现代密码学

Modern Cryptography

张方国

中山大学计算机学院

Office: Room 305, IM School Building

E-mail: isszhfg@mail.sysu.edu.cn

HomePage: https://cse.sysu.edu.cn/content/2460

第九讲 私钥密码学：DES

- DES历史
- DES算法
- DES安全性

DES

美国数据加密标准——DES（Data Encryption Standard）。

1. 美国制定数据加密标准简况

通信与计算机相结合是人类步入信息社会的一个阶梯，它始于六十年代末，完成于90年代初。计算机通信网的形成与发展，要求信息作业**标准化**，安全保密亦不例外。

美国NBS在1973年5月15公布了征求建议。1974年8月27日NBS再次出公告征求建议，对建议方案提出如下要求：（1）算法必须完全确定而无含糊之处；（2）算法必须有足够高的保护水准，即可以检测到威胁，恢复密钥所必须的运算时间或运算次数足够大；（3）保护方法必须只依赖于密钥的保密；（4）对任何用户或产品供应者必须是不加区分的。

DES，标准简况

IBM公司从60年代末即看到通信网对于这种加密标准算法的需求，投入了相当的研究力量开发，成立了以Tuchman博士为领导的研究新密码体制的小组，H. Fistel进行设计，并在1971年完成的LUCIFER密码 (64 bit分组，代换-置换，128 bit密钥)的基础上，改进成为建议的DES体制。

1975年3月17日NBS公布了这个算法，并说明要以它作为联邦信息处理标准，征求各方意见。1977年1月15日建议被批准为联邦标准[FIPS PUB 46]，并设计推出DES芯片。DES开始在银行、金融界广泛应用。1981年美国ANSI 将其作为标准，称之为DEA[ANSI X3.92]。1983年国际标准化组织(ISO)采用它作为标准，称作DEA-1。

DES ，标准简况

1984年9美国总统签署145号国家安全决策令(NSDD)，命令NSA着手发展新的加密标准，用于政府系统非机密数据和私人企事业单位。1998年DES不再用。

虽然DES不会长期地作为数据加密标准算法，但它仍是迄今为止得到最广泛应用的一种算法，也是一种最有代表性的分组加密体制。

1993年4月，Clinton政府公布了一项建议的加密技术标准，称作密钥托管加密技术标准**EES**(Escrowed Encryption Standard)。其开发设计始于1985年，由NSA负责研究。1990年完成评价工作。其算法为SKIPJACK。已由Mykotronix公司开发芯片产品，编程后为MYK-78。算法属美国政府SECRET密级。

DES ，标准简况

DES发展史确定了发展公用标准算法模式，而EES的制定路线与DES的背道而驰。1995年5月AT&T Bell Lab的M. Blaze博士在PC机上用45分钟时间使SKIPJACK的 LEAF协议失败，伪造ID码获得成功。虽NSA 声称已弥补，但丧失了公众对此体制的信心。1995年7月美国政府宣布放弃用EES来加密数据。

重新回到制定DES标准立场。1997年1月美国NIST着手进行AES（Advanced Encryption Standard）的研究，成立了标准工作室。1997年4月15日讨论了AES的评估标准，开始在世界范围征集AES的建议算法，截止时间为1998年6月15日。1998年8月20~22日经评审选定并公布了15个候选算法。

Horst Feistel

1915 - 1990

- MIT, Stanford
- @IBM: Feistel network

- moved to the US as 19 years old
- placed under house arrest during WWII
- then became American

Feistel网络

- Horst Feistel, （working at IBM Thomas J Watson Research Labs ）70's初，设计了这样的结构，我们现在叫做**feistel cipher**
- 思想是把输入块分成左右两部分 L(i-1) 和R(i-1), 变换是在密码的第I轮只使用R(i-1)
- 函数 g incorporates one stage of the S-P network的每个阶段有g 工作，由第I 个密钥控制（叫子密钥）

