Cryptography and Network Security Chapter 3

Fourth Edition

by William Stallings

Lecture slides by Lawrie Brown

Chapter 3 – Block Ciphers and the Data Encryption Standard

All the afternoon Mungo had been working on Stern's code, principally with the aid of the latest messages which he had copied down at the Nevin Square drop. Stern was very confident. He must be well aware London Central knew about that drop. It was obvious that they didn't care how often Mungo read their messages, so confident were they in the impenetrability of the code.

—***Talking to Strange Men,*** **Ruth Rendell**

Modern Block Ciphers

- now look at modern block ciphers
- one of the most widely used types of cryptographic algorithms
- provide secrecy /authentication services
- focus on DES (Data Encryption Standard)
- to illustrate block cipher design principles

Block vs Stream Ciphers

- block ciphers process messages in blocks, each of which is then en/decrypted
- like a substitution on very big characters
  - 64-bits or more
- stream ciphers process messages a bit or byte at a time when en/decrypting
- many current ciphers are block ciphers
- broader range of applications

Block Cipher Principles

- most symmetric block ciphers are based on a **Feistel Cipher Structure**
- needed since must be able to **decrypt** ciphertext to recover messages efficiently
- block ciphers look like an extremely large substitution
- would need table of 2^{64} entries for a 64-bit block
- instead create from smaller building blocks
- using idea of a product cipher

Ideal Block Cipher

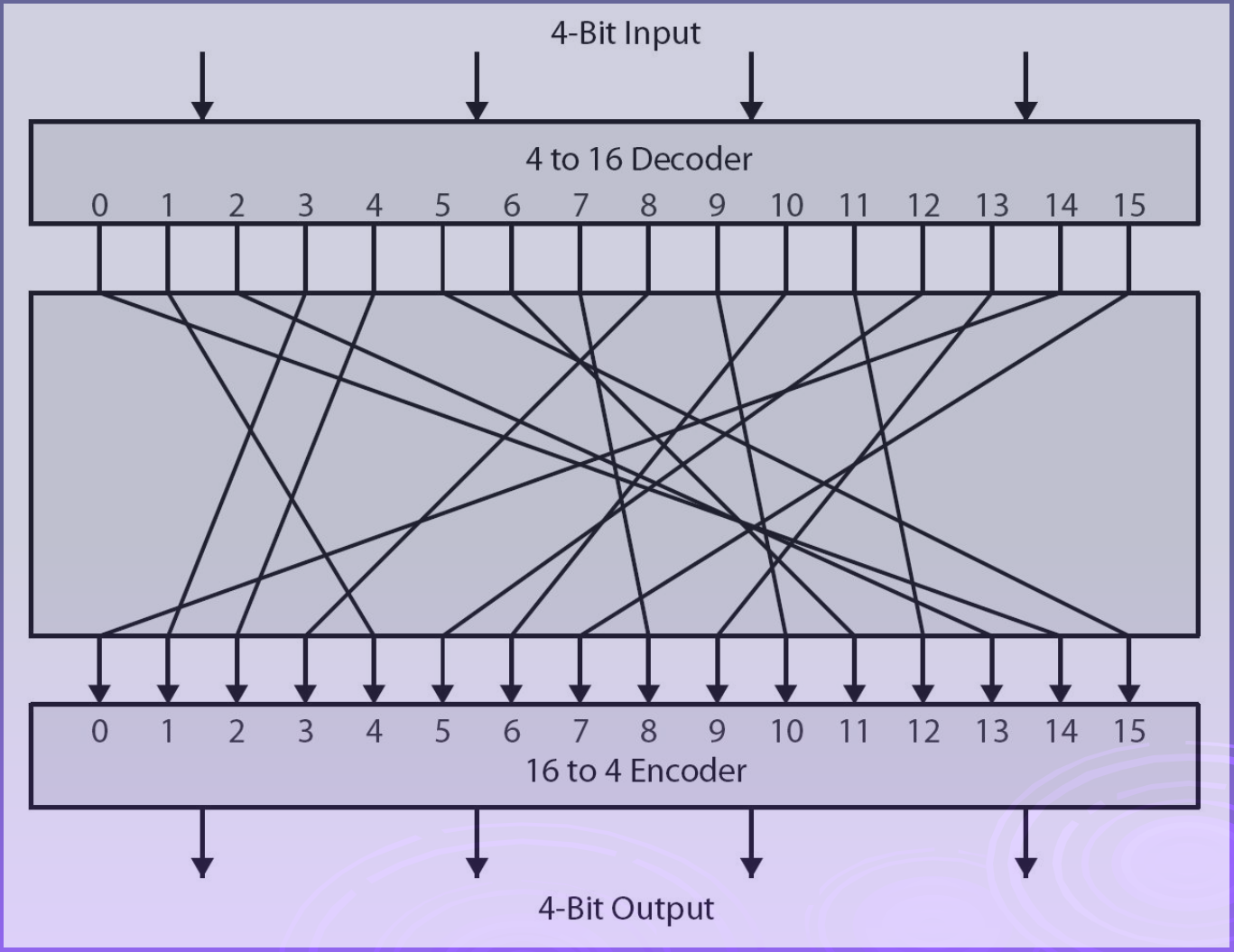


Table 3.1 Encryption and Decryption Tables for Substitution Cipher of Figure 3.4

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

Claude Shannon and Substitution-Permutation Ciphers

- Claude Shannon introduced idea of substitution-permutation (S-P) networks in 1949 paper
- form basis of modern block ciphers
- S-P nets are based on the two primitive cryptographic operations seen before:
  - *substitution* (S-box)
  - *permutation* (P-box)
- provide *confusion* & *diffusion* of message & key

