现代密码学理论与实践

第3章 分组密码和数据加密标准

第3章 分组密码和数据加密标准

- 分组密码是一种加密解密算法，将输入明文分组当做一个整体处理，输出一个等长的密文分组。
- 许多分组密码都采用Feistel结构，这样的结构由许多相同的轮函数组成。每一轮里，对输入数据的一半进行代换，接着用一个置换来交换数据的两个部分，扩展初始的密钥使得每一轮使用不同的子密钥。
- DES是应用最为广泛的分组密码，它扩展了经典的Feistel结构。DES的分组和密钥分别是64位和56位。
- 差分分析和线性分析是两种重要的密码分析方法。DES对这两种攻击有一定的免疫性。

3.1 分组密码的原理

- 流密码(Stream Cipher)和分组密码(Block Cipher)

  如果密文不仅与最初给定的算法和密钥有关，同时也与明文位置有关(是所处位置的函数)，则称为流密码体制。加密以明文比特为单位，以伪随机序列与明文序列模2加后，作为密文序列，一次一比特/字节

  如果经过加密所得到的密文仅与给定的密码算法和密钥有关，与被处理的明文数据在整个明文中的位置无关，则称为分组密码体制。通常以大于等于64位的数据块为单位，加密得相同长度的密文。

乘积密码的设计思想

- 1949年，Claude Shannon引进了Substitution-Permutation (S-P) Networks的思想，即现代的乘积加密器，形成了现代分组加密的基础。S-P Networks 是基于替代和置换这两个基本操作的.
- 乘积加密提供了对明文信息处理所做的confusion (扰乱)和diffusion (扩散)
  - Shannon认为，为了对付基于统计分析的密码破译，必须对明文作confusion(扰乱)和diffusion(扩散)处理，以减少密文的统计特性，为统计分析制造障碍。
  - **diffusion** –扩散，明文统计结构扩散消失到大批密文统计特性中，使明文和密文之间统计关系尽量复杂；
  - **confusion** –扰乱，使密文和加密密钥之间的关系尽量复杂。

3.1.2 Feistel密码结构的设计动机

- 分组密码
  - 大多数分组密码基于 **Feistel Cipher Structure**
  - 分组加密器本质上就是一个巨大的替换器
  - 64位的分组就有 2^{64}种输入
  - 采用了乘积加密器的思想，即轮流使用替代和置换
- Feistel密码结构的设计动机
  - 分组密码对n比特的明文分组进行操作，产生一个n比特的密文分组，共有2^n个不同的明文分组，每一种都必须产生一个唯一的密文分组，这种变换称为可逆的或非奇异的。

