分组密码算法设计

清华大学计算机系
于红波
2023年3月15日

推荐文献

- 密码学原理与实践， 第三章
- Wikipedia
  - 分组密码(Block Cipher)
    http://en.wikipedia.org/wiki/Block_cipher
  - Feistel structure
    http://en.wikipedia.org/wiki/Feistel_cipher
  - Substitution–permutation network
    https://en.wikipedia.org/wiki/Substitution%E2%80%93permutation_network
  - DES
    http://en.wikipedia.org/wiki/Data_Encryption_Standard

提纲

- 计算安全
- Feistel结构和SPN结构
- DES算法
- 多重加密-Triple-DES

One-Time Pad (OTP)

- 如何加强Vigenere密码
  - 密钥产生
    - 密钥长度与消息长度相同
    - 密钥是随机的
  - 加密
    - 每个密钥仅加密一个消息
- 上面的密码体制被称为一次一密（OTP）

One-Time Pad (OTP)

- 1917年 (World War I)
- Gilbert Vernam：AT&T Bell Labs 工程师
- Joseph Mauborgne：时为美军上尉(Captain)后为少校(Major), 曾在1914年首次发表对Playfair密码的解决方案。

一次一密（one-time pad）

- 一次一密乱码本：一个**大**的**不重复**的**真随机**密钥**字母集**被写在几张纸上并粘在一起成为一个乱码本。
- 发方：用乱码本中的每一密钥字母加密一个明文字符(明文与密钥模26加)，每个密钥仅对一个消息使用一次，加密后销毁乱码本中用过的部分。
- 收方：有一个同样的乱码本，并依次使用每个密钥去解密密文的每个字符。收方解密后也同样销毁乱码本中用过的部分。
- **新的消息**用乱码本**新的密钥**加密，不能重复使用。
- 所以叫做“one-time pad”。

One-time Pad(OTP)

- 例：

明文 H E L L O　　L A T E R

密钥 X M C K L　　T Q U R A

密文 E Q N V Z　　E Q N V Z

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

H→7, X→23, 7+23(mod 26)=4(E)

One-time Pad

- 现代的OTP处理二进制数据（bit 序列）
  - 使用“模2加”(XOR)代替“模26加”
  - (a+b) mod 2 ， a XOR b,$a \oplus b$ ， a^b
- 例：

明文　1010011000

密钥 $\oplus$ 0110101110

密文　1100110110

香农简介

- 1949年《保密系统的通信理论》
  "Communication Theory of Secrecy Systems"
- 利用信息熵在理论上**证明了一次一密是无法被破译的，同时证明了一个无法被破译的密码系统必须具备与一次一密相同的条件**，即密钥必须有以下特征：
  - 完全随机
  - 不能重复使用
  - 保密
  - 和明文一样长
- 使保密通信由艺术变成科学
- 密码设计的新思想，对现代密码体制设计非常重要。

OTP的完善保密性

- 一个密码体制的完善保密性是指已知的密文不会泄露明文的任何信息
- 定义：一个密码体制具有完善保密性， 如果对任意的明文p和任意的密文c, 都有

$$Pr(P=p|C=c)=Pr(P=p)$$

  - Pr(P=p|C=c)是已知密文c时明文p的后验概率
  - Pr(P=p)是明文p的后验概率
- 即使知道密文后，攻击者也不能以更高的概率猜测出明文

OTP的完善保密性

- One-time pad
  - $P=C=K=\{0,1\}^n$
  - K随机产生
    - $\Pr(K=k)=1/2^n$
  - 证明$\Pr(P=p|C=c)=P(P=p)$
- 证明

$$\Pr[C=c \mid P=p]=\Pr[K=p\oplus c]=1/2^n$$

$$\Pr[C=c]=\sum_{p\in P}\Pr[P=p]\Pr[C=c \mid P=p]$$

$$=1/2^n\sum_{p\in P}\Pr[P=p]=1/2^n$$

$$\therefore \Pr(P=p \mid C=c)=\frac{\Pr[C=c \mid P=p]\Pr[P=p]}{\Pr[C=c]}=\Pr[P=p]$$