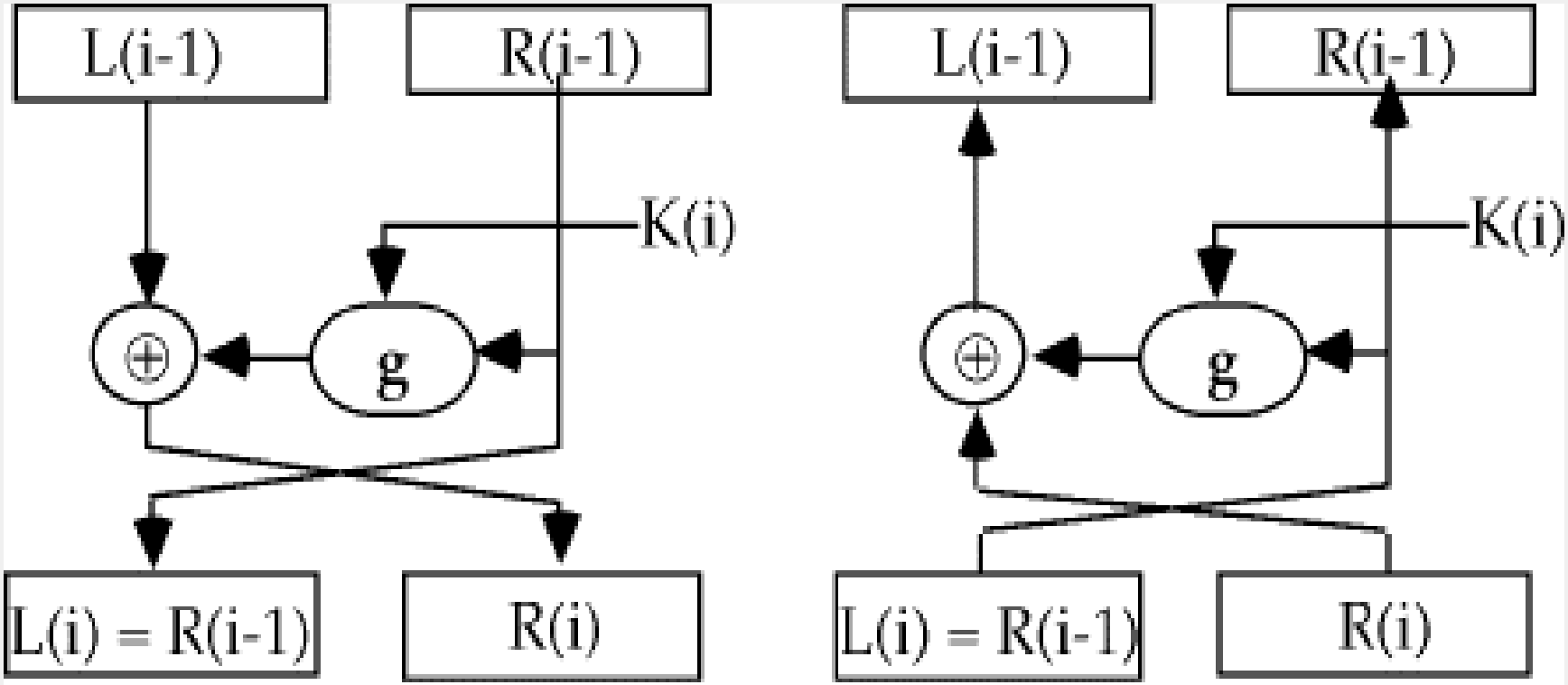


Fig 2.4 - A Round of a Feistel Cipher

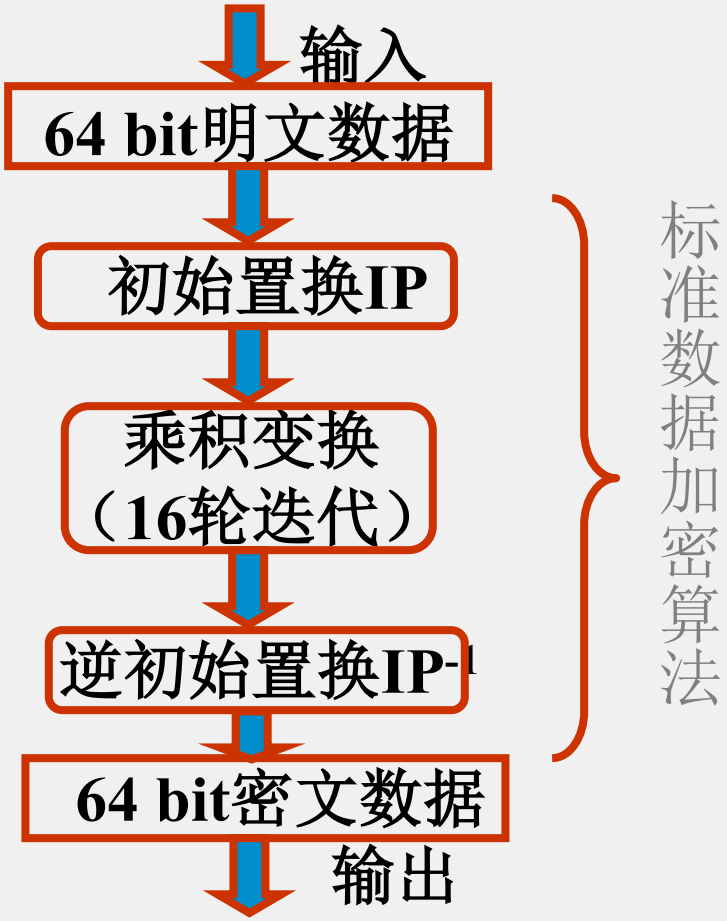
DES，算法

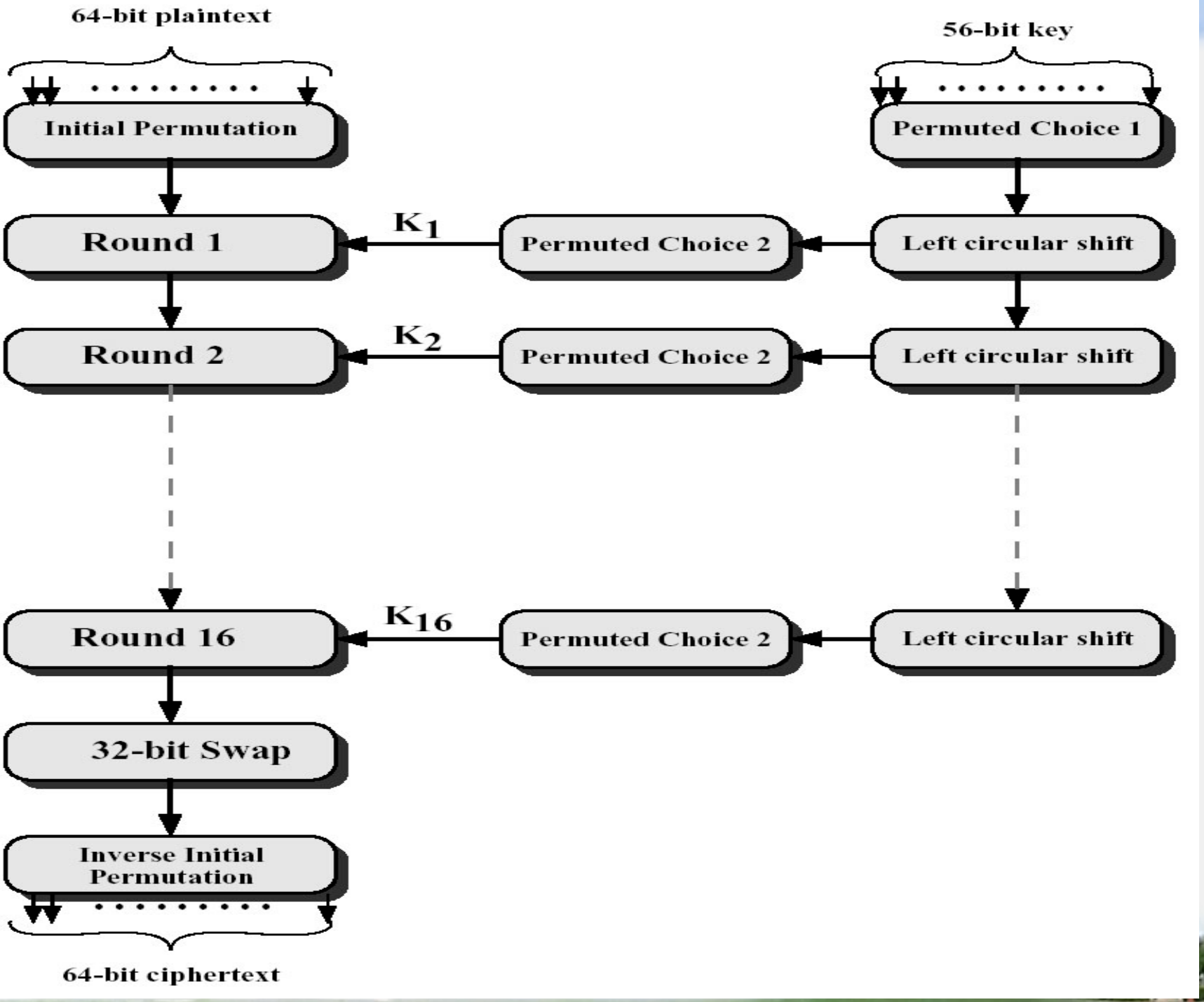
算法：分组长度为64 bits (8 bytes)，**密文分组长度**也是64 bits。**密钥长度**为64 bits，有8 bits奇偶校验，有效密钥长度为56 bits。**算法主要包括：初始置换*IP*、16轮迭代的乘积变换、逆初始置换IP^{-1}**以及16个**子密钥产生器**。

初始置换*IP*：将64 bit明文的位置进行置换，得到一个乱序的64 bit明文组，而后分成左右两段，每段为32 bit，以L_0和R_0表示，*IP*中各列元素位置号数相差为8，相当于将原明文各字节按列写出，各列比特经过偶采样和奇采样置换后，再对各行进行逆序。将阵中元素按行读出构成置换输出。