可逆映射		不可逆映射	
00	11	00	11
01	10	01	10
10	00	10	01
11	01	11	01

n = 4时的一个普通代换密码的结构

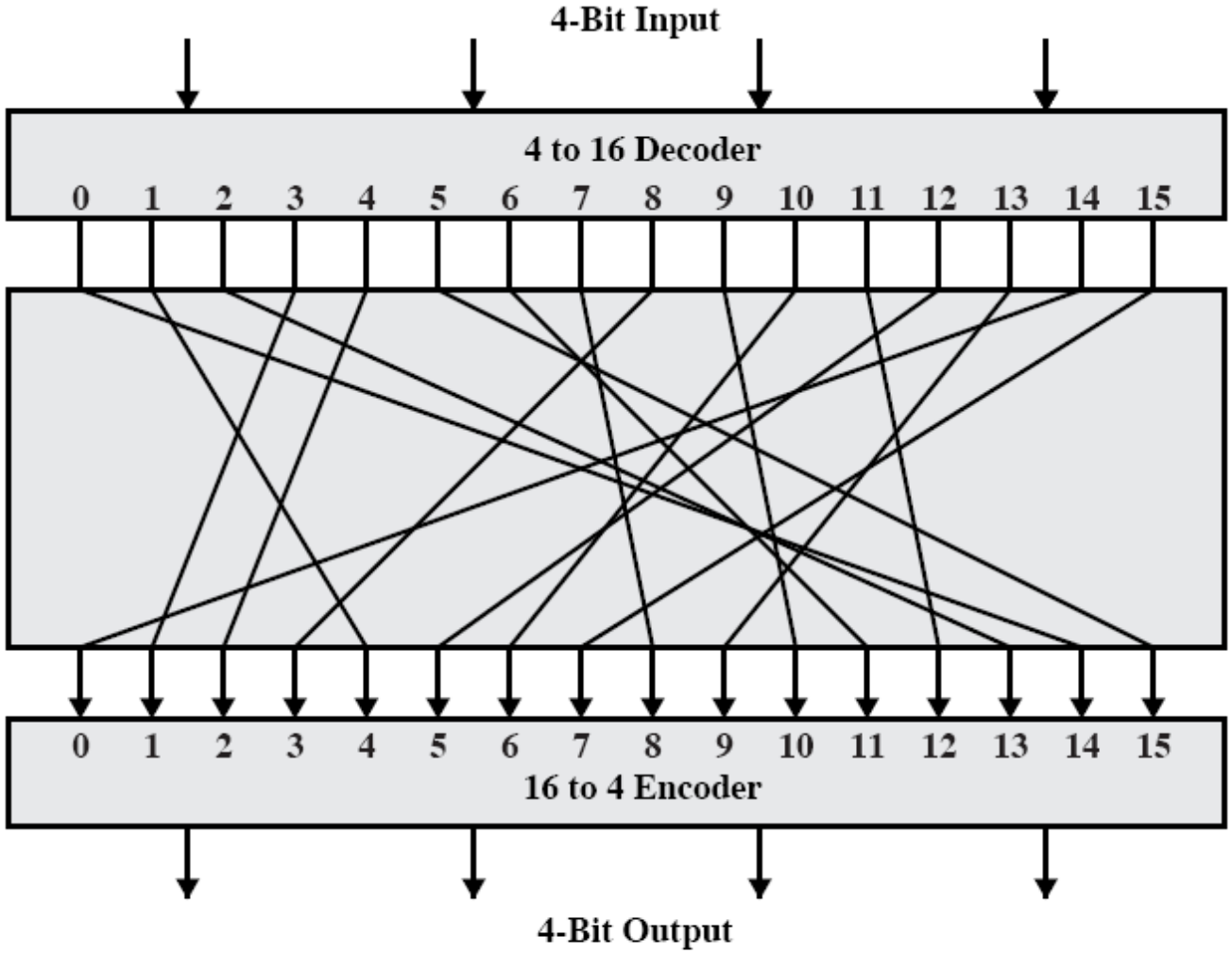


Figure 3.1 General *n*-bit-*n*-bit Block Substitution (shown with $n = 4$)

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

3.1.3 Feistel密码结构

- 1973年，Horst Feistel提出了基于可逆乘积加密器概念的Feistel Cipher：
  - 将输入分组分成左右两部分，实施Shannon’s的substitution-permutation network 概念
  - 对左半部数据实施多回合的替代操作(substitution)
  - 对右半部数据和子密钥应用轮函数F，其输出与左一半做异或
  - 将这两部分进行互换(permutation swapping)