密码体制的安全性

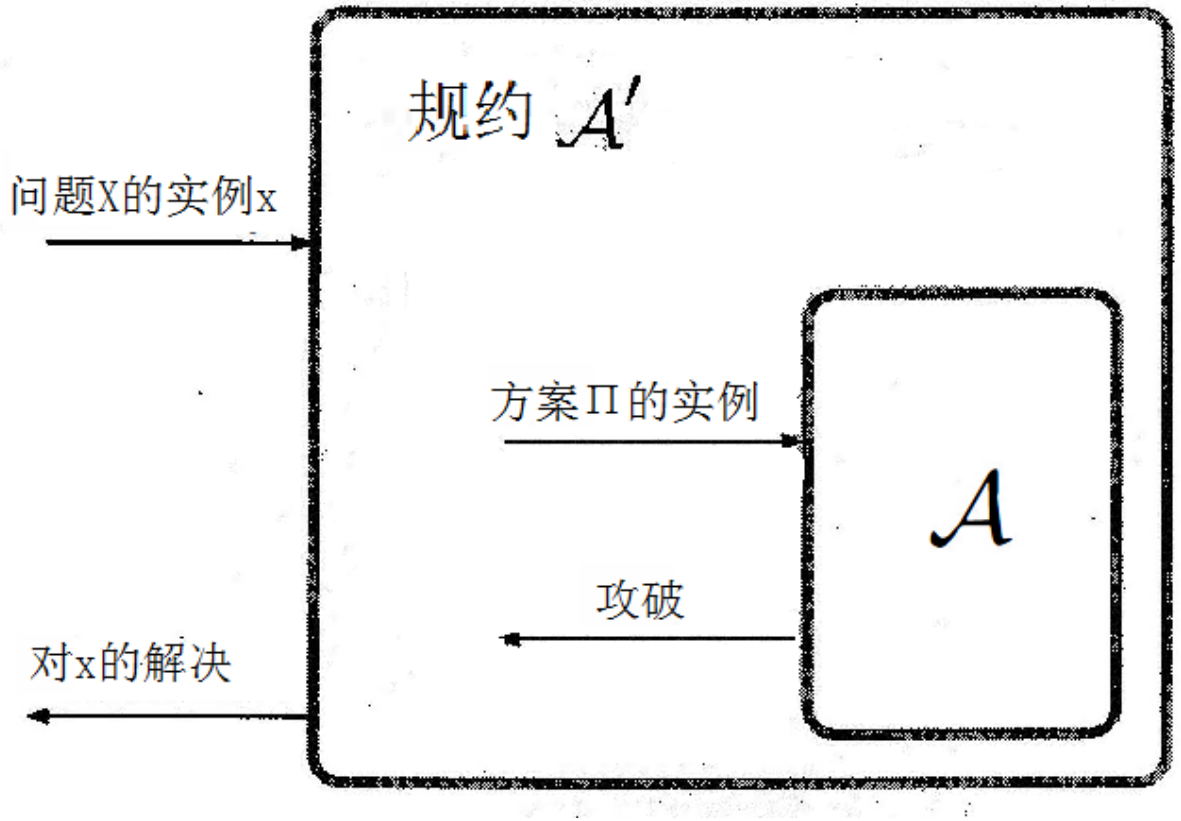
- 无条件安全性(unconditional security)即完善保密性(perfect security)
  - 即使攻击者有无限的计算资源也不可能攻破密码体制，则该密码体制是无条件安全的
  - OTP是无条件安全的
- 计算安全性(computational security)
  - 破译一个密码体制所做的计算上的努力
  - 如果使用最好的算法破译一个密码体制至少需要N次操作（N是一个特定的非常大的数字），定义该密码体制是计算安全的
- 可证明安全性(provable security)
  - 通过规约的方式为安全性提供证据
  - 如果可以破译密码体制A，则就可以解决一个数学难题B（分解因子问题，离散对数问题）

规约证明方法

- 规约证明过程
  - 指定一个有效的敌手A的攻击Π，将A的成功率$\varepsilon(n)$表示出来
  - 构造一个有效算法A′，A′将敌手A作为子程序使用，试图解决难度X
    - A′不需要知道A如何工作
    - A被A′调用时，A所见的视图和A本身和Π交互的视图相同
    - 如果A攻破A′模拟的Π的实例，那么A′给出解决难题X的实例
  - 如果$\varepsilon(n)$不可忽略，那么A′解决X的概率不可忽略，矛盾
  - 不存在有效的敌手A可以攻破Π，也就是Π计算安全

规约证明方法

- 规约证明过程



现有的计算资源

- Intel Core-i5, i7 CPUs
  - Widely used CPUs in 2021 (notebook, desktop computers)
  - Normally each costs RMB 4000-10000
  - Each has two or four cores, runs at the speed about 2.5GHz
  - Each core performs about 2^{32} 64-bit integer (or floating point) operations per second
- The most powerful supercomputers in the world in 2022：Top500，

全球超级计算机排行榜TOP500，2023年11月最新HPC top500排行榜榜单,2023/2022/2021年6月/11月超算排名榜前十名-中存储网 (chinastor.com)

计算安全

- 一种密码术如果不是数学上无法破译的，则必须在实践上无法破译
  - 实践上无法破译：使用一种不能在“合理的时间”内，以任何“合理的成功率”攻破的方案
  - 如：一个方案理论上能够被攻破，但在200年内，使用最快的可用的超级计算机，不能以大于2^{-30}的概率攻破
- 计算安全的概念包括了两种放宽了的完美安全概率
  - 仅仅在对抗“有效”的敌手时，安全性存在
  - 敌手潜在的成功率是非常低的

计算安全

- 假设一个加密方案，密钥空间K比消息M小得多，无论该方案如何构造，都会存在两种攻击
  - 给定密文c,敌手遍历所有的密钥k, 能够破译c (给出c对应着所有的可能的消息，泄露消息的一些信息）
  - 已知明文的攻击。敌手掌握了密文$c_1,c_2, \ldots c_t$, 对应着明文$m_1,m_2,\ldots,m_t$. 敌手猜测一个密钥k, 看是否对所有的i, 都有$DECk(c_i)=m_i$.
- 短密钥加密多个消息，要实现安全性
  - 限定敌手的运行时间
  - 不认为非常小的成功概率是一种攻破
- 任何加密方案中密钥空间必须足够大，以至于敌手不能够遍历

密码分析类型

- 根据敌手获取信息的能力
  - 唯密文攻击(ciphertext-only attack)
    - 敌手只能获取一定数量的密文
  - 已知明文攻击(known-plaintext attack)
    - 敌手可以获取一定数量的用相同密钥加密的明密文对
  - 选择明文攻击(chosen-plaintext attack)
    - 敌手可以选择明文，并得到对应的密文
  - 选择密文攻击(chosen-ciphertext attack)
    - 敌手不仅可以选择明文，还可以选择密文，并得到对应的明文

实际的对称密码算法

- 实际的对称加密算法需要满足两个条件
  - 计算安全
    - 攻击者知道所使用的密码体制（Kerckhoff假设）
    - 能够抵抗已知明文攻击
  - 容易使用
    - 相对短的密钥能够用来加密很多消息
- 实际的对称加密体制
  - 分组密码（Block Cipher)
  - 序列密码 （Stream Cipher)