逆初始置换IP^{-1}。将16轮迭代后给出的64 bit组进行置换，得到输出的密文组。输出为阵中元素按行读得的结果。

DES，算法


输入
64 bit明文数据
初始置换IP
乘积变换
（16轮迭代）
逆初始置换IP-1
64 bit密文数据
输出
标准数据加密算法


64-bit plaintext
56-bit key
Initial Permutation
Permuted Choice 1
Round 1
K1
Permuted Choice 2
Left circular shift
Round 2
K2
Permuted Choice 2
Left circular shift
Round 16
K16
Permuted Choice 2
Left circular shift
32-bit Swap
Inverse Initial Permutation
64-bit ciphertext

DES ，算法

IP和IP^{-1}在密码意义上作用不大，因为输入组x与其输出组$y=IP(x)$ (或$IP^{-1}(x)$)是已知的一一对应关系。它们的作用在于打乱原来输入x的ASCII码字划分的关系，并将原来明文的校验位$x_8, x_{16}, \cdots, x_{64}$变成为$IP$输出的一个字节。

乘积变换。它是DES算法的核心部分。将经过IP置换后的数据分成32 bit的左右两组，在迭代过程中彼此左右交换位置。每次迭代时只对右边的32 bit进行一系列的加密变换，在此轮迭代即将结束时，把左边的32 bit与右边得到的32 bit逐位模2相加，作为下一轮迭代时右边的段，并将原来右边未经变换的段直接送到左边的寄存器中作为下一轮迭代时左边的段。在每一轮迭代时，右边的段要经过**选择扩展运算E、密钥加密运算、选择压缩运算S、置换运算P和左右混合运算**。