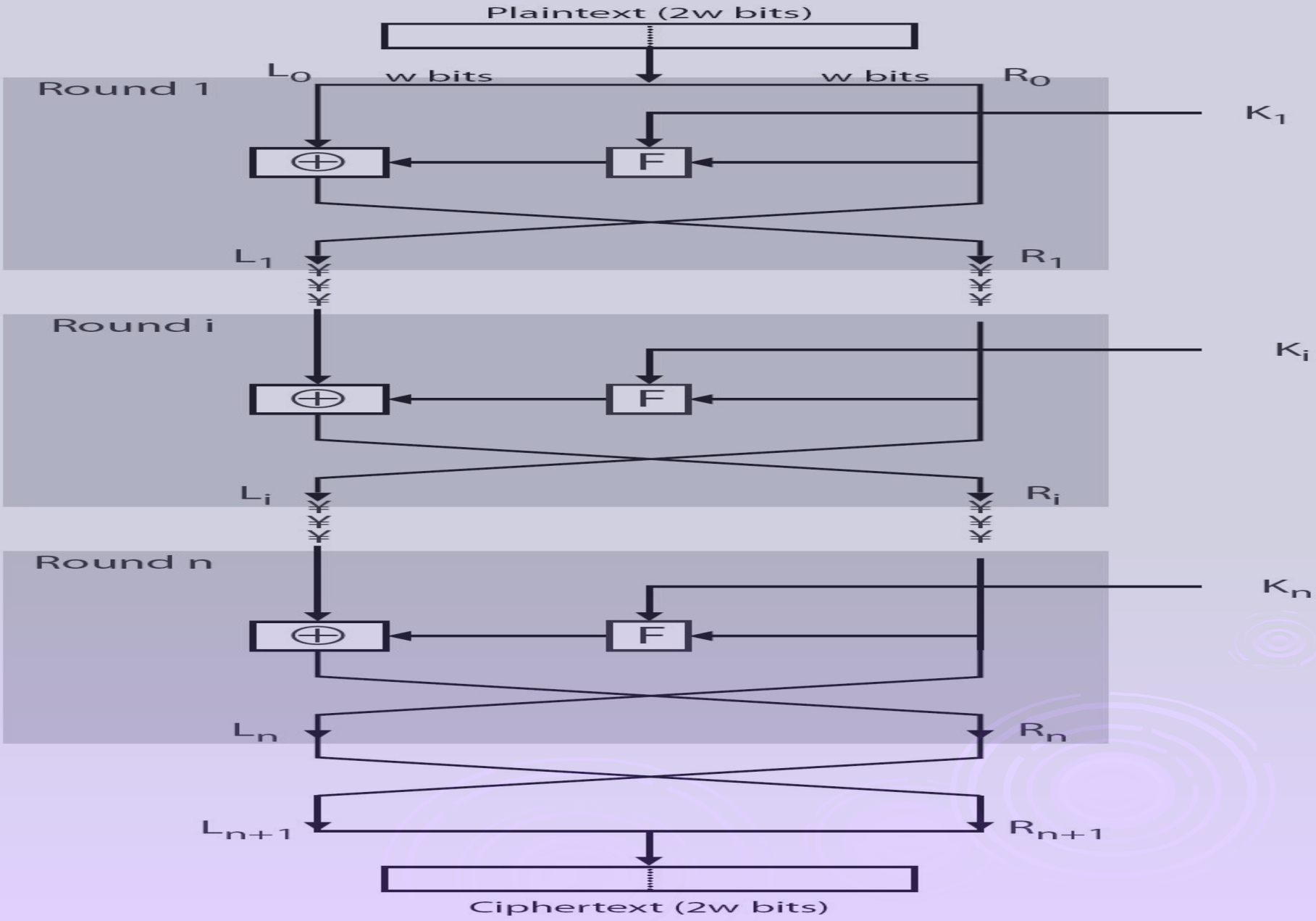
Confusion and Diffusion

- cipher needs to completely obscure statistical properties of original message
- a one-time pad does this
- more practically Shannon suggested combining S & P elements to obtain:
- **diffusion** – dissipates statistical structure of plaintext over bulk of ciphertext
- **confusion** – makes relationship between ciphertext and key as complex as possible

Feistel Cipher Structure

- Horst Feistel devised the **feistel cipher**
  - based on concept of invertible product cipher
- partitions input block into two halves
  - process through multiple rounds which
  - perform a substitution on left data half
  - based on round function of right half & subkey
  - then have permutation swapping halves
- implements Shannon's S-P net concept

