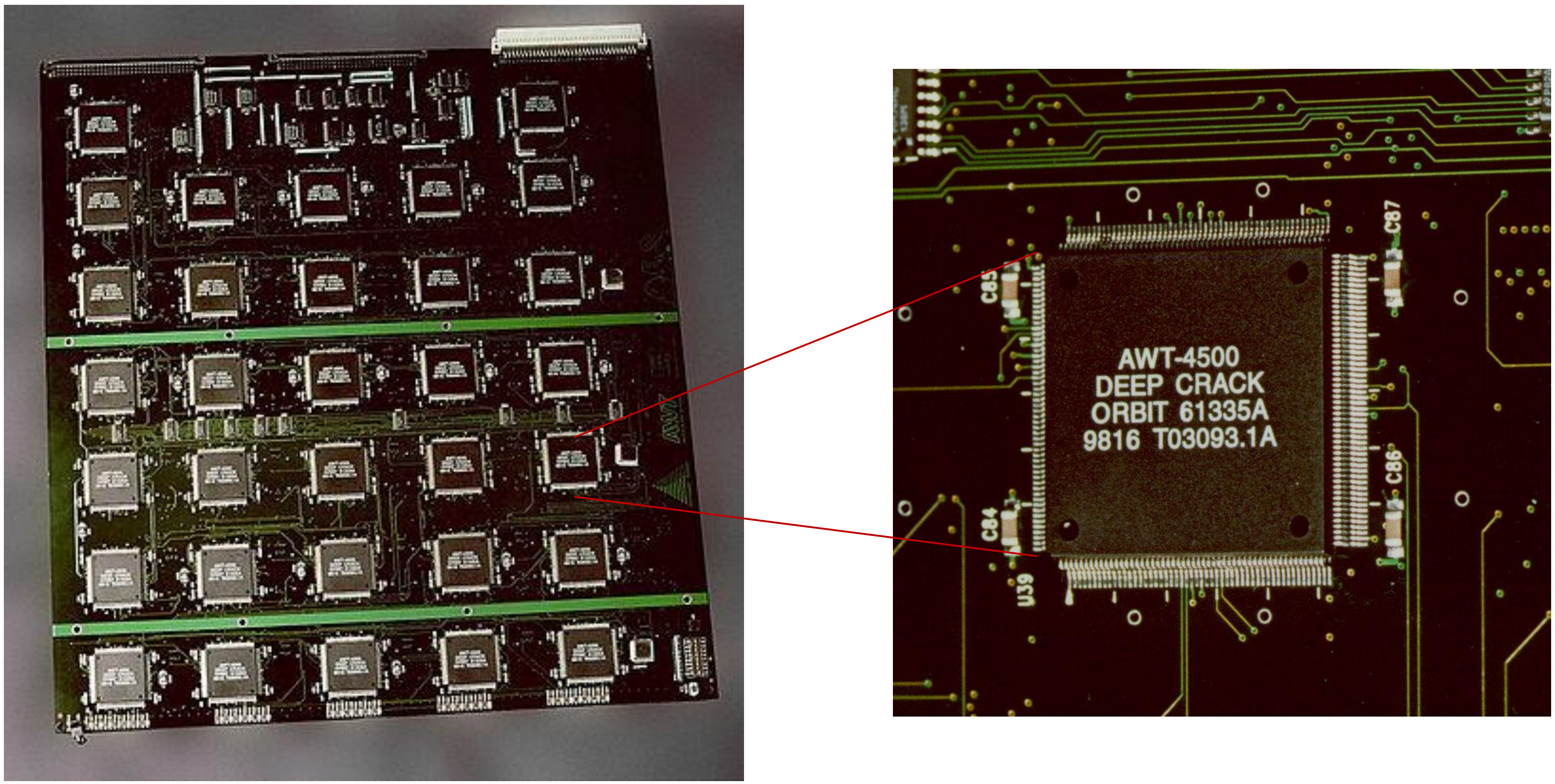



**Image source:** https://www.eff.org/

# Multiple Encryption & DES

- Problem: Clearly, a replacement for DES was needed
- Solution: Use multiple encryption with DES using multiple keys
  - Multiple encryption: a technique in which an encryption is used multiple times.
- Triple-DES is the chosen form
  - Use three stages of the DES algorithm
  - Use a total of two or three distinct keys