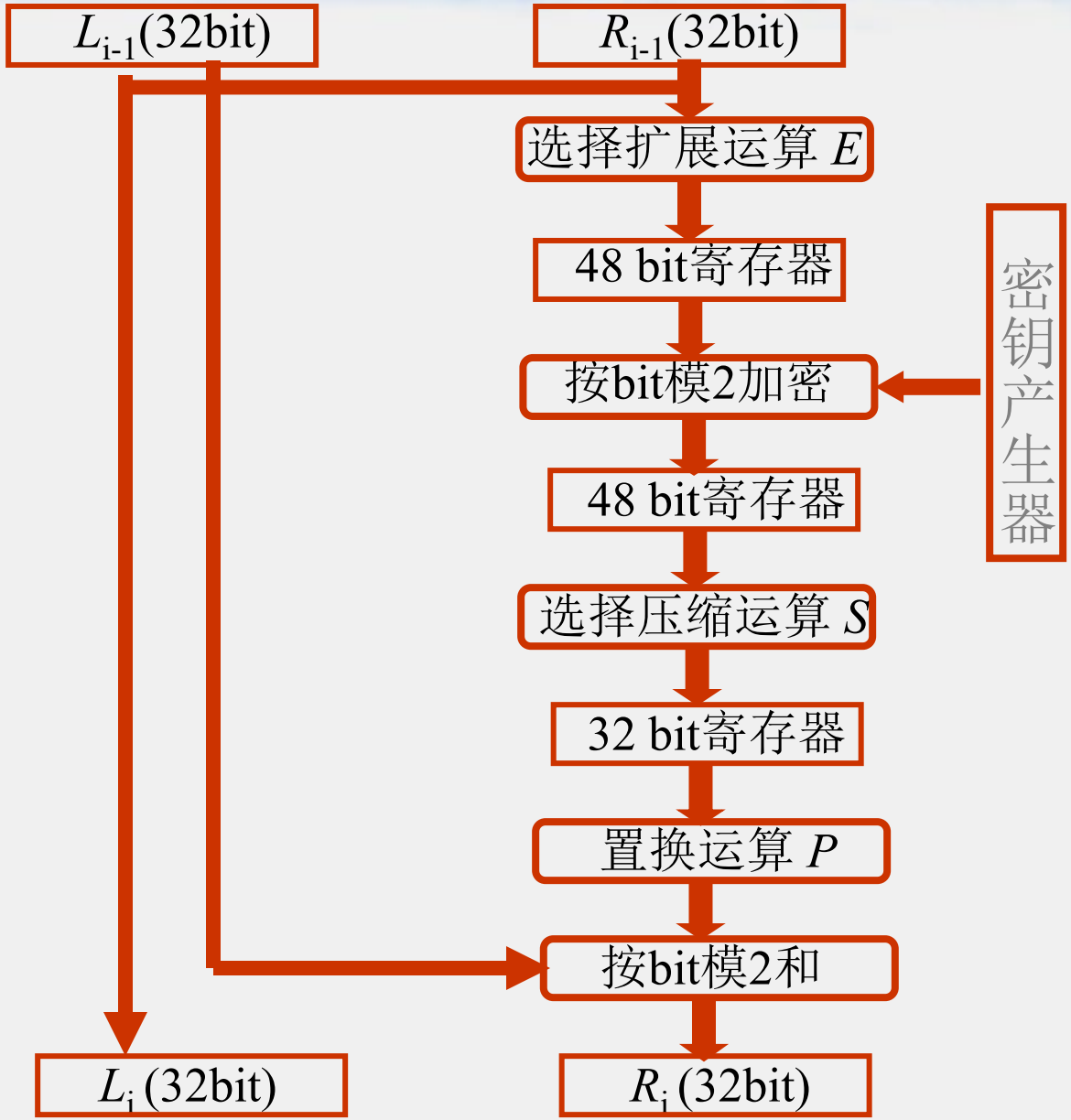
初始置换IP和初始逆置换IP^{-1}

初始置换 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

初始逆置换 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


L_{i-1}(32bit)
R_{i-1}(32bit)
选择扩展运算 E
48 bit寄存器
密钥产生器
按bit模2加密
48 bit寄存器
选择压缩运算 S
32 bit寄存器
置换运算 P
按bit模2和
L_i (32bit)
R_i (32bit)

乘积变换框图