分组密码

- 代换密码（substitution cipher）
  - 代换表太小， 26个字母
  - 不能够抵抗唯密文攻击和已知明文攻击
- 如果把代换表增加到2^{128}个元素？
  - The resulting cipher is strong!
  - 密钥太长，没法存储
    - Internation Data Corporation （IDT）估计：到2009年，全世界产生的数据是 1.2 millision Petabyte (2^{70} byte=10^{21})
    - 全世界在2018年创建、捕获、复制和消耗的数据总量为33泽字节（ZB），相当于33万亿GB。2020年，这一数字增长到59 ZB，预计到2025年将达到令人难以想象的175 ZB。1ZB是8,000,000,000,000,000,000,000比特。

分组密码算法

- 分组密码可以看成是一个巨大的“代换密码”
  - DES：2^{64}个元素的代换表
  - AES：2^{128}个元素的代换表
- 表中每个元素只被计算一次
  - 无需存储巨大的表
  - 如何计算每个元素？
    - 等价于： 如何加密一个消息分组？
      如何设计一个分组密码算法？

分组密码研究进展

- DES时代
  - 1973年：NBS(NIST)征集加密标准，IBM提交
  - 1974年：NBS第二次征集
    - IBM提交64比特密钥的算法
    - NSA加入设计，建议缩减密钥长度
  - 1976年：DES被NBS采纳为数据加密标准(FIPS 46)
  - 1983年：DES被再次确认为标准
  - 1988年：1993年：第二次，第三次确认标准
  - 1998年：DES被破译 （55小时，暴力攻击）
  - 1999年：DES第四次被确认（FIPS 46-3， Triple-DES）

分组密码研究进展（续）

- AES时代
  - 1997年，NIST公开征数据加密算法以取代DES
  - 1998年，共收到15个算法
  - 1999年，从15个中选中5个算法：MARS、RC6、Rijndael、Serpent和Twofish
  - 2000年10月：Rijndael获胜，Vincent Rijmen 和Joan Daemen
  - 2001年11月：NIST公布新的数据加密标准AES

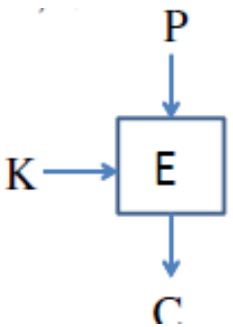
分组密码研究进展（续）

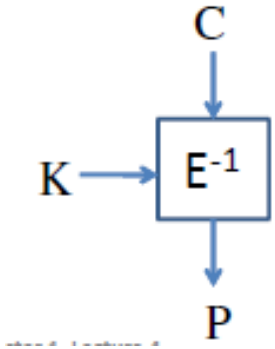
- 欧洲NESSIE工程
  - 2000年启动，共征集到17个分组密码算法
  - 2001年，进入第二轮的7个分组密码算法
  - 2003年，MSITY1、Camellia和SHACAL-2获胜
- 韩国标准：SEED和ARIA
- 我国：SMS4（SM4），用于无线局域网产品
- ISO/IET标准：2006年
  - 64位：TDEA、MISTY1和CAST-128
  - 128位：AES、Camellia、SEED
- 轻量级分组密码算法：
  - Present、CLEFIA、SIMON、SPECK

分组密码算法

- n-比特的明文分组加密成n-比特的密文分组
  - 分组长度n比特
    - DES： 64比特
    - AES： 128比特
  - 密钥长度
    - DES：56比特（不安全）
    - AES：128、192和256比特