# Double-DES?

- Could use 2 DES encrypts on each block

$$C = E(K2, E(K1, P))$$

- Issue of reduction to single stage
  - Would it be possible to find a key K3 such that

$$E(K2, E(K1, P)) = E(K3, P)$$

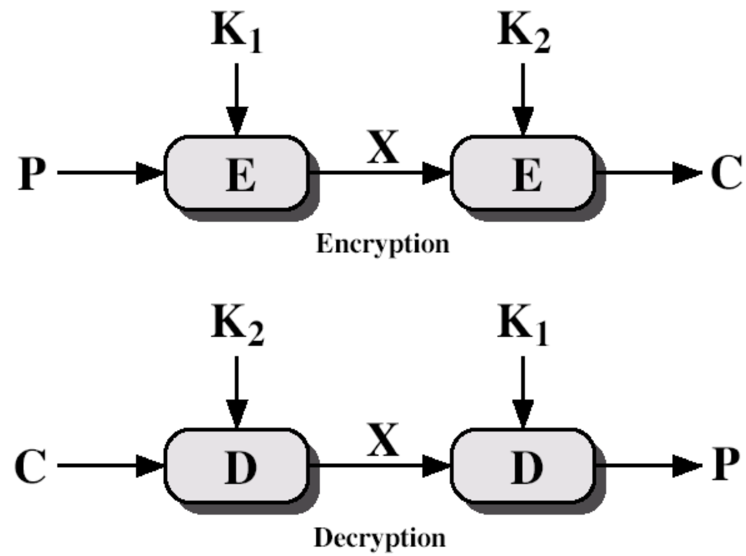
  - Answer: NO - proved in 1992

# Double-DES?

- Meet-in-the-middle attack
  - Works whenever use a cipher twice
  - Based on the observation:

**If C = E(K2, E(K1, P)),**

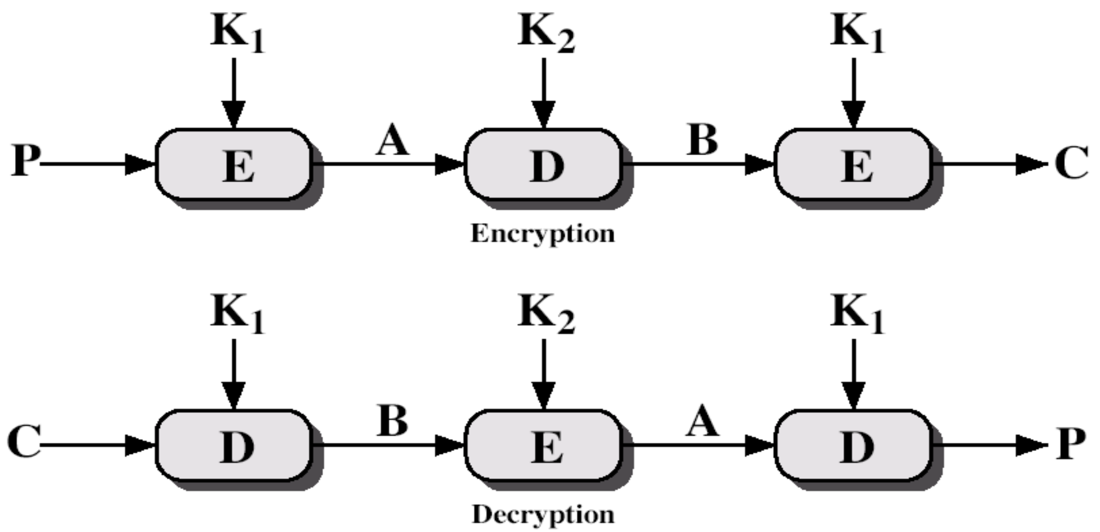
**then X = E(K1,P) = D(K2,C)**




- Encrypt P for all $2^{56}$ keys. Store the result in a table and then sort the table.
- Decrypt C with $2^{56}$ keys and check the result against the table for a match.
- If a match occurs, then test the two resulting keys against some more known plaintext-ciphertext pairs.

# Triple-DES with Two-Keys

- An obvious counter to the meet-in-the-middle attack is to use three stages of encryption with 3 different keys ➔ requiring key length 168-bits
  - Three times slower to run
- Can use 2 keys with E-D-E sequence
  - **$C = E(K1, D(K2, E(K1,P)))$**
  - The only advantage of the use of decryption for the second stage is: if **K1=K2** then can work with single DES



# Triple-DES with Three-Keys

- No current known practical attacks
  - Brute-force: $2^{112}$
  - Differential cryptanalysis: > $10^{52}$ plaintext-ciphertext pairs
- Can use Triple-DES with 3 Keys to avoid these
  - **C = E(K3, D(K2, E(K1, P)))**
- Has been adopted by some Internet applications, e.g., PGP, S/MIME

# Other Attacks Against DES and Block Ciphers: Differential Cryptoanalysis (1)

- A cryptoanalytic technique developed by Eli Biham and Adi Shamir in late 1980s
- They attempted its use to cryptoanalyze DES:
  - Discovered that ***DES S-Boxes were designed to be resilient to it!***
  - Evidence that NSA who co-developed DES in 1970s knew about the technique before they discovered it
    - Is this why S-Box design was kept secret?