DES，算法

选择扩展运算*E*。将输入的32 bit R_{i-1}扩展成48 bit的输出，其变换表在下面给出。令*s*表示*E*原输入数据比特的原下标，则*E*的输出是将原下标$s \equiv 0$或1(mod 4)的各比特重复一次得到的，即对原第32, 1, 4, 5, 8, 9, 12, 13, 16, 17, 20, 21, 24, 25, 28, 29各位都重复一次,实现数据扩展。将表中数据按行读出得到48 bit输出。

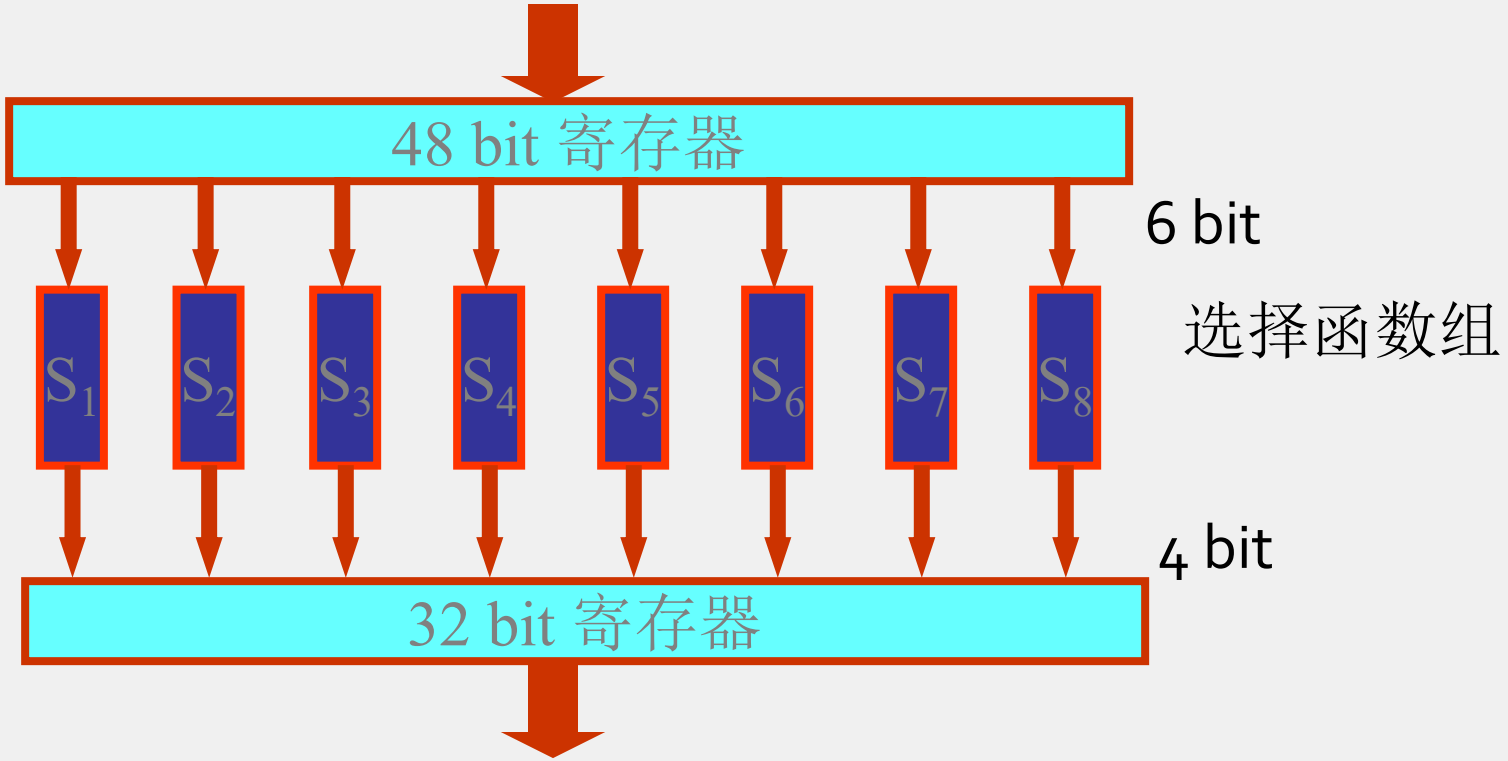
密钥加密运算。将子密钥产生器输出的48 bit子密钥k_i与选择扩展运算*E*输出的48 bits数据按位模2相加。