加密：
$C = E_K(P)$

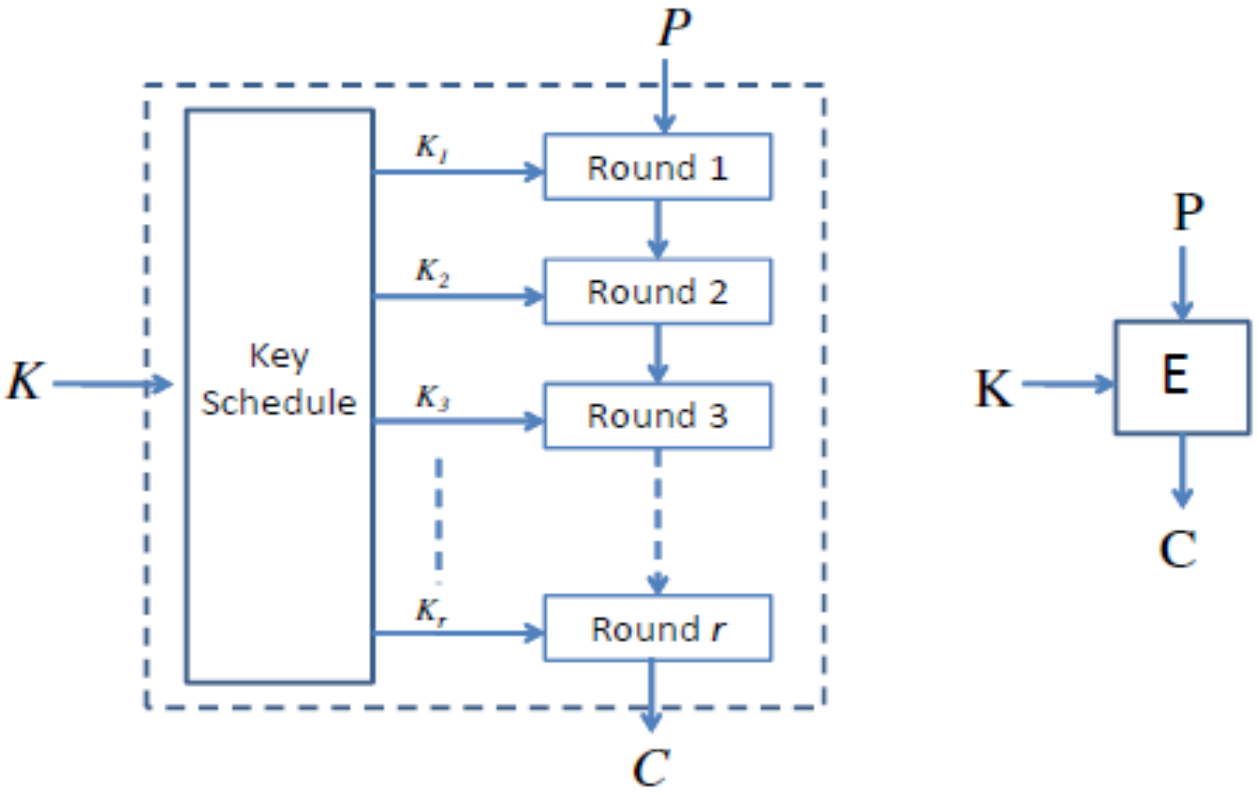




解密：
$P = E^{-1}_K(C)$

迭代结构

- 基于轮函数的迭代结构分组密码算法（乘积密码）：便于设计、安全性评估和实现



轮函数

- 轮函数的设计策略
  - 混淆(Confusion): 非线性部件
    - 小的代换表 S-box
    - 乘法与异或
    - 加与异或
    - ……
  - 扩散(Diffusion)：让所有的比特互相影响
    - 线性变换
    - 置换
    - 移位、循环移位
    - ……..

轮函数（续）

- 设计轮函数的两种方案
  - Feistel Network
    - DES
  - Substitution-Permutation Network(SPN)
    - AES
  - 其他方案

密钥方案

- 密钥生成方案（Key Schedule)
  - 多种方法
  - 目前为止没有完美的方法
  - 基本准则：密钥的每个比特应该影响不同位置的很多轮的轮子密钥

Feistel Network

- Feistel Network
  - 由Horst Feistel发明(1973年)
  - DES的设计者之一
- 该结构被广泛使用，AES最终的5个候选算法中有3个使用了Feistel网络（Mars、RC6、Twofish)
- AES (rijndael）使用SPN结构

Horst Feistel
1915-1990

Feistel Network

加密过程

$(L_0, R_0) = P$

$L_1 = R_0$

$R_1 = L_0 \oplus F(\mathrm{K}_0, R_0)$

…

$L_i = R_{i-1}$

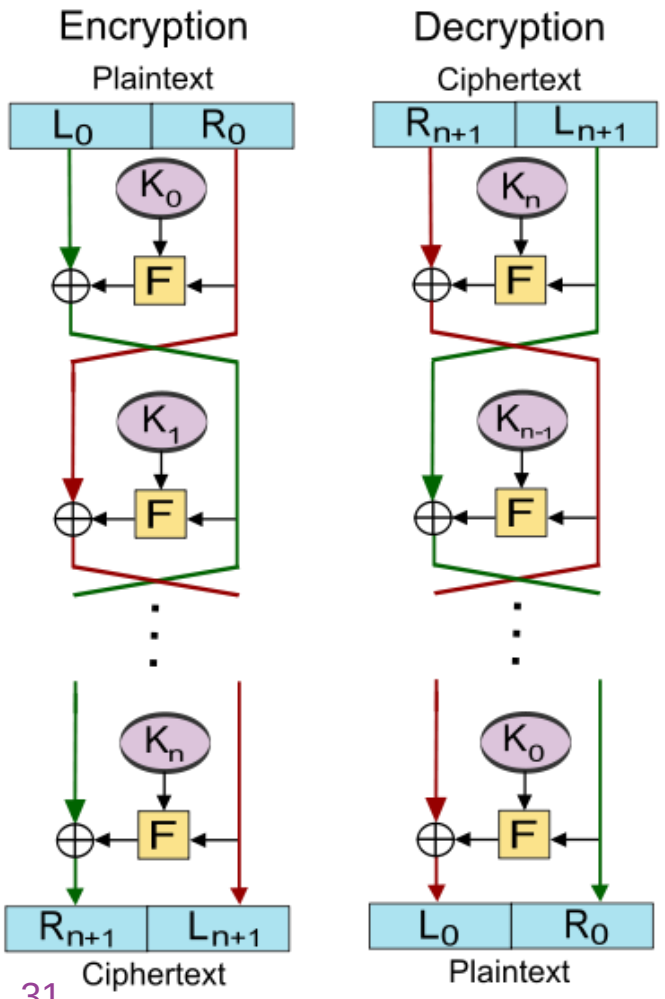
$R_i = L_{i-1} \oplus F(K_{i-1}, R_{i-1})$

…

$L_r = R_{r-1}$

$R_r = L_{r-1} \oplus F(K_{r-1}, R_{r-1})$

$C = (R_r, L_r)$



DES: Feistel Network

- 加解密过程

Encryption:

$(L_0, R_0) = P$

$L_1 = R_0$

$R_1 = F(K_1, L_0)$

...

$L_i = R_{i-1}$

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

...

$L_r = R_{r-1}$

$R_r = L_{r-1} \oplus F(K_r, R_{r-1})$

$C = (R_r, L_r)$

Decryption:

$(L_0, R_0) = C$

$L_1 = R_0$

$R_1 = F(K_r, L_0)$

...

$L_i = R_{i-1}$

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

...