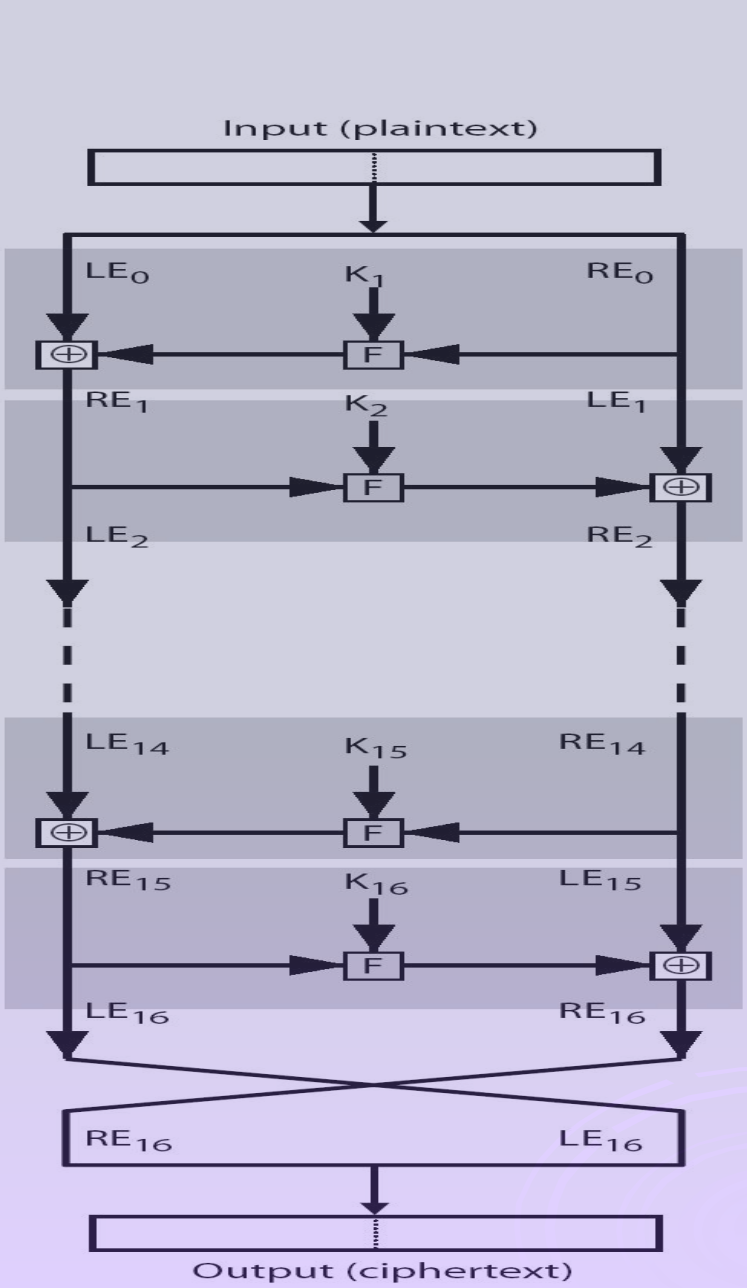
Feistel Cipher Structure

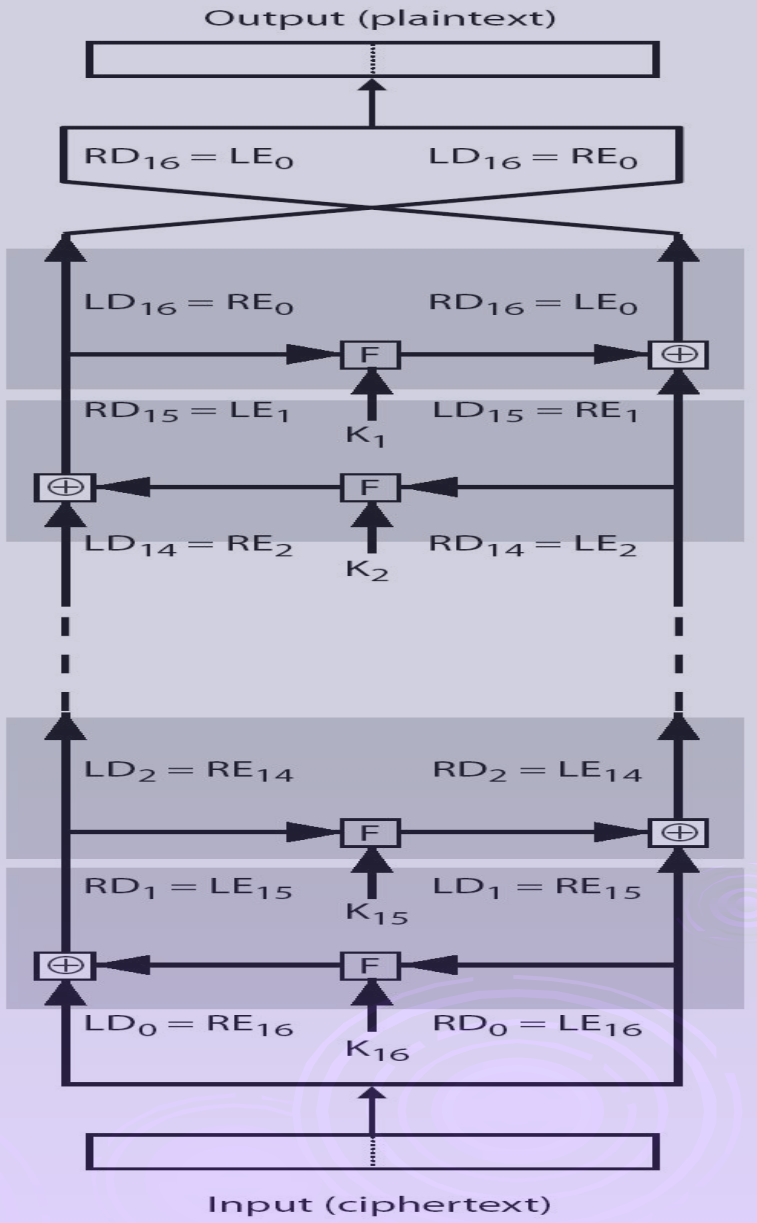


Feistel Cipher Design Elements

- block size
- key size
- number of rounds
- subkey generation algorithm
- round function
- fast software en/decryption
- ease of analysis

Feistel Cipher Decryption





Data Encryption Standard (DES)

- most widely used block cipher in world
- adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- has widespread use
- has been considerable controversy over its security

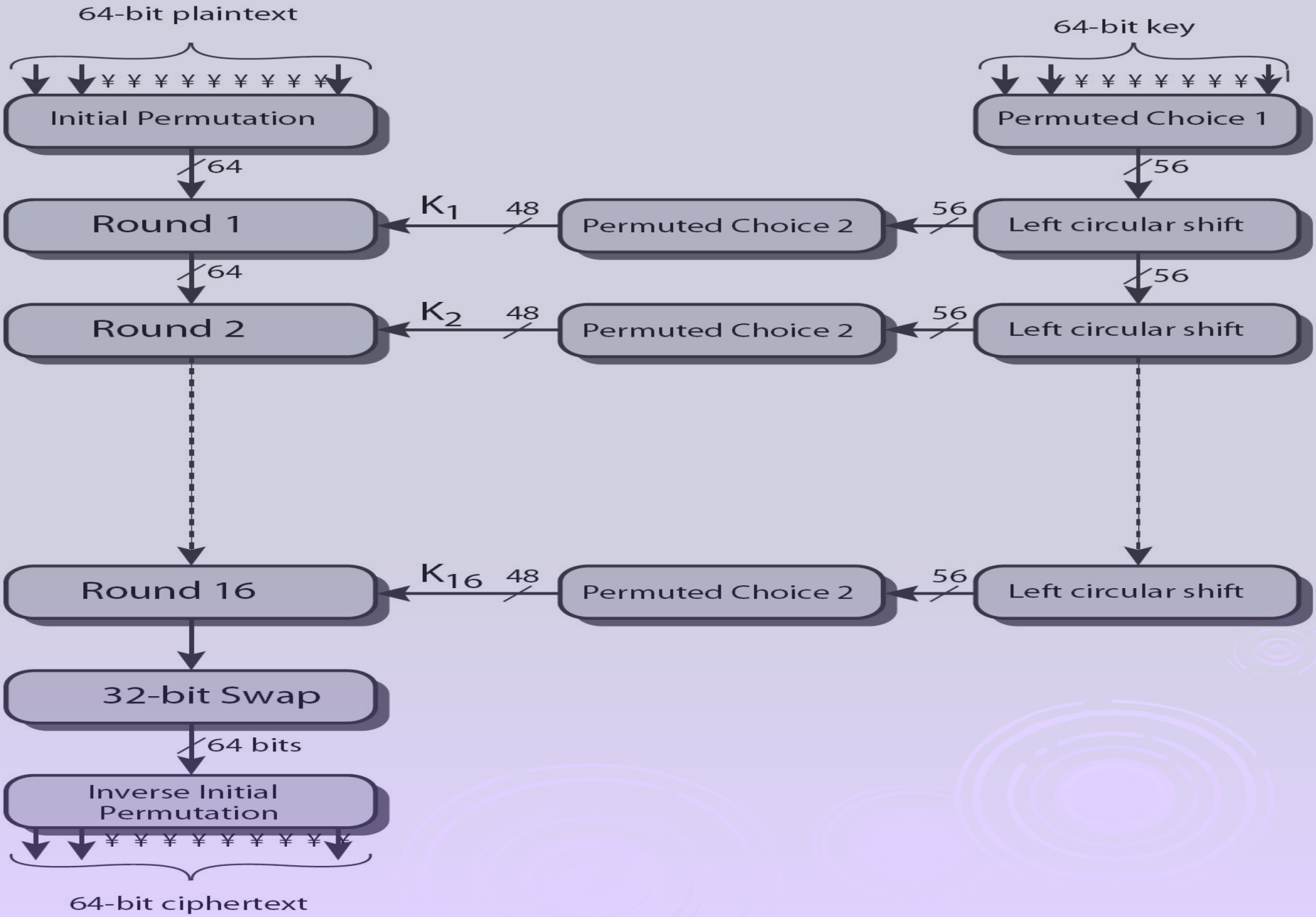
DES History

- IBM developed Lucifer cipher
  - by team led by Feistel in late 60's
  - used 64-bit data blocks with 128-bit key
- then redeveloped as a commercial cipher with input from NSA and others
- in 1973 NBS issued request for proposals for a national cipher standard
- IBM submitted their revised Lucifer which was eventually accepted as the DES