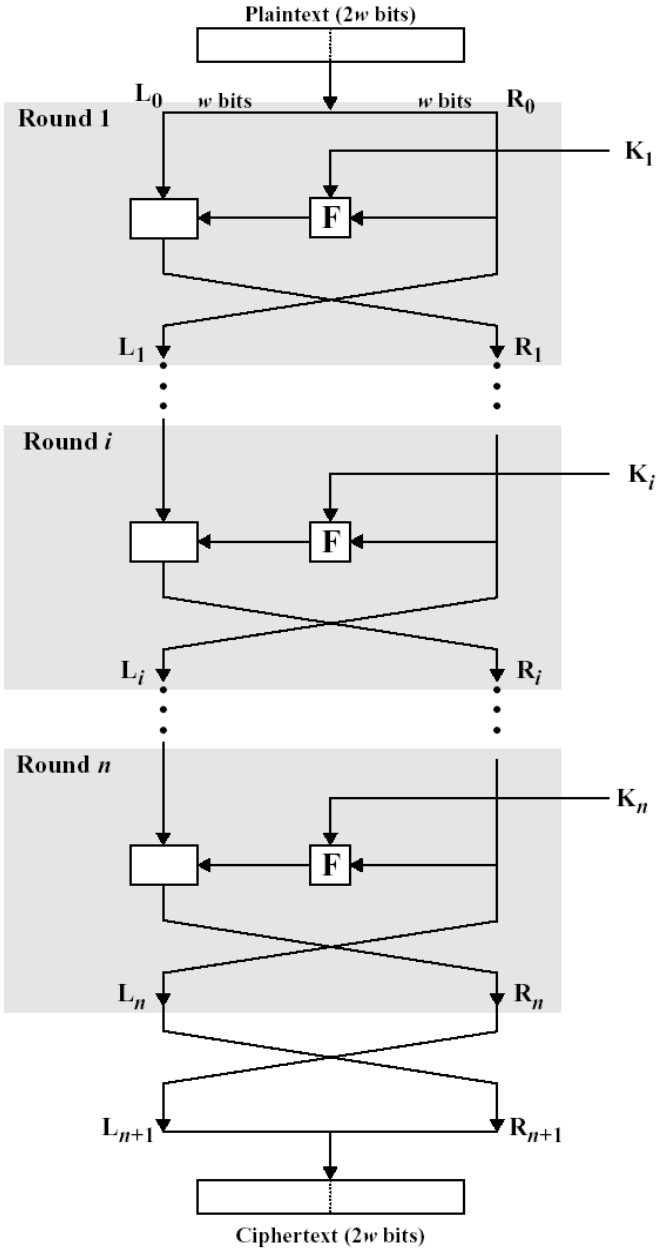


Figure 3.5 Classical Feistel Network

Feistel加密器设计原则

- 分组长度：分组越长则安全性越高，但加/解密速度越低，分组长度为64位是一个合理的折衷
- 密钥长度：密钥越长越安全，但加/解密速度越低，64位长的密钥已被证明是不安全的，128位是常用的长度
- 迭代次数：迭代越多越安全，通常为16次迭代
- 子密钥产生算法：越复杂则密码分析越困难
- 轮循环函数：越复杂则抗密码分析的能力越强
- 快速的软件加密/解密：算法的执行速度很重要
- 简化分析难度：算法简洁清楚，易于分析弱点，发现问题
- Feistel解密算法：以密文作为算法的输入，以相反的次序使用密钥K_i，K_n、K_{n-1}、…、K_0.

Feistel Cipher Encryption and Decryption

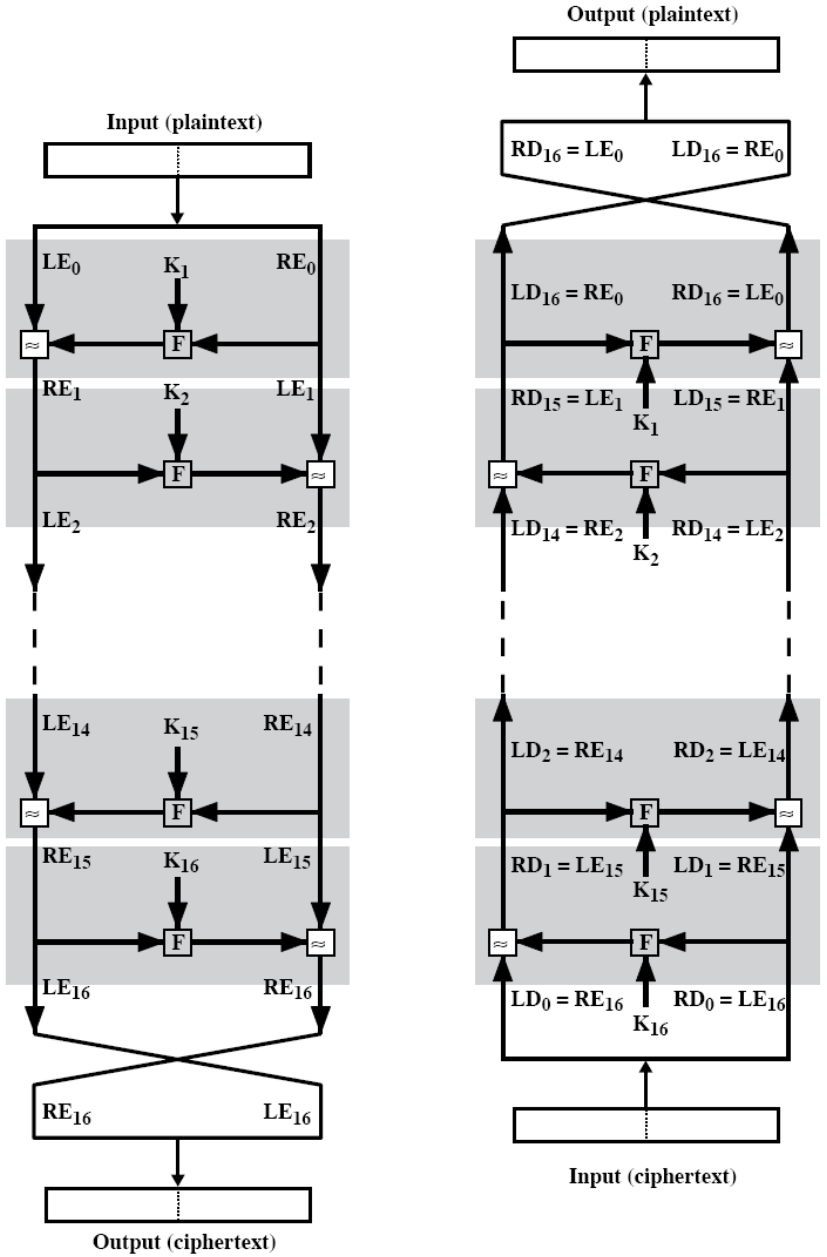


Figure 3.3 Feistel Encryption and Decryption

3.2 数据加密标准DES

- 历史的回顾
  - IBM公司在1971年由Horst Feistel领导开发了Lucifer Cipher，使用128位密钥加密64位的分组
  - Tuchman-Mayer在此基础上开发了一个商用密码，使用56位密钥加密64位分组，容易在单个芯片上硬件实现
  - 1973美国国家标准局NBS全面征集加密方案，作为国家密码标准
  - IBM提交了经过修改的Lucifer加密器，并最终在1976年被接受，公布为数据加密标准DES
- DES加密