$L_r = R_{r-1}$

$R_r = L_{r-1} \oplus F(K_1, R_{r-1})$

$P = (R_r, L_r)$

Feistel Network

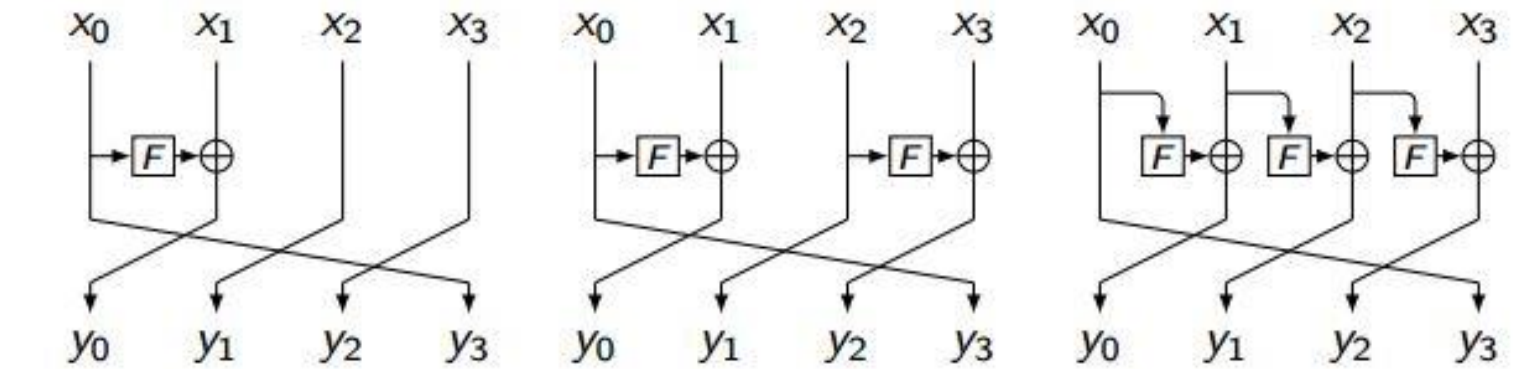
- Feistel结构的属性
  - 可逆性

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

  - 加解密使用通一套逻辑：除了密钥使用顺序不用
    - 加密：$K_1, K_2, K_3, \ldots, K_r$
    - 解密：$K_r, K_{r-1}, K_{r-2}, \ldots, K_1$

其他Feistel结构

- Unbalance Feistel Cipher
  - L_0和R_0长度不相等：如Skipjack
- Generalized Feistel networks: 多分枝



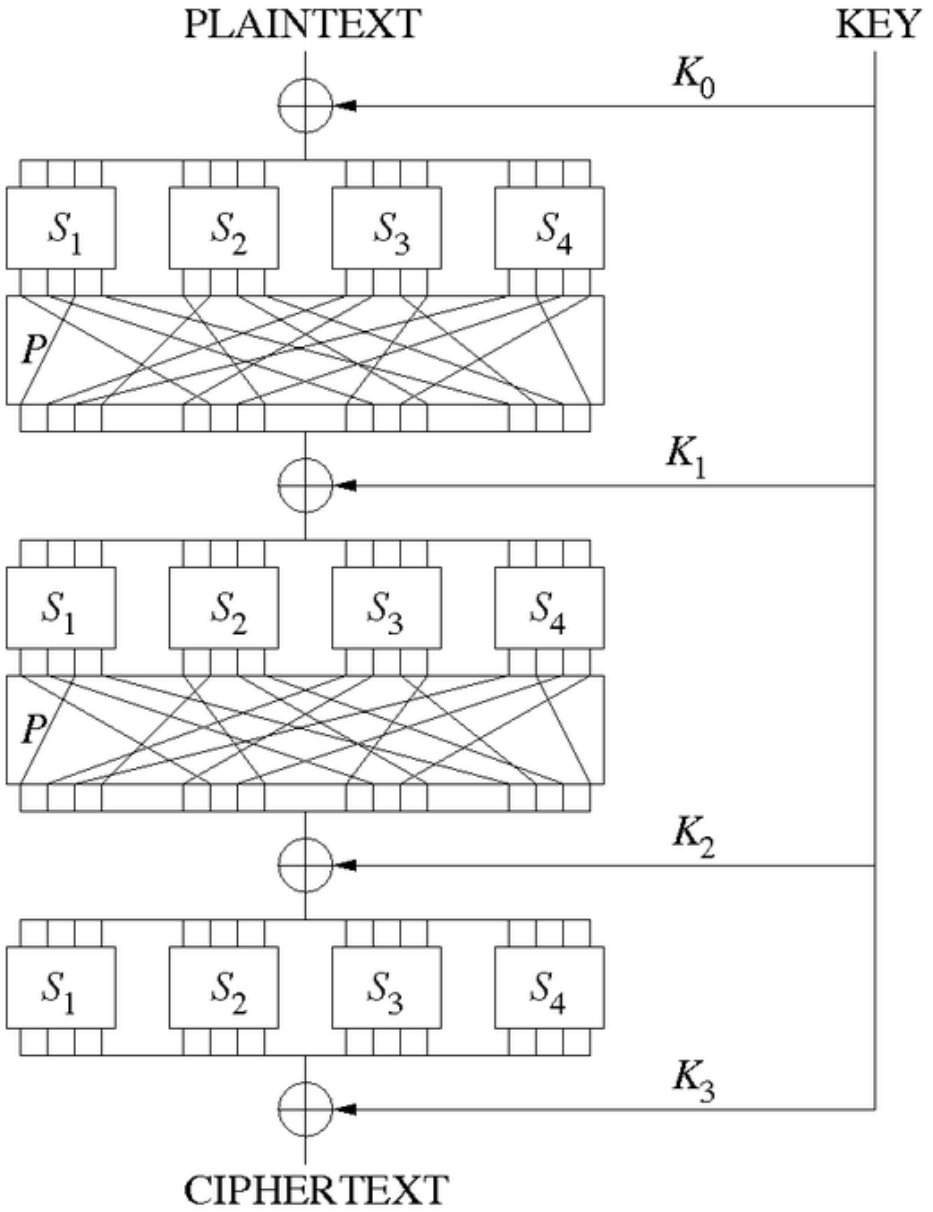
Type-1 (CAST-256, Lesamnta) Type-2 (HIGHT, CLEFIA) Type-3