DES Design Controversy

- although DES standard is public
- was considerable controversy over design
  - in choice of 56-bit key (vs Lucifer 128-bit)
  - and because design criteria were classified
- subsequent events and public analysis show in fact design was appropriate
- use of DES has flourished
  - especially in financial applications
  - still standardised for legacy application use

DES Encryption Overview



DES Round Structure

- uses two 32-bit L & R halves
- as for any Feistel cipher can describe as:

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

- F takes 32-bit R half and 48-bit subkey:
  - expands R to 48-bits using perm E
  - adds to subkey using XOR
  - passes through 8 S-boxes to get 32-bit result
  - finally permutes using 32-bit perm P

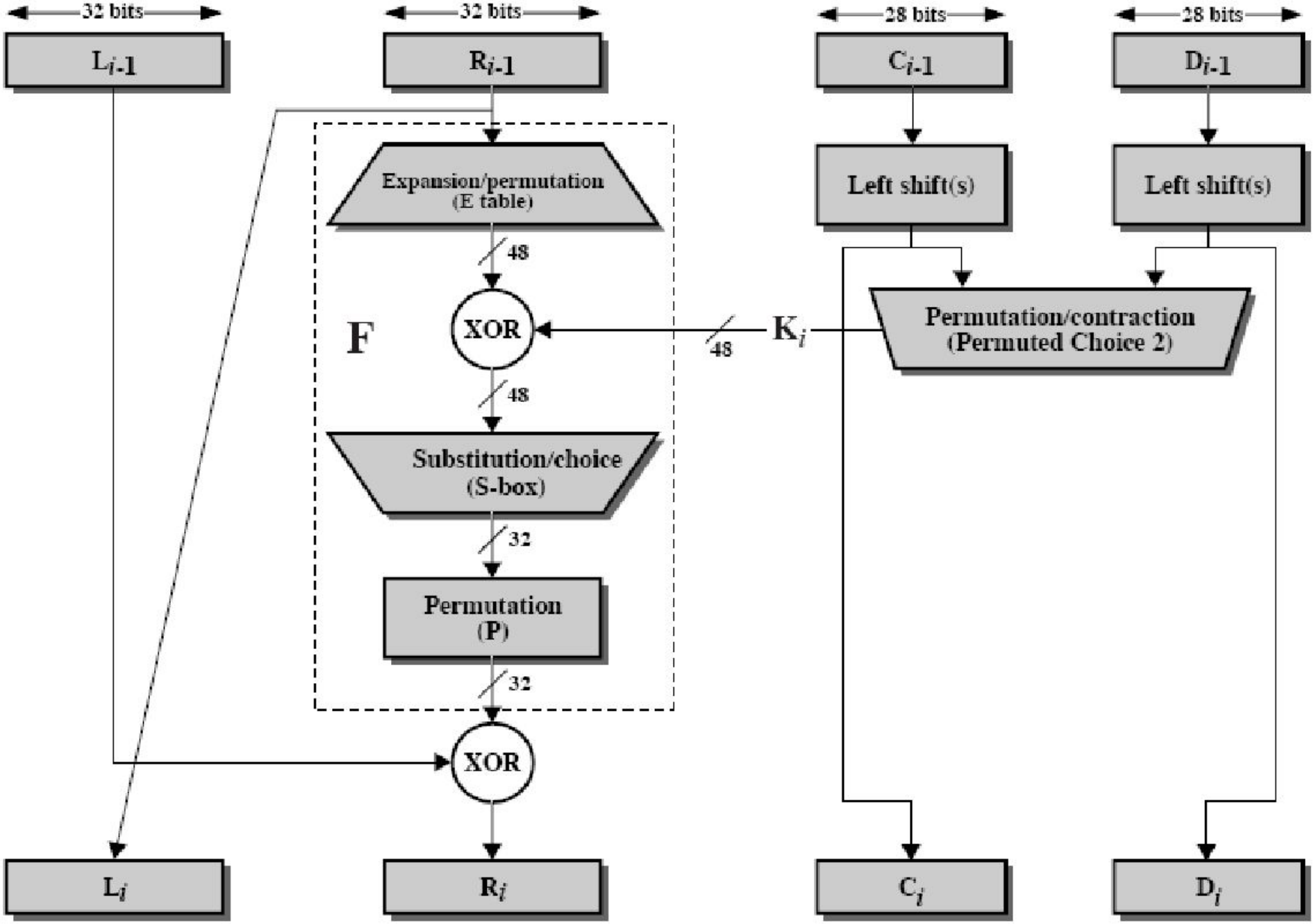


Figure 3.5 Single Round of DES Algorithm

Initial Permutation IP

- first step of the data computation
- IP reorders the input data bits
- even bits to LH half, odd bits to RH half
- quite regular in structure (easy in h/w)
- example:

```
IP(675a6967 5e5a6b5a) = (ffb2194d
004df6fb)
```