DES加密算法的一般描述

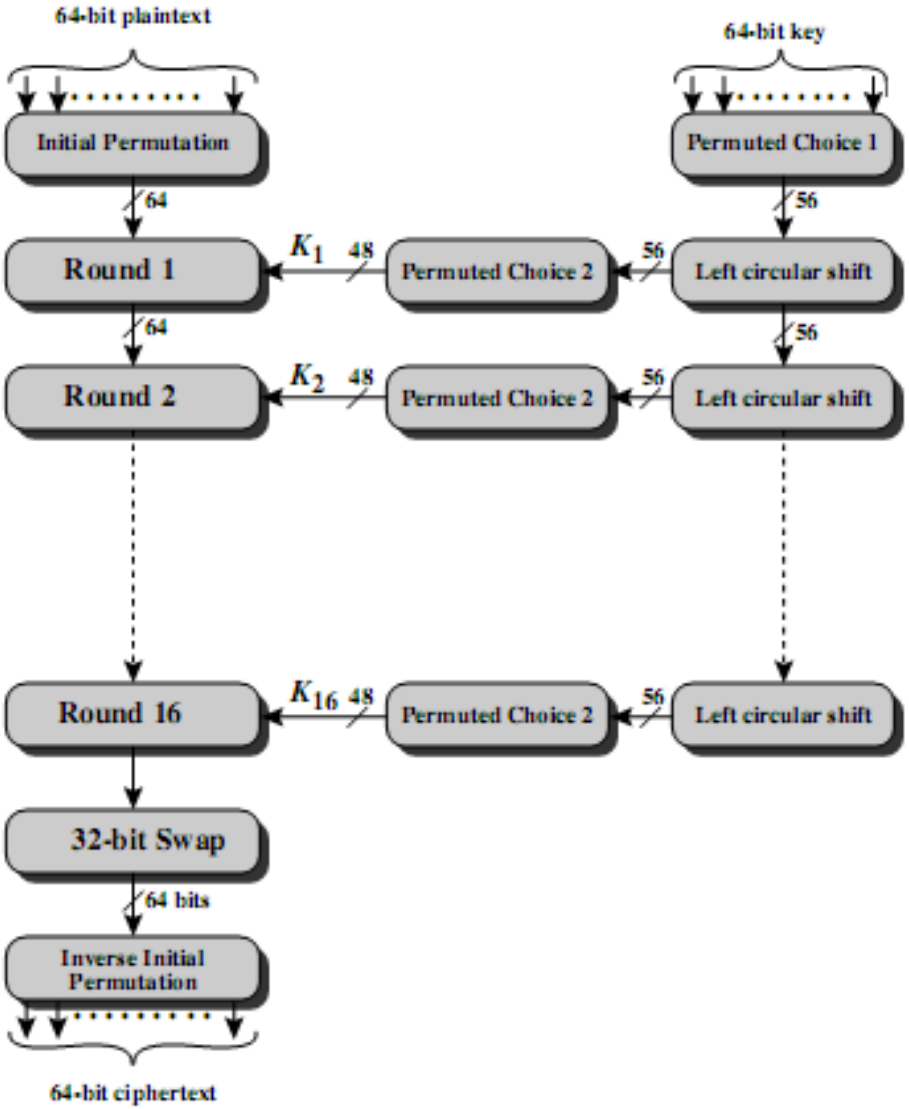


Figure 3.4 General Depiction of DES Encryption Algorithm

初始置换IP (Initial Permutation)和逆置换IP^{-1}

Table 3.2 Permutation Tables for DES

(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

扩充置换(E)和置换函数(P)

(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

Feistel Cipher分组加密循环细节

- 将明文分成左右两部分

  $L_i = R_{i-1}$

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

- 将32-bit右半部和48-bit子密钥做以下动作
  - 使用置换表E，将32位右半部R扩展成48位
  - 与48位子密钥做异或
  - 48位结果送给8个替换盒S-boxes，得到32位结果
  - 最后使用32位置换表P，把32位结果再进行一次置换处理