Feistel结构的安全性

- Michael Luby and Charles Rackoff证明
  - 如果轮函数是安全的伪随机函数，将轮密钥S_i作为种子，则由Feistel结构构造的分组密码三轮就是一个伪随机置换，4轮是一个强的伪随机置换（即使允许敌手访问逆置换）
- 使用Feistel结构的算法
  - Feistel or modified Feistel：Blowfish,Camellia, CAST-128, DES, FEAL, GOST 28147-89, ICE, KASUMI, LOKI97,Lucifer,MARS,MAGENTA,MISTY1,RC5,Simon,TEA, Triple DES,Twofish,XTEA
  - Generlized Feistel：CAST-256, CLEFIA, MacGuffin, RC2, RC6, Skipjack, SMS4

Substitution–permutation network

- SP-network, SPN
  - AES, Kuznyechik, PRESENT, SAFER, SHARK, Square等
  - S：substitue cipher (confusion)
  - P：transpostion cipher (diffision)



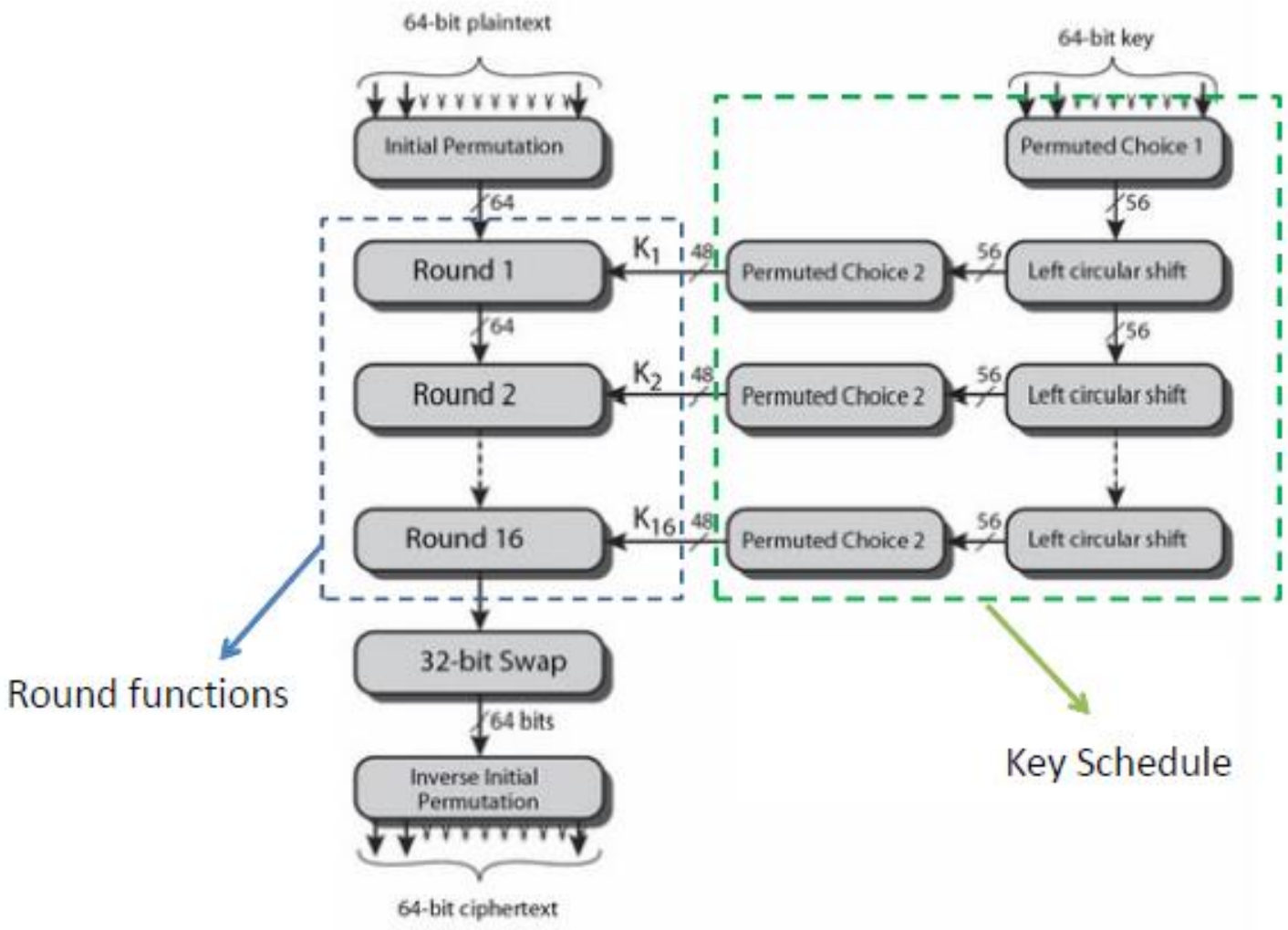
Substitution–Permutation Network

- S盒属性：
  - 改变输入的1比特，输出比特中有一半的比特改变
  - 输出的每一个比特依赖输入的所有比特
- P盒属性：
  - 好的P置换是使得S盒的输出比特分布到下一轮的尽可能多的S盒
- 扩散：扩散就是让明文中的每一位影响密文中的许多位，或者说让密文中的每一位受明文中的许多位的影响．这样可以隐蔽明文的统计特性.
- 混淆：是指让密文和密钥之间的统计关系变得复杂,使得敌手不能通过密文的统计关系,推测出密钥的统计关系 （非线性替换）