(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

(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

DES Round Structure

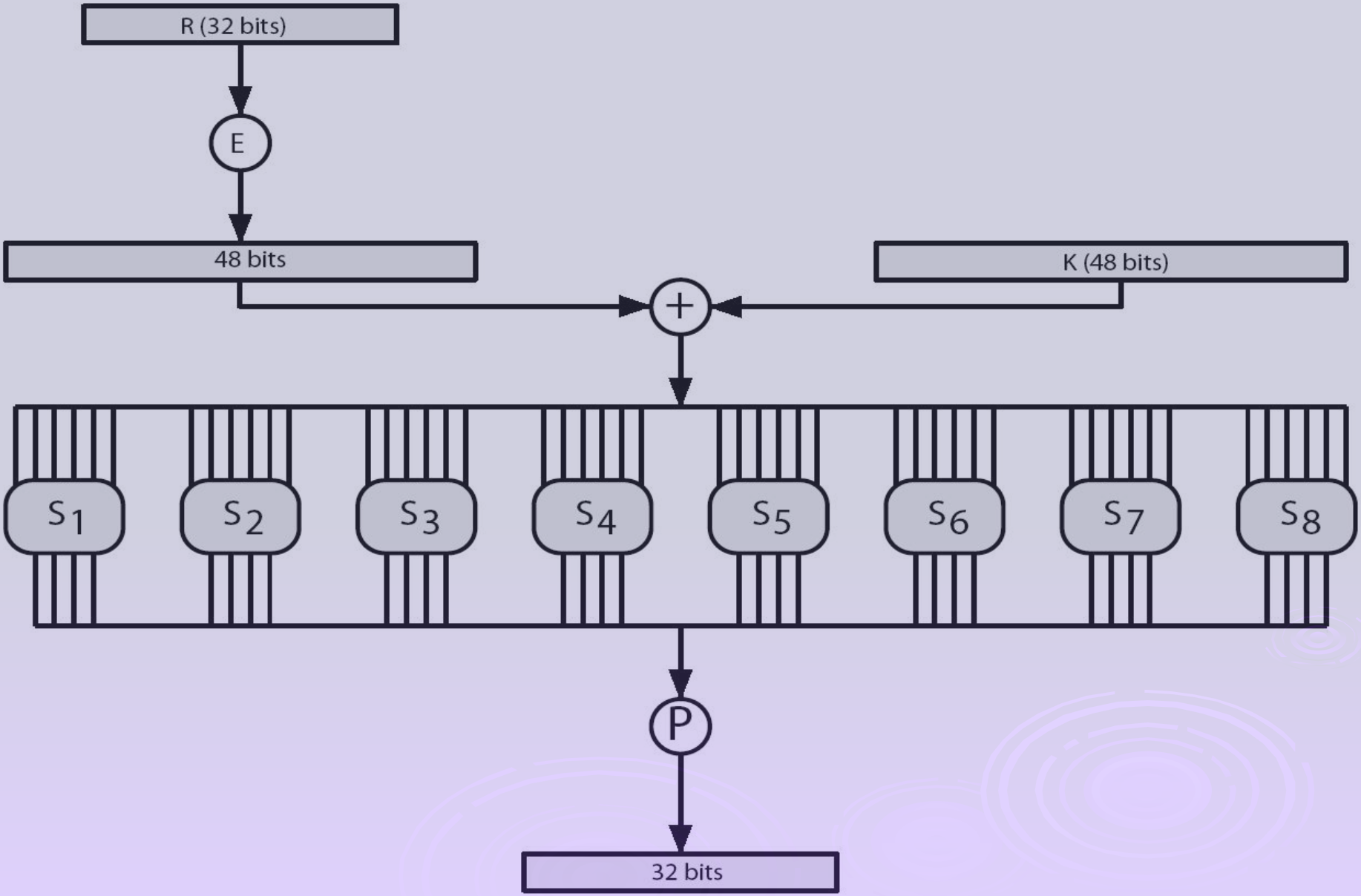


Table 3.3 Definition of DES S-Boxes

S_1	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

Table 3.3 Definition of DES S-Boxes

S_1	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_2	15	1	8	14	6	11	3	4	9	7	2	13	12	0	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
S_3	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	13	7	0	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
S_4	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	13	8	11	5	6	15	0	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	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5	2	12	4	1	7	10	11	6	8	5	3	15	13	0	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	0	14
	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	10	15	4	2	7	12	9	5	6	1	13	14	0	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

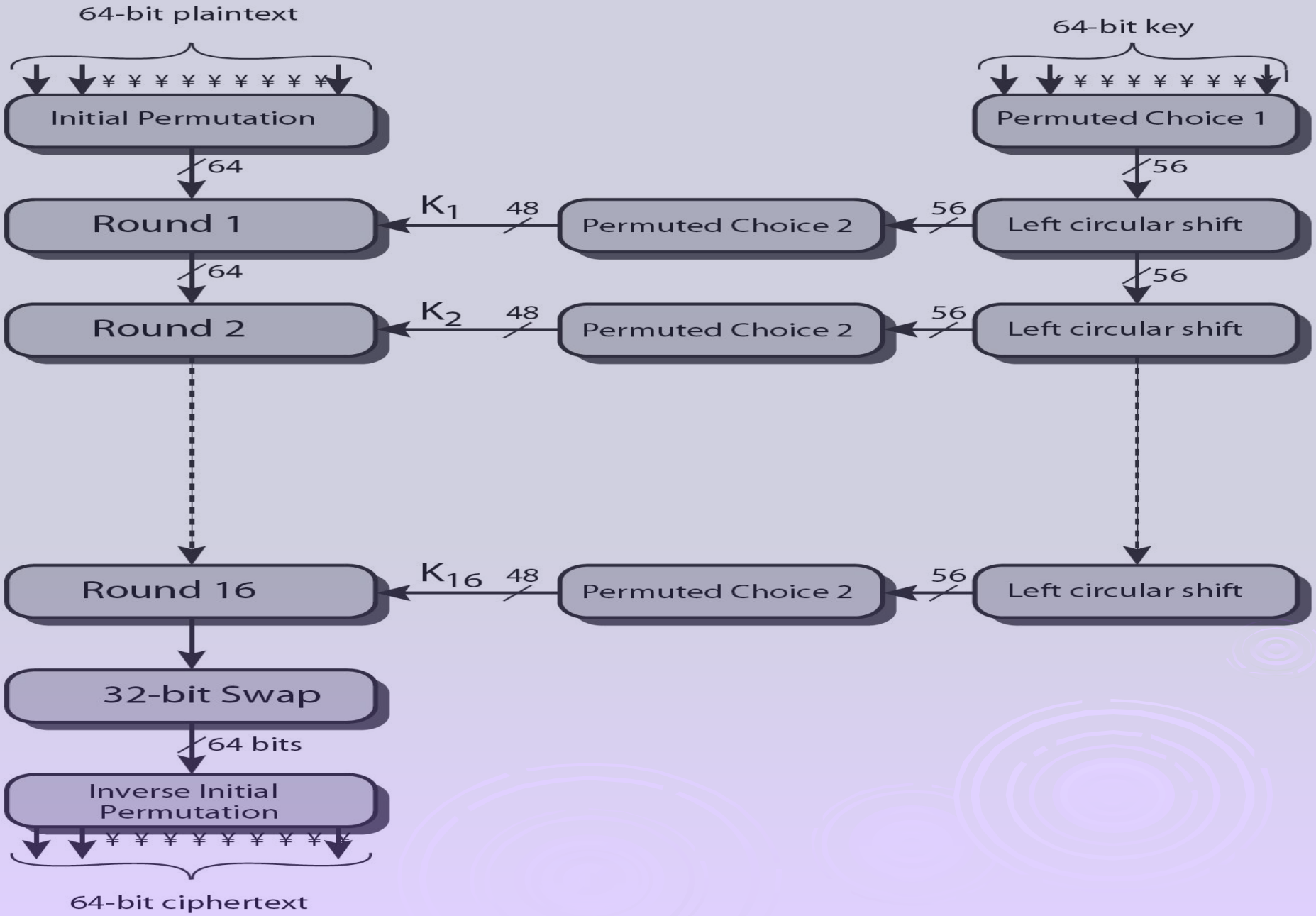
Substitution Boxes S

- have eight S-boxes which map 6 to 4 bits
- each S-box is actually 4 little 4 bit boxes
  - outer bits 1 & 6 (**row** bits) select one row of 4
  - inner bits 2-5 (**col** bits) are substituted
  - result is 8 lots of 4 bits, or 32 bits
- row selection depends on both data & key
  - feature known as autoclaving (autokeying)
- example:
  - `S(18 09 12 3d 11 17 38 39) = 5fd25e03`