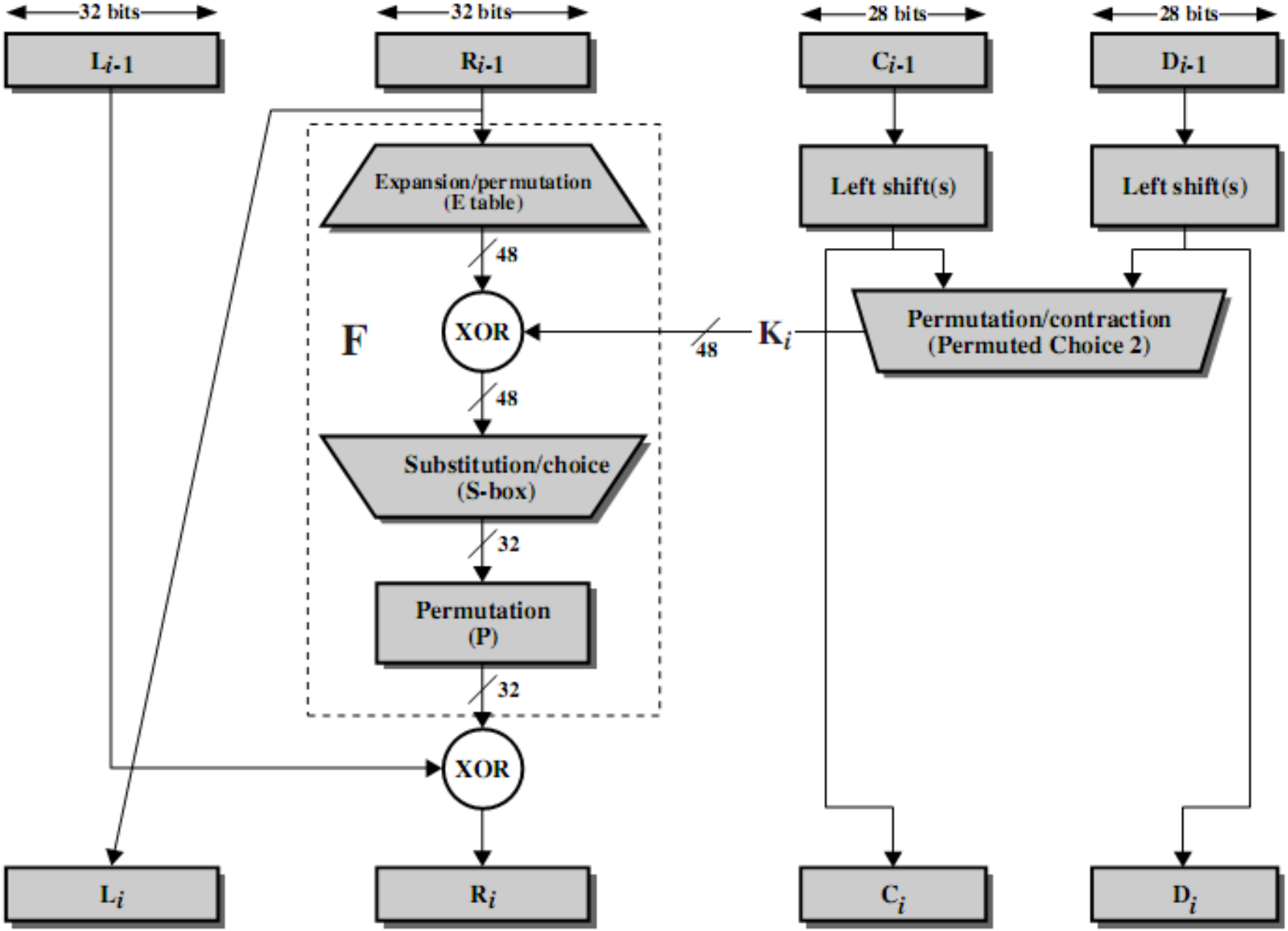


Figure 3.5 Single Round of DES Algorithm

S盒(Substitution Boxes)

- 有8个将6位数据映射成4位数据的S盒
- 6到4的映射规则是
  - 外侧的第1位和第6位用作行选择
  - 其余4位(2-5bit)用作列选择
  - 这样每盒就有4行16列，输出4位，8个S盒输出32位
- 行的选择依赖于数据和密钥
- 例如