选择压缩运算*S*。将前面送来的48 bit数据自左至右分成8组，每组为6 bit。而后并行送入8个*S*一盒，每个S盒为一非线性代换网络，有4个输出。

选择扩展运算

32	01	02	03	04	05
04	05	06	07	08	09
08	09	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	01

DES ，算法

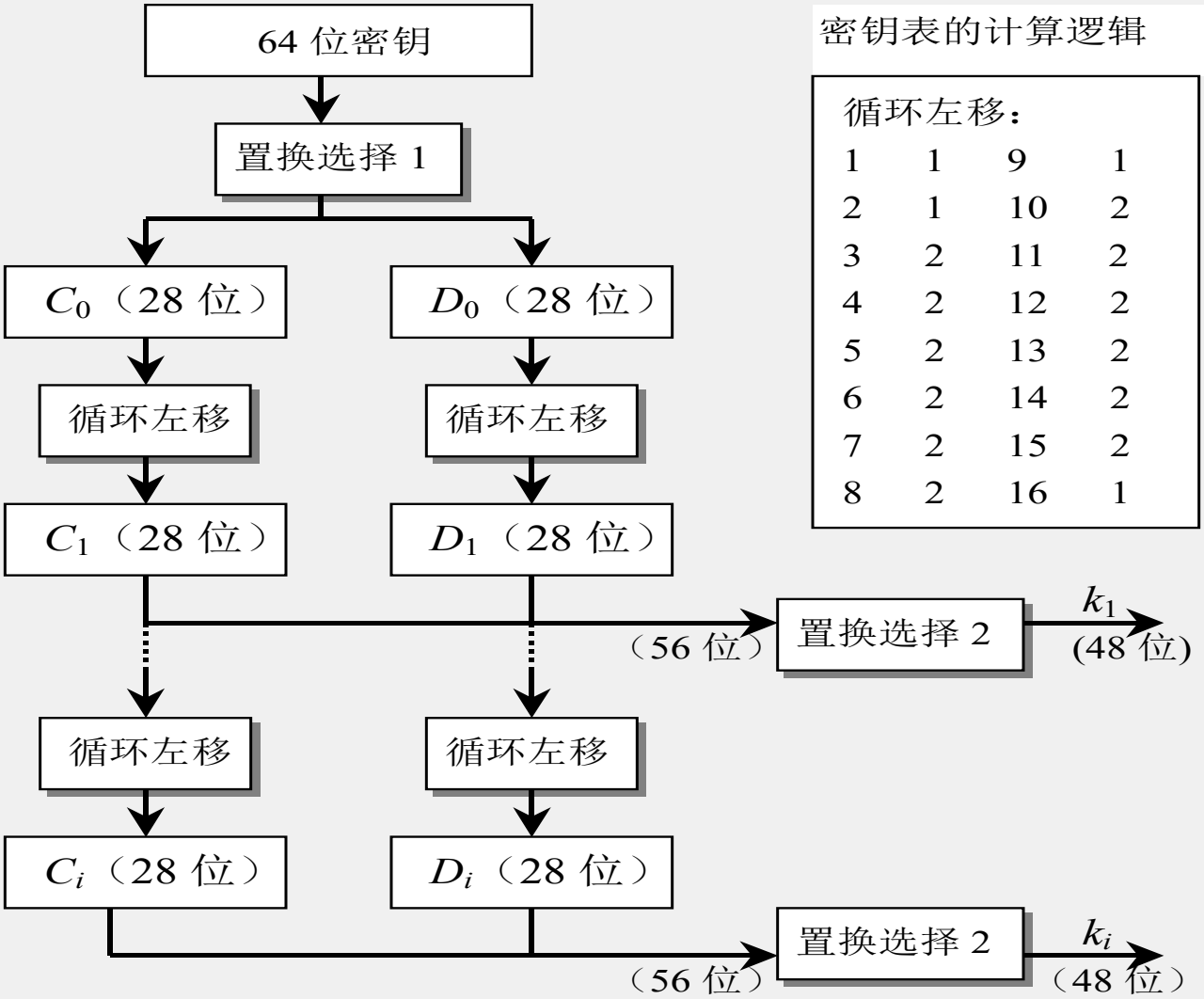


选择压缩运算S

DES 子密钥产生器

将64 bit初始密钥经过**置换选择PC1、循环移位置换、置换选择PC2**给出每次迭代加密用的子密钥k_i。在64 bit初始密钥中有8位为校验位，其位置号为8、16、32、48、56和64。其余56位为有效位，用于子密钥计算。将这56位送入置换选择*PC1*。经过坐标置换后分成两组，每级为28 bit，分别送入*C*寄存器和*D*寄存器中。在各次迭代中，*C*和*D*寄存器分别将存数进行左循环移位置换，移位次数已规定好。每次移位后，将*C*和*D*寄存器原存数送给置换选择*PC2*。置换选择*PC2*将C中第9、18、22、25位和*D*中第7、9、15、26位删去，并将其余数字置换位置后送出48 bit数字作为第*i*次迭代时所用的子密钥k_i。

子密钥的产生



密钥表的计算逻辑

循环左移：

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

	行\列	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	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
S_2	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	2	1	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

S-Box

- 对每个盒，6比特输入中的第1和第6比特组成的二进制数确定的行，中间4位二进制数用来确定的列。中相应行、列位置的十进制数的4位二进制数表示作为输出。例如输入为101001，则行数和列数的二进制表示分别是11和0100，即第3行和第4列，的第3行和第4列的十进制数为3，用4位二进制数表示为0011，所以的输出为0011。

置换选择1（PC-1）和置换选择2（PC-2）

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

DES，算法

加密过程:运算进行16次后就得到密文组。

$L_0R_0 \leftarrow IP(\langle 64\ \text{bit 输入码} \rangle)$

$$L_i \leftarrow R_{i-1} \qquad i=1,\ldots,16$$

$$R_i \leftarrow L_{i-1} \oplus f(R_{i-1}, \boldsymbol{k}_i) \qquad i=1,\ldots,16$$

$\langle 64\ \text{bit密文} \rangle \leftarrow IP^{-1}(R_{16}L_{16})$

解密过程:DES的加密运算是可逆的，其解密过程可类似地进行。

$R_{16}L_{16} \leftarrow IP(\langle 64\ \text{bit密文} \rangle)$

$$R_{i-1} \leftarrow L_i \qquad i=16,\ldots,1$$

$$L_{i-1} \leftarrow R_i \oplus f(L_{i-1}, \boldsymbol{k}_i) \qquad i=16,\ldots,1$$

$\langle 64\ \text{bit明文} \rangle \leftarrow IP^{-1}(R_0L_0)$

DES安全性

DES的密钥量为$2^{56}=7.2\times10^{16}=$ 72 057 594 037 927 936$\approx10^{17}$个。若要对DES进行密钥搜索破译，分析者在得到一组明文-密文对条件下，可对明文用不同的密钥加密，直到得到的密文与已知的明文-密文对中的相符，就可确定所用的密钥了。密钥搜索所需的时间取决于密钥空间的大小和执行一次加密所需的时间。若假定DES加密操作需时为100μs(一般微处理器能实现)，则搜索整个密钥空间需时为7.2×10^{15}秒，近似为2.28×10^{8}年。若以最快的LSI器件，DES加密操作时间可降到5μs，也要1.1×10^{4}年才能穷尽密钥。但是由于差分和线性攻击法的出现以及计算技术的发展，按Wiener介绍，在1993年破译DES的费用为100万美元，需时3个半小时。如果将密钥加大到80 bits，采用这类搜索机找出一个密钥所需的时间约为6700年。

DES ，安全性

- 1997年1月28日，美国的RSA数据安全公司在RSA安全年会上公布了一项“秘密密钥挑战”竞赛，其中包括悬赏1万美元破译密钥长度为56比特的DES。美国克罗拉多洲的程序员Verser从1997年2月18日起，用了96天时间，在Internet上数万名志愿者的协同工作下，成功地找到了DES的密钥，赢得了悬赏的1万美元。
- 1998年7月电子前沿基金会（EFF）使用一台25万美圆的电脑在56小时内破译了56比特密钥的DES。
- 1999年1月RSA数据安全会议期间，电子前沿基金会用22小时15分钟就宣告破解了一个DES的密钥。