DES Key Schedule

- forms subkeys used in each round
  - initial permutation of the key (PC1) which selects 56-bits in two 28-bit halves
  - 16 stages consisting of:
    - rotating **each half** separately either 1 or 2 places depending on the **key rotation schedule** K
    - selecting 24-bits from each half & permuting them by PC2 for use in round function F
- note practical use issues in h/w vs s/w

DES Encryption Overview



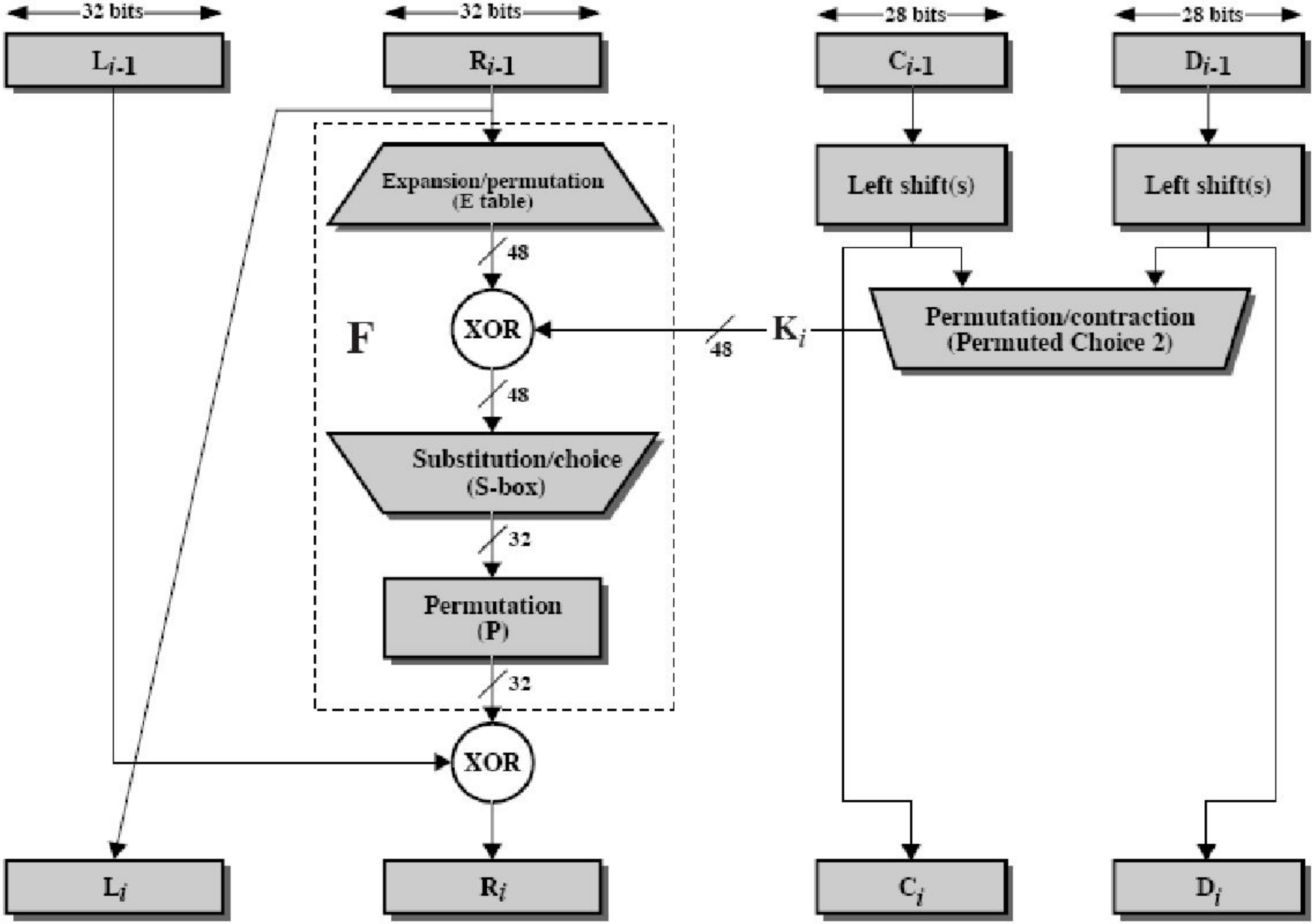


Figure 3.5 Single Round of DES Algorithm

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

(a) 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

(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

DES Decryption

- decrypt must unwind steps of data computation
- with Feistel design, do encryption steps again using subkeys in reverse order (SK16 … SK1)
  - 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

Avalanche Effect

- key desirable property of encryption alg
- where a change of **one** input or key bit results in changing approx **half** output bits
- making attempts to “home-in” by guessing keys impossible
- DES exhibits strong avalanche

Table 3.5 Avalanche Effect in DES

(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
- recent advances have shown is possible
  - in 1997 on Internet in a few months
  - in 1998 on dedicated h/w (EFF) in a few days
  - in 1999 above combined in 22hrs!
- still must be able to recognize plaintext
- must now consider alternatives to DES

Strength of DES – Analytic Attacks

- now have several analytic attacks on DES
- these utilise some deep structure of the cipher
  - by gathering information about encryptions
  - can eventually recover some/all of the sub-key bits
  - if necessary then exhaustively search for the rest
- generally these are statistical attacks
- include
  - differential cryptanalysis
  - linear cryptanalysis
  - related key attacks

Strength of DES – Timing Attacks

- attacks actual implementation of cipher
- use knowledge of consequences of implementation to derive information about some/all subkey bits
- specifically use fact that calculations can take varying times depending on the value of the inputs to it
- particularly problematic on smartcards

Differential Cryptanalysis

- one of the most significant recent (public) advances in cryptanalysis
- known by NSA in 70's cf DES design
- Murphy, Biham & Shamir published in 90’s
- powerful method to analyse block ciphers
- used to analyse most current block ciphers with varying degrees of success
- DES reasonably resistant to it, cf Lucifer

Differential Cryptanalysis

- a statistical attack against Feistel ciphers
- uses cipher structure not previously used
- design of S-P networks has output of function *f* influenced by both input & key
- hence cannot trace values back through cipher without knowing value of the key
- differential cryptanalysis compares two related pairs of encryptions