DES

- 分组加密算法
- 64比特消息分组
- 56比特密钥
  - 另加８个冗余比特用于奇偶检验
- Feistel 结构
- 共16轮　＋　起始置换　＋　末置换

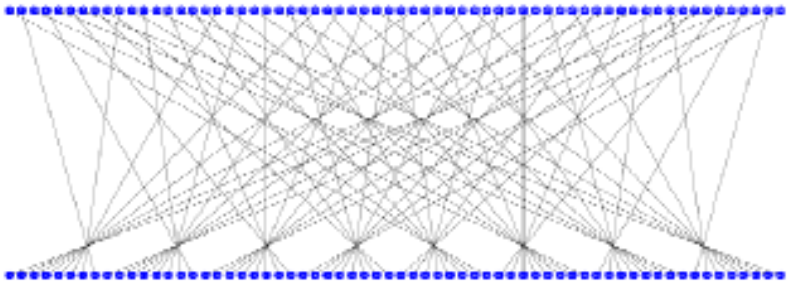
DES 总体结构

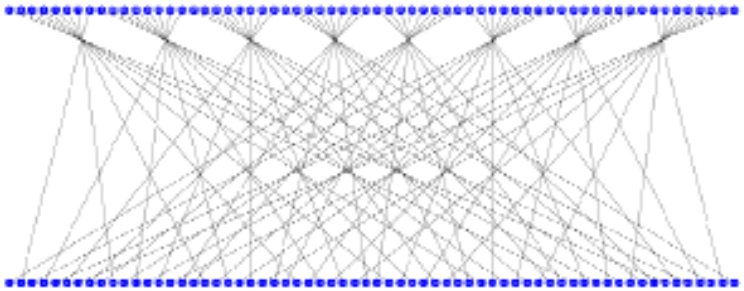

64-bit plaintext
64-bit key
Initial Permutation
Permuted Choice 1
64
56
Round 1
K1
48
Permuted Choice 2
56
Left circular shift
64
56
Round 2
K2
48
Permuted Choice 2
56
Left circular shift
Round 16
K16
48
Permuted Choice 2
56
Left circular shift
32-bit Swap
64 bits
Inverse Initial Permutation
64-bit ciphertext
Round functions
Key Schedule

起始置换和末置换

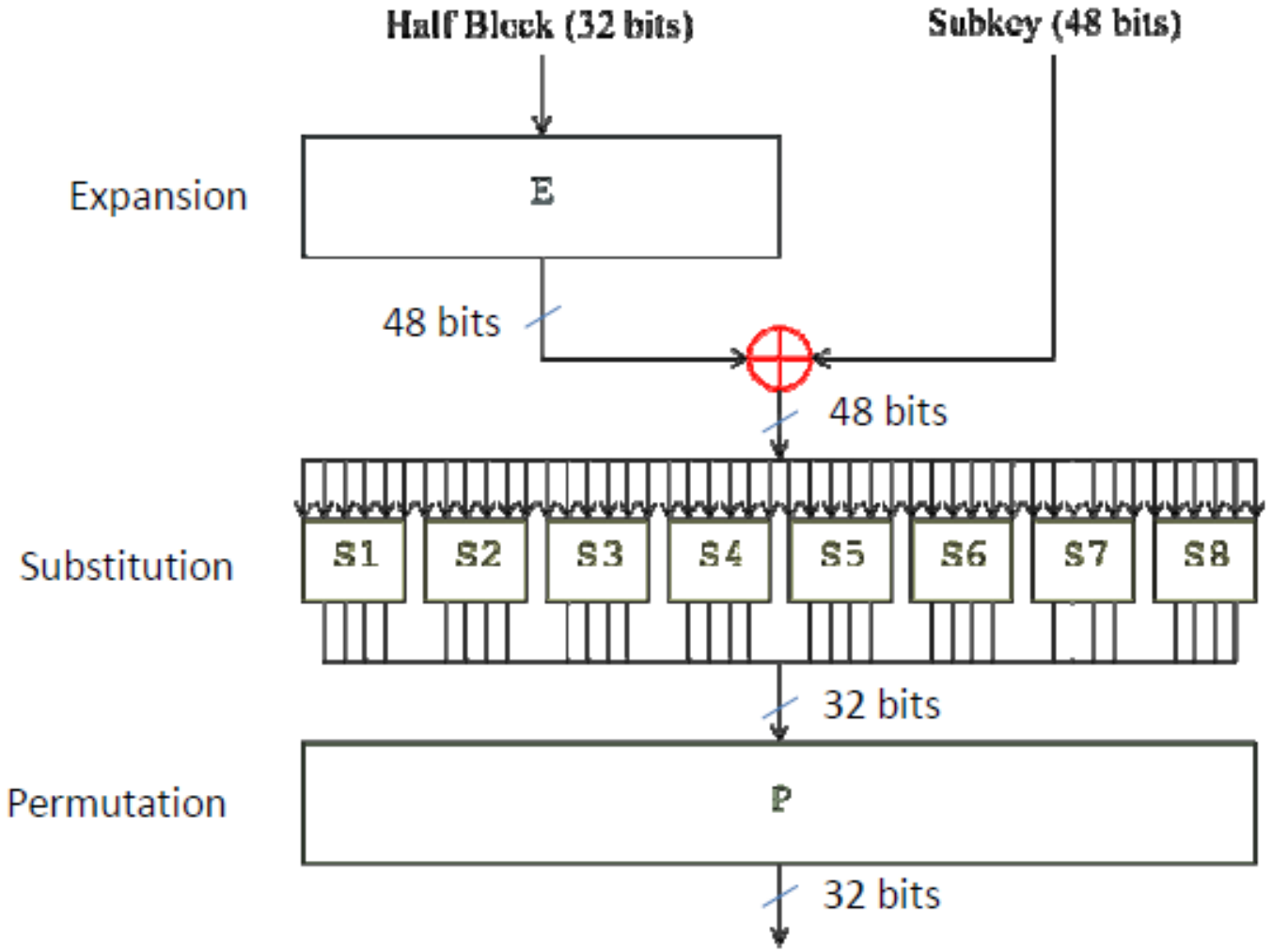
IP							
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