  S(18 09 12 3d 11 17 38 39) = 5fd25e03

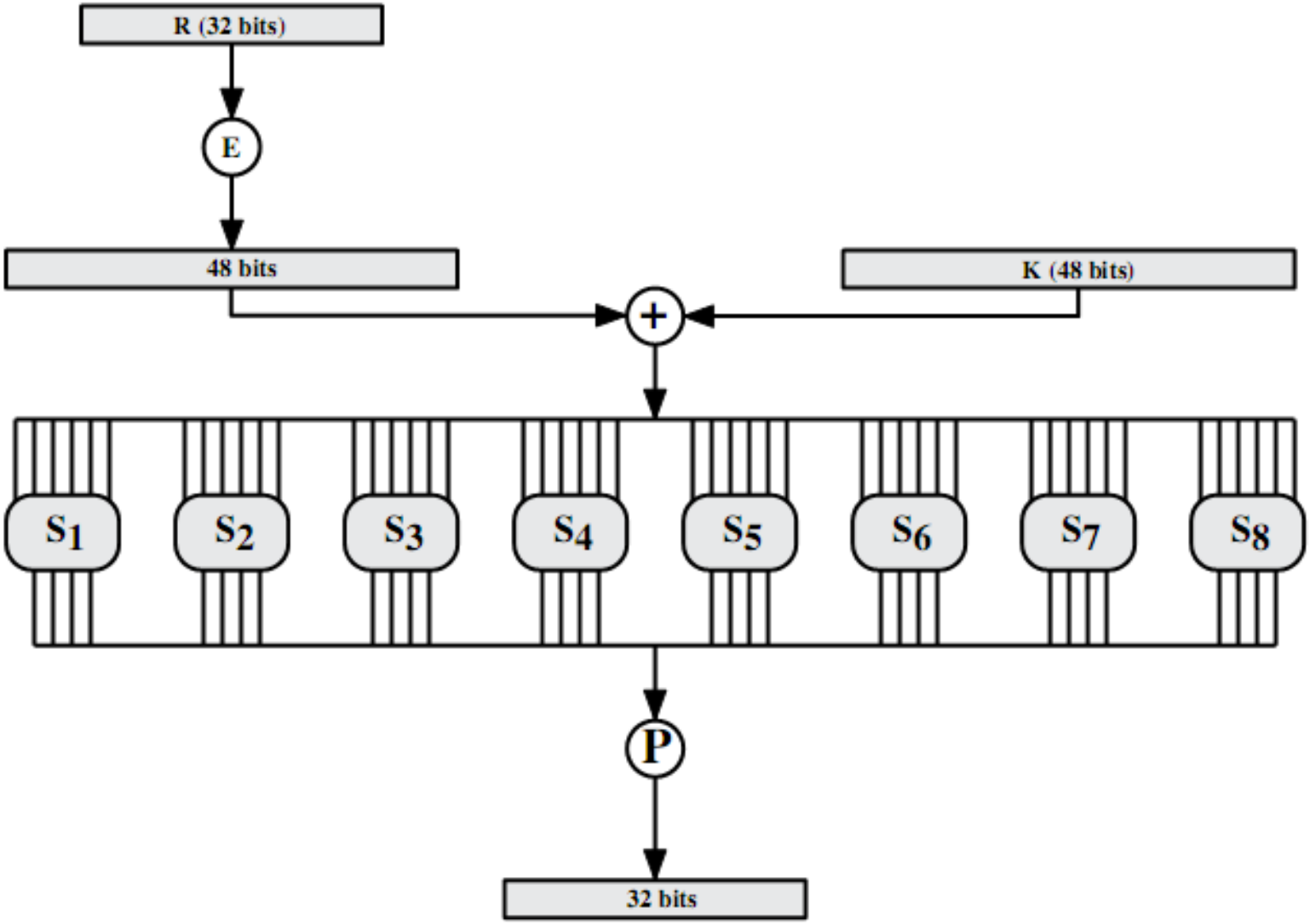


Figure 3.6 Calculation of F(R, K)

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

子密钥的产生

- 每一轮都要生成一个子密钥以供加密使用
- 子密钥生成过程包括
  - 使用密钥置换选择1(PC-1)，将56位密钥分成两半，每一部分28位
  - 使用置换选择2(PC-2)，从每一半中选出24位，形成48位子密钥用在某一轮的F函数中
  - 根据密钥左移表K将这两半分别左移1位或2位
  - 重复置换选择2(PC-2)，形成新的48位子密钥用在下一轮的F函数中

子密钥的产生

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

子密钥的产生

(c) Permuted Choice Two (PC-2)

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

(d) Schedule of Left Shifts

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

- DES的解密

  DES的解密是加密的逆过程，采用相同算法，但是子密钥使用的次序正好相反。

DES加密的雪崩效应

- 雪崩效应 Avalanche Effect

  明文或密钥的一比特的变化，引起密文许多比特的改变。如果变化太小，就可能找到一种方法减小有待搜索的明文和密文空间的大小。

- 如果用同样密钥加密只差一比特的两个明文：

  0000000000000000.....00000000

  1000000000000000.....00000000

  3次循环以后密文有21个比特不同，16次循环后有34个比特不同

- 如果用只差一比特的两个密钥加密同样明文：

  3次循环以后密文有14个比特不同，16次循环后有35个比特不同

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

3.3 DES的安全强度

- 密钥的长度问题
  - 56-bit 密钥有2^{56} = 72,057,584,037,927,936 ≈ 7.2亿亿之多
  - 强力搜索(brute force search) 似乎很困难，20世纪70年代估计要1000－2000年
  - 技术进步使穷举搜索成为可能
    - 1997年1月29日，RSA公司发起破译RC4、RC5、MD2、MD5，以及DES的活动，破译DES奖励10000美金。明文是：Strong cryptography makes the world a safer place. 结果仅搜索了24.6%的密钥空间便得到结果，耗时96天。
    - 1998年在一台专用机上(EFF)只要三天时间即可
    - 1999年在超级计算机上只要22小时!
    - 现在只需要10小时！

DES的安全强度

- S-box问题
  - 其设计标准没有公开，但是迄今没有发现S盒存在致命弱点
- 计时攻击
  - 计时攻击利用的事实是加密或解密算法对于不同的输入所花的时间有细微的差别
  - DES能够很好地抵抗计时攻击
- 差分密码分析攻击问题
  - DES对差分分析攻击有较好的免疫力
  - 针对DES的密码分析攻击主要利用了加密器的深层结构
    - 搜集加密信息
    - 最终设法恢复部分或全部子密钥的位
    - 如果必要的话对其余部分再辅以穷举搜索
  - 这些攻击实际上是统计分析，包括
    - 差分分析、线性分析、相关密钥攻击