Differential Cryptanalysis Compares Pairs of Encryptions

- with a known difference in the input
- searching for a known difference in output
- when same subkeys are used

$$\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1}$$

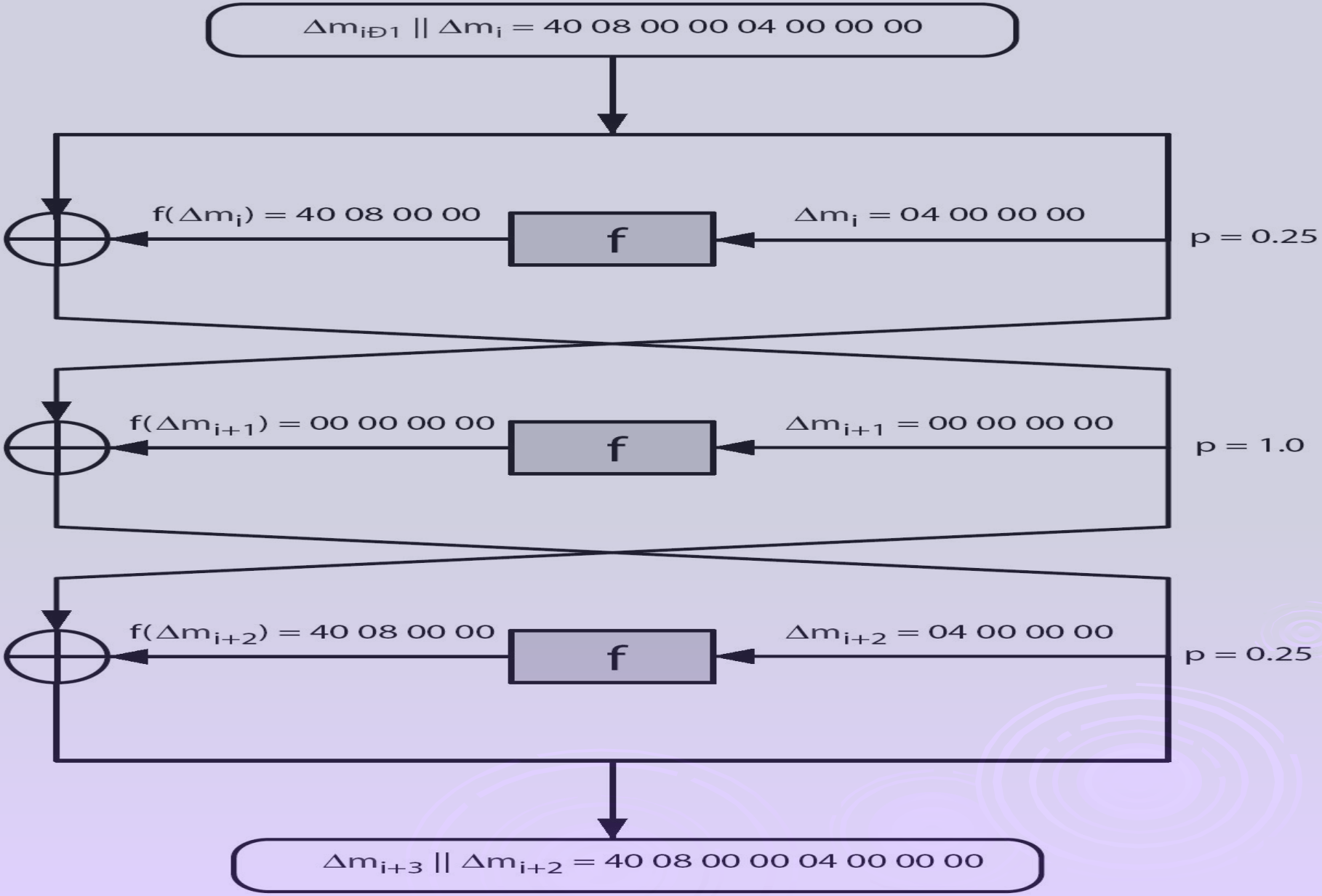
$$= [m_{i-1} \oplus \mathrm{f}(m_i, K_i)] \oplus [m'_{i-1} \oplus \mathrm{f}(m'_i, K_i)]$$

$$= \Delta m_{i-1} \oplus [\mathrm{f}(m_i, K_i) \oplus \mathrm{f}(m'_i, K_i)]$$

Differential Cryptanalysis

- have some input difference giving some output difference with probability p
- if find instances of some higher probability input / output difference pairs occurring
- can infer subkey that was used in round
- then must iterate process over many rounds (with decreasing probabilities)

Differential Cryptanalysis



Differential Cryptanalysis

- perform attack by repeatedly encrypting plaintext pairs with known input XOR until obtain desired output XOR
- when found
  - if intermediate rounds match required XOR have a **right pair**
  - if not then have a **wrong pair**, relative ratio is S/N for attack
- can then deduce keys values for the rounds
  - right pairs suggest same key bits
  - wrong pairs give random values
- for large numbers of rounds, probability is so low that more pairs are required than exist with 64-bit inputs
- Biham and Shamir have shown how a 13-round iterated characteristic can break the full 16-round DES

$S1$ Differential Distribution Table

Input x'	Output y'															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	64	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
01	0	0	0	6	0	2	4	4	0	10	12	4	10	6	2	4
02	0	0	0	8	0	4	4	4	0	6	8	6	12	6	4	2
03	14	4	2	2	10	6	4	2	6	4	4	0	2	2	2	0
⋮																
0C	0	0	0	8	0	6	6	0	0	6	6	4	6	6	14	2
⋮																
34	0	8	16	6	2	0	0	12	6	0	0	0	0	8	0	6
⋮																
3E	4	8	2	2	2	4	4	14	4	2	0	2	0	8	4	4
3F	4	4	4	2	4	0	2	4	4	2	4	8	8	6	2	2

Consider the input XOR 34. The possible output XORs are

Output:	1	2	3	4	7	8	D	F
Occurs:	8	16	6	2	12	6	8	6

$34 \longrightarrow 4$ has two occurrences. These input pairs are duals: (α, β) and (β, α)

When we construct the differential distribution table for $S1$, we discover these inputs as 13 and 27

$$\begin{aligned} 13 &= 01\ 0011 \\ 27 &= 10\ 0111 \\ 13 \oplus 27 &= 11\ 0100 \\ &= 34 \\ S1(13) &= 0110 \\ S1(27) &= 0010 \\ S1(13) \oplus S1(27) &= 0100 \\ &= 4 \end{aligned}$$