# Other Attacks Against DES and Block Ciphers: Differential Cryptoanalysis (2)

- Differential Cryptoanalysis: attempts to derive the secret key by studying the how the input plaintext affects the ciphertext.
  - Usually involves a form of a chosen plaintext attack where the attacker can encrypt any plaintext of choice
- Differential cryptoanalysis of blockciphers:
  - Monitors the changes in plaintext as it is processed through multiple rounds of transformation
  - Looks for instances where the changes appear to be non-random
  - Uses non-random changes to extract the secret key

# Other Attacks Against DES and Block Ciphers: Linear Cryptoanalysis (2)

- First proposed by Matsuri in 1993 by Mitsuru Matsui and Atsuhiro Yamagishi for breaking a FEAL cipher
  - Newer than differential cryptoanalysis
- Was also applied to cryptoanalysis of DES:
  - Found that DES is more vulnerable to this technique than differential cryptoanalysis because S-Boxes were (probably) not designed with this attack in mind
  - However, a successful DES linear cryptoanalysis attack would require $2^{43}$ known plaintext/ciphertext pairs, which is ***impractical***
- Now a popular technique for attacking block ciphers

# Other Attacks Against DES and Block Ciphers: Linear Cryptoanalysis (2)

- A form of a known plaintext attack, where attacker has a plaintext and its corresponding ciphertext pair. Tries to find the key
- Basic Idea: Assumes a linear relationship between the plaintext, key, and the ciphertext and tries to derive the key based on it.
- Formally: If ciphertext = E(Key, plaintext) where E is the encryption function, linear cryptoanalysis tries to find a linear function E' that approximates E
- Defense: Design S-Boxes so that the relationship between ciphertext, key, and plaintext is non-linear

# DES Attacks Timeline

| Year | Proposed/ implemented DES Attack |
|---|---|
| 1977 | Diffie & Hellman, (under-)estimate the costs of a key search machine |
| 1990 | Biham & Shamir propose differential cryptanalysis ($2^{47}$ chosen ciphertexts) |
| 1993 | Mike Wiener proposes design of a very efficient key search machine: Average search requires 36h. Costs: $1.000.000 |
| 1993 | Matsui proposes linear cryptanalysis ($2^{43}$ chosen ciphertexts) |
| Jun. 1997 | DES Challenge I broken, 4.5 months of distributed search |
| Feb. 1998 | DES Challenge II--1 broken, 39 days (distributed search) |
| Jul. 1998 | DES Challenge II--2 broken, key search machine *Deep Crack* built by the Electronic Frontier Foundation (EFF): 1800 ASICs with 24 search engines each, Costs: $250 000, 15 days average search time (required 56h for the Challenge) |
| Jan. 1999 | DES Challenge III broken in 22 h 15 m (distributed search assisted by *Deep Crack*) |
| 2006-2008 | Reconfigurable key search machine *COPACOBANA* developed at the Universities in Bochum and Kiel (Germany), uses 120 FPGAs to break DES in 6.4 days (avg.) at a cost of $10 000. |

# Other Block Ciphers and DES Alternatives

- In January 1997 National Institute of Standards (NIST) announced a competition for a cipher to replace DES and become Advanced Encryption Standard (AES):

| Algorithm | Block Size (bits) | Key Lengths (bits) | Comments |
|---|---|---|---|
| Rijndael | 128 | 128/192/256 | The winner! Used as basis for AES |
| Mars | 128 | 128/192/256 | DES Replacement Finalist |
| RC6 | 128 | 128/192/256 | DES Replacement Finalist |
| Serpent | 128 | 128/192/256 | DES Replacement Finalist |
| Twofish | 128 | 128/192/256 | DES Replacement Finalist |
| IDEA | 64 | 128 | DES Replacement Finalist |

# TwoFish (1)

- TwoFish was one of the candidates to replace DES
- Supports keys of up to 256 bits
- Uses 128-bit blocks
- Performs 16 rounds of Feistal Cipher when using 256 bit keys

# TwoFish (2)

- Based on four S-Boxes
- Each S-Box outputs four bytes
- The S-Boxes are generated based on the key to achieve confusion
  - Different from DES where the S-Boxes are fixed
- Change in even 1 bit of the key results in very different S-Boxes

# TwoFish (3)

- Uses Maximum Distance Separable (MDS) matrix to create diffusion
  - The output of S-Boxes is multiplied by the MDS

# TwoFish (4)

- Where TwoFish is used:
  - KeePass password manager
  - File encryption
  - GSM mobile phones
  - Client-side encryption in cloud services
  - More: https://www.schneier.com/academic/twofish/products.html