3.5 差分分析和线性分析

- 差分密码分析Differential Cryptanalysis
  - 1990年，Murphy、Biham和Shamir首次提出用差分密码分析攻击分组密码和散列函数，是第一种可以以少于2^{55}的复杂性对DES进行破译的方法。若有2^{47}个选择明文，用差分分析就可以在2^{47}次加密运算内成功攻击DES。尽管2^{47}比2^{55}小得多，但是要拥有2^{47}个选择明文的条件使得这种方法只具有理论上的意义。对DES并不奏效。
  - 差分密码分析攻击方法

    关注一对明文在加密过程中通过轮函数的演变情况，而不是观测单个明文分组的演变。

    两个报文的异或$\Delta m=m\oplus m'$，中间过程有$\Delta m_i=m_i\oplus m_i'$，则

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

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

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

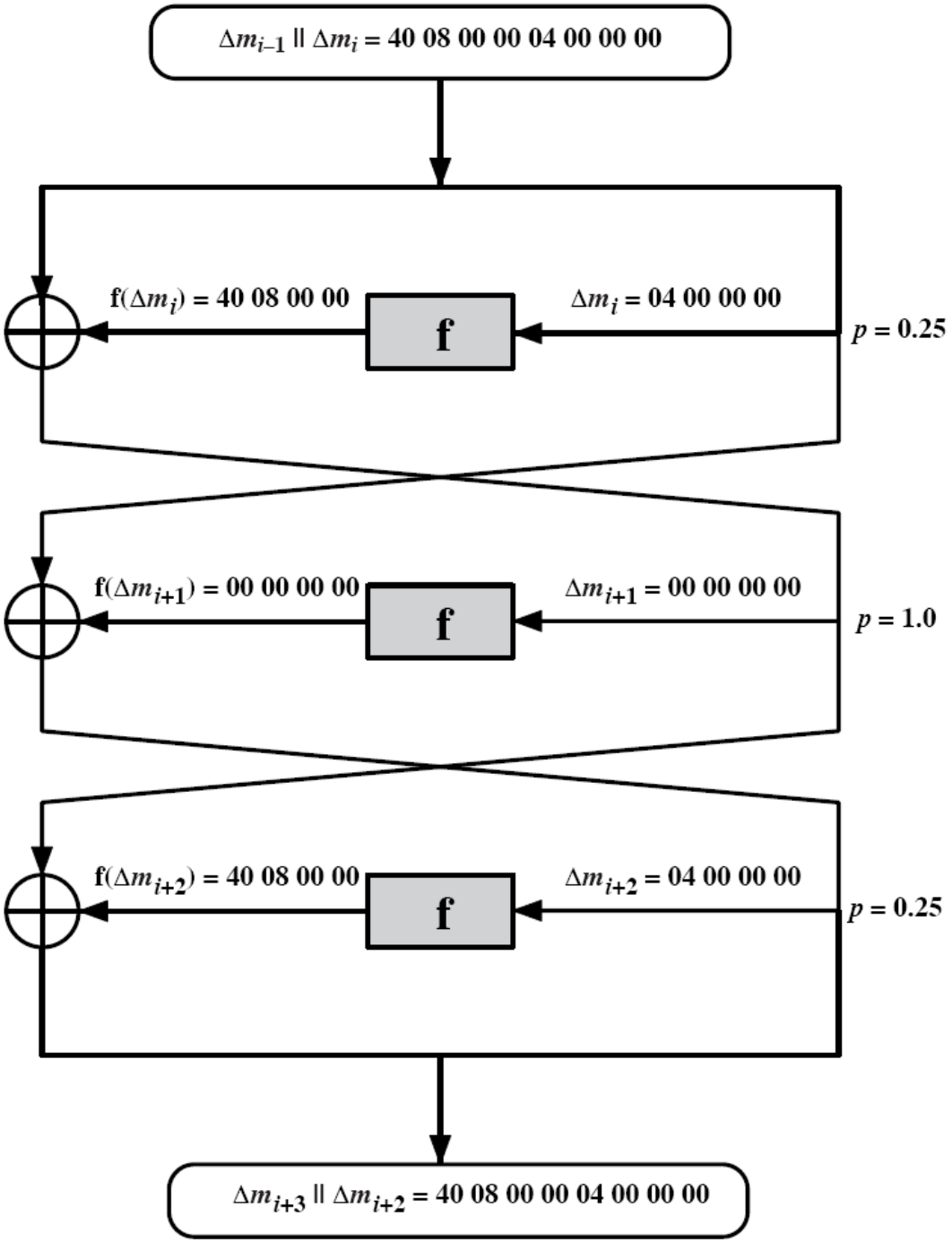
差分密码攻击

- 假设函数*f*有许多对具有相同差分的输入，当使用相同的子密钥时产生相同差分的输出。精确地说，若差分为X的所有输入，产生差分为Y的输出占所有输出的百分比为*p*，则称由差分X导致差分Y的概率为*p*。
- 假设有许多X都会以很高的概率产生某些特定的差分，因此如果我们以很高的概率知道Δm_{i-1}和Δm_i，就能以很高的概率知道Δm_{i+1}。而且，如果知道很多有关差分的数据，那么确定函数*f*所使用的子密钥就是可行的。