List of possible input values for $S1$ box with input XOR 34

34 → 1: 03, 0F, 1E, 1F, 2A, 2B, 37, 3B

34 → 2: 04, 05, 0E, 11, 12, 14, 1A, 1B, 20, 25, 26, 2E, 2F, 30, 31, 3A

34 → 3: 01, 02, 15, 21, 35, 36

34 → 4: 13, 27

34 → 7: 00, 08, 0D, 17, 18, 1D, 23, 29, 2C, 34, 39, 3C

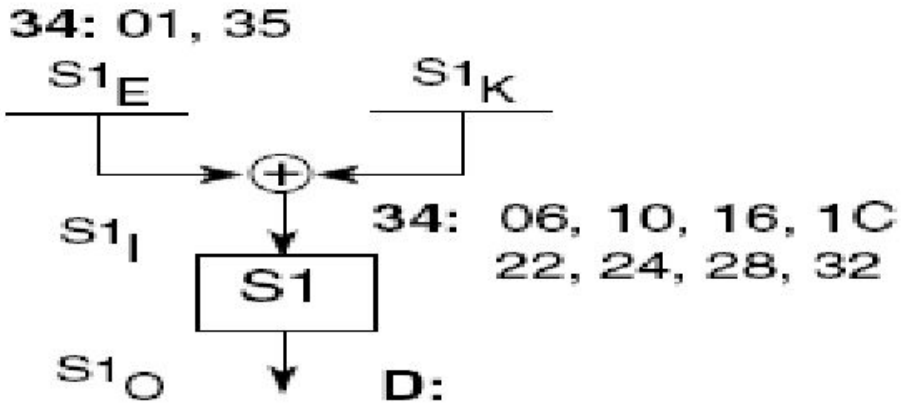
34 → 8: 09, 0C, 19, 2D, 38, 3D

34 → D: 06, 10, 16, 1C, 22, 24, 28, 32

34 → F: 07, 0A, 0B, 33, 3E, 3F

Determination of the key:

Suppose we know two inputs to $S1$ as 01 and 35 which XORs to 34, and the output XOR as D



The input XOR is 34, regardless of the value of the key because

$$\begin{aligned} S1'_I &= S1_I \oplus S1^*_I \\ &= (S1_E \oplus S1_K) \oplus (S1^*_E \oplus S1_K) \\ &= S1_E \oplus S1^*_E \\ &= S1'_E \end{aligned}$$

Also since

$$S1_I = S1_E \oplus S1_K$$

we have

$$S1_K = S1_I \oplus S1_E$$

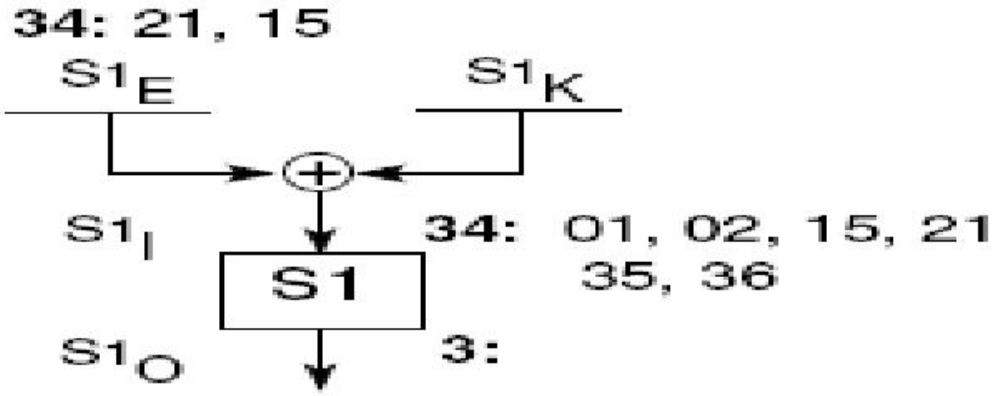
which gives

$$\begin{array}{lcl} 06 \oplus 01 & = & 07 \\ 10 \oplus 01 & = & 11 \\ 16 \oplus 01 & = & 17 \\ 1C \oplus 01 & = & 1D \\ 22 \oplus 01 & = & 23 \\ 24 \oplus 01 & = & 25 \\ 28 \oplus 01 & = & 29 \\ 32 \oplus 01 & = & 33 \end{array} \qquad \begin{array}{lcl} 06 \oplus 35 & = & 33 \\ 10 \oplus 35 & = & 25 \\ 16 \oplus 35 & = & 23 \\ 1C \oplus 35 & = & 29 \\ 22 \oplus 35 & = & 17 \\ 24 \oplus 35 & = & 11 \\ 28 \oplus 35 & = & 1D \\ 32 \oplus 35 & = & 07 \end{array}$$

Thus, possible keys are:

$$\{07, 11, 17, 1D, 23, 25, 29, 33\}$$

Furthermore, suppose we know two input $S1$ as 21 and 15 which XORs to 34, and output XOR as 3



This gives the key values:

$$01 \oplus 21 = 20 \qquad 01 \oplus 15 = 14$$
$$02 \oplus 21 = 23 \qquad 02 \oplus 15 = 17$$
$$15 \oplus 21 = 34 \qquad 15 \oplus 15 = 00$$
$$21 \oplus 21 = 00 \qquad 21 \oplus 15 = 34$$
$$35 \oplus 21 = 14 \qquad 35 \oplus 15 = 29$$
$$36 \oplus 21 = 17 \qquad 36 \oplus 15 = 23$$

as

$$\{00, 14, 17, 20, 23, 34\}$$

The correct key value must appear in both of these sets:

$$\{07, 11, 17, 1D, 23, 25, 29, 33\}$$

$$\{00, 14, 17, 20, 23, 34\}$$

Intersecting these two sets, we obtain

$$\{17, 23\}$$

Thus, the key value is either 17 or 23

In order to determine which one of these is the correct value, we need more input/output XORs

Characteristic

The differential input with the highest probability, which can be traced through several rounds

Two observations:

The XOR of pairs is linear in the E expansion:

$$E(X) \oplus E(X^*) = E(X \oplus X^*) = E(X')$$

The XOR of pairs is independent of the key:

$$\begin{aligned} S_I &= S_E \oplus S_K \\ S_I^* &= S_E^* \oplus S_K \\ S_I \oplus S_I^* &= S_E \oplus S_K \oplus S_E^* \oplus S_K \\ S_I' &= S_E \oplus S_E^* \\ S_I' &= S_E' \end{aligned}$$

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^{43} known plaintexts, easier but still in practise infeasible

Linear Cryptanalysis

- find linear approximations with prob p != ½

  $P[i_1,i_2,\ldots,i_a] \oplus C[j_1,j_2,\ldots,j_b] = K[k_1,k_2,\ldots,k_c]$

  where i_a,j_b,k_c 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: $|p-{}^1/_2|$

DES Design Criteria

- as reported by Coppersmith in [COPP94]
- 7 criteria for S-boxes provide for
  - non-linearity
  - resistance to differential cryptanalysis
  - good confusion
- 3 criteria for permutation P provide for
  - increased diffusion

Block Cipher Design

- basic principles still like Feistel's in 1970's
- number of rounds
  - more is better, exhaustive search best attack
- function f:
  - provides "confusion", is nonlinear, avalanche
  - have issues of how S-boxes are selected
- key schedule
  - complex subkey creation, key avalanche

Summary

- have considered:
  - block vs stream ciphers
  - Feistel cipher design & structure
  - DES
    - details
    - strength
  - Differential & Linear Cryptanalysis
  - block cipher design principles