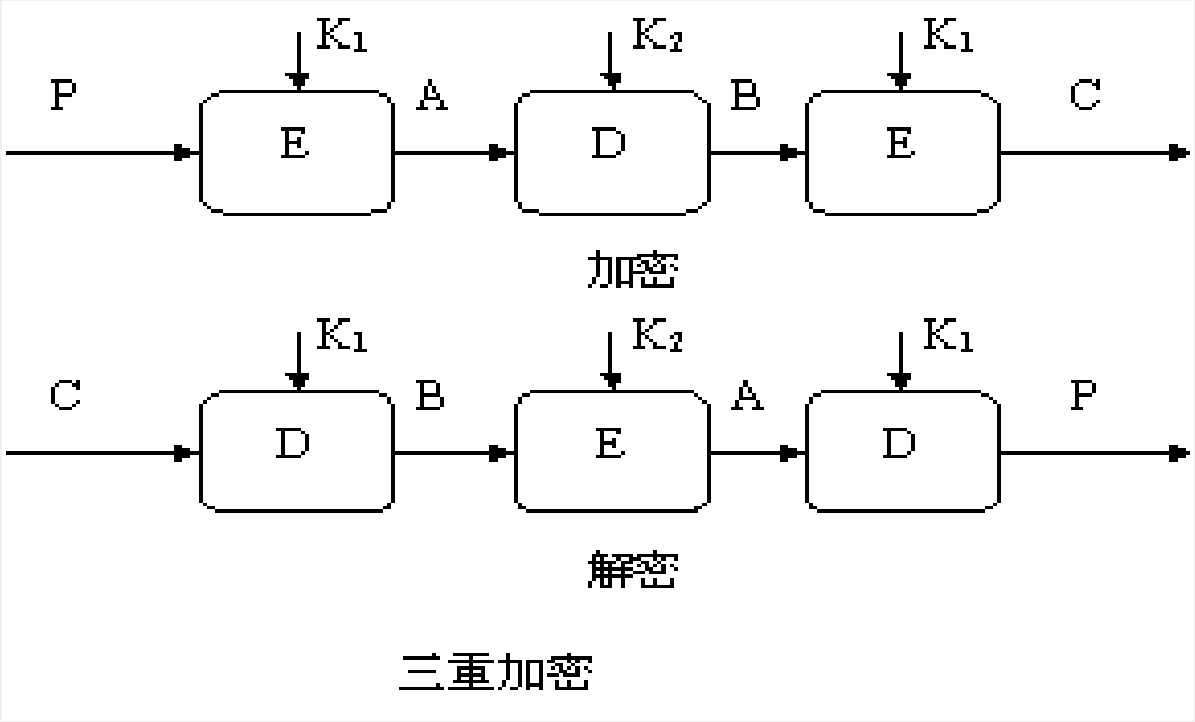
DES安全性

- 1991年Biham和Shamir的差分分析(Differential Cryptandysis)：一种选择明文攻击（如果DES只是使用8轮的话，则在个人计算机上只需要几分钟就可以破译）

- 1994年Matsui的线性分析：一种已知明文攻击方法。这种方法可用 2^21 个已知明文破译8-轮DES，可用 2^47 个已知明文破译16轮DES。

双密钥的三重DES (Triple DES with Two Keys)

- $C=E_{K1}(D_{K2}(E_{K1}(P))) \Leftrightarrow P=D_{K1}(E_{K2}(D_{K1}(C)))$



对双密钥的三重DES的分析

- 该模式由IBM设计, 可与常规加密算法兼容
- 这种替代DES的加密较为流行并且已被采纳用于密钥管理标准（The Key Manager Standards ANSX9.17和ISO8732).
- 交替使用K_1和K_2可以抵抗中间相遇攻击.如果 $C=E_{K2}(E_{K1}(E_{K1}(P)))$,只需要2^{56+2}次加密
- 到目前为止，还没有人给出攻击三重DES的有效方法。对其密钥空间中密钥进行蛮干搜索，那么由于空间太大为$2^{112}=5\times10^{33}$，这实际上是不可行的。若用差分攻击的方法，相对于单一DES来说复杂性以指数形式增长，要超过10^{52}。

随堂测试

- 掷一对无偏的骰子，若告诉你得到的总的点数为 7，请问获得了多少信息量？
- 有限域GF(9)是利用GF(3)上不可约多项式x^2+1构造的，x+1是GF(9)的一个元素，请计算出它的逆。
- 按照黑板上给出的S盒，计算随机变量$X_2 \oplus X_3 \oplus Y_2$的偏差。