Differential Cryptanalysis

- 反复加密已知输入异或的明文对，直到得到期望的输出异或
- 如果找到
  - 中间轮次有满足所需的异或，则找到了正确的一对输入 right pairs
  - 否则找到的是错误的一对输入wrong pairs，对攻击来说比例是S/N
- 这样可以推测中间轮次使用的密钥值
  - right pairs 说明了同样的密钥位
  - wrong pairs 说明是随机数
- 对于大轮次加密来说，找到正确的概率很低，要分析的对数比存在的64-bit inputs多
- Biham和Shamir表明，13轮的分析可以破解整个16轮的DES

- 如果使用相同子密钥，对于f，具有系统差值的许多输入对将产生相同的输出差值，即如果在异或值为X的输入对中，有p部分使输出异或值为Y，则X产生Y的概率为p。假定存在很多X，具有很大概率产生一个特定的输出差值。如果已知Δm_{i-1}和Δm_i具有很大概率，则Δm_{i+1}也具有很大概率。如果确定很多这样的差值，则容易确定函数f中使用的子密钥。
- 右图说明差值经过三次循环后的传播情况，输出差值为所示差值的概率为0.25x1x0.25=0.0625



线性密码分析Linear Cryptanalysis

- 是1993年提出的另一种统计攻击，基于找到DES中进行变换的线性近似来进行攻击，可以在有2^{43}个已知明文的情况下破译DES密钥，但仍然是不可行的。
- 基本原理
  - 令明文分组为P[1], ..., P[n], 密文分组为C[1], ..., C[n], 密钥为K[1], ..., K[m], 则定义：

    $A[i, j, ..., k] = A[i] \oplus A[j] \oplus ... \oplus A[k]$
  - 线性密码分析的目标是找到如下有效线性方程：

    $P[\alpha_1, \alpha_2, ..., \alpha_a] \oplus C[\beta_1, \beta_2, ..., \beta_b] = K[\gamma_1, \gamma_2, ..., \gamma_c]$
  - 其中，x=0或1；1≤a，b≤n，1≤c≤m，α，β 和γ等表示固定的唯一的比特位置。方程以概率p≠0.5成立，p离0.5越远，方程越有效。
  - 对于大量的明文密文对，计算方程左边的值，如果结果中有一半以上为0，则假定$K[\gamma_1, \gamma_2, ..., \gamma_c] = 0$；如果大多为1，则假定$K[\gamma_1, \gamma_2, ..., \gamma_c] = 1$。

3.5 分组密码的设计原理

- S-boxes的设计准则：增加扰乱性
  - 输出比特不应太接近输入比特的一个线性函数
  - 每一行应该包括所有16种比特组合
  - 两个输入相差一个比特，输出必须相差两个比特
  - 如果两个输入刚好在两个中间比特上不同，输出必须在至少两个比特上不同
  - 两个输入前两位不同而最后两位相同，两个输出必须不同
  - 具有非零6比特差值的输入，32对中有不超过8对输出相同
- 置换P的设计准则：增加扩散性
  - 第i次循环时每个S盒输出的四个比特被分布开，以便其中两个影响下一循环的中间比特，两个影响两端的比特
  - 每个S盒输出的四个比特影响下一循环的6个不同的S盒，并且任何两个都不会影响同一个S盒
  - 如果S_j的一个输出比特影响下一循环S_k的中间比特，则S_k的一个输出比特就不能影响S_j的一个中间比特。

分组密码的设计原理

- 迭代轮数

  迭代次数越多则进行密码分析的难度就越大，选择准则是要使已知的密码分析工作量大于简单的穷举密钥搜索的工作量。

- 函数F的设计
  - 函数F的设计准则(非线性、严格雪崩效应、位独立)

    提供扰乱作用，要求强非线性，良好的雪崩性质
  - S-boxes的设计
    - 希望S盒输入向量的任何变动在输出方都产生看似随机的变动，这两种变动之间的关系应该是非线性的并难以用线性函数近似。
    - S盒的大小：较大的抗攻击能力强，但越大实现越困难
    - S盒的组织：要求高度非线性，随机性
- 密钥扩展算法
  - 选择子密钥时要使得推测各子密钥和由此推出主密钥难度尽可能大，保证密钥/密文的严格雪崩效应准则和位独立准则。

Summary

We have considered in this chapter:

- Block cipher design principles
- DES
  - details
  - strength
- Differential & Linear Cryptanalysis