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





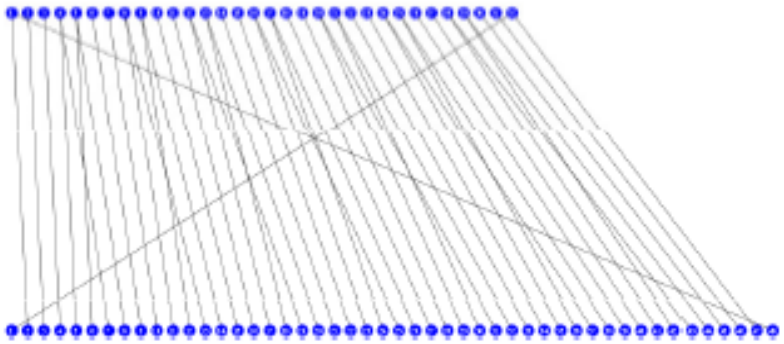
DES: F 函数

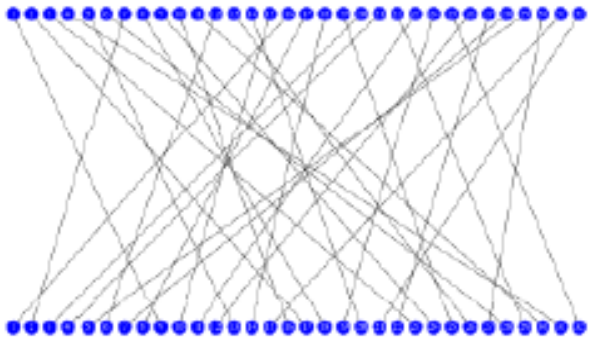

Half Block (32 bits)
Subkey (48 bits)
Expansion
E
48 bits
48 bits
Substitution
S1
S2
S3
S4
S5
S6
S7
S8
32 bits
Permutation
P
32 bits

DES: F函数中的扩展和置换

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

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: S-box

- 使用8个S盒： $S_1 S_{2\ldots\ldots} S_8$

  每一个S盒S_i :$\{0,1\}^6$->$\{0,1\}^4$

  用4乘16的矩阵描述

- 6比特的串$B=b_1b_2b_3b_4b_5b_6$
  - b_1b_6决定矩阵的行，$b_2b_3b_4b_5$决定矩阵的列
  - $C_j=S_j(B_j)$

S1盒

	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

S2盒

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

S3盒

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

S4盒

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

S5盒

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

S6盒

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

S7盒

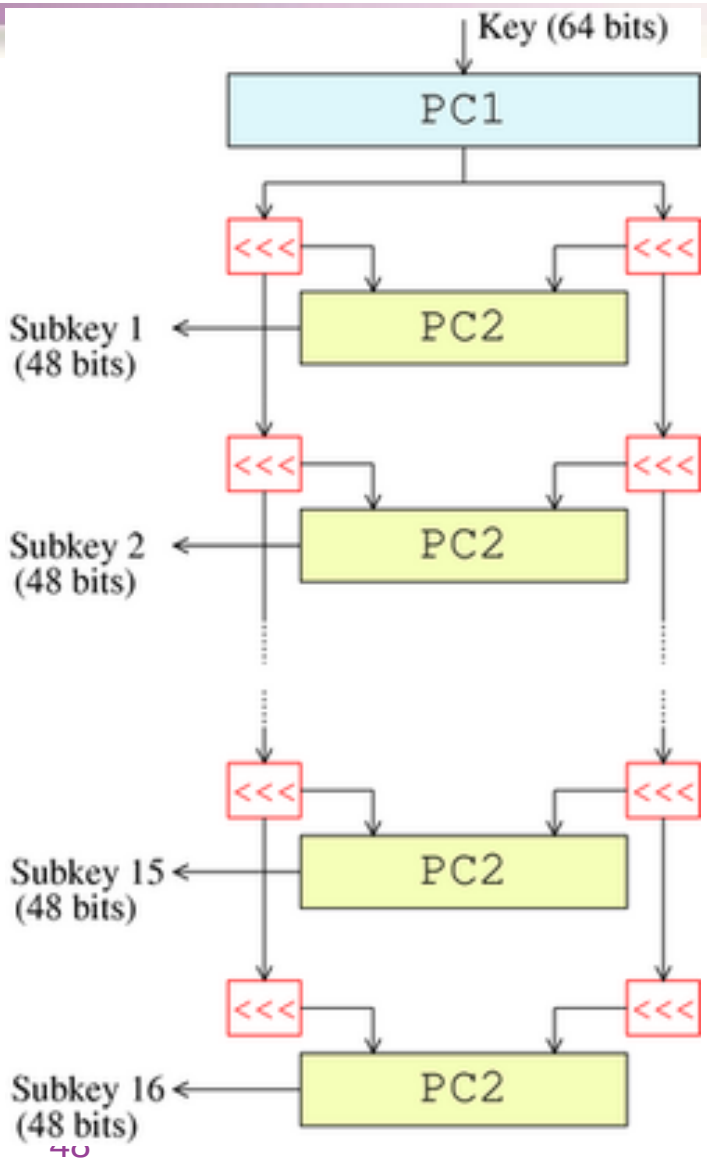
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S8盒

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

DES密钥生成算法

- 当i=1,2,9,16时，移位数位1
- 当i=3,4,5,6,7,8,10,11,12,13,14,15时，移位数为2
- PC1为一56比特的置换
- PC2从56比特中选择48比特



PC1置换表

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

PC2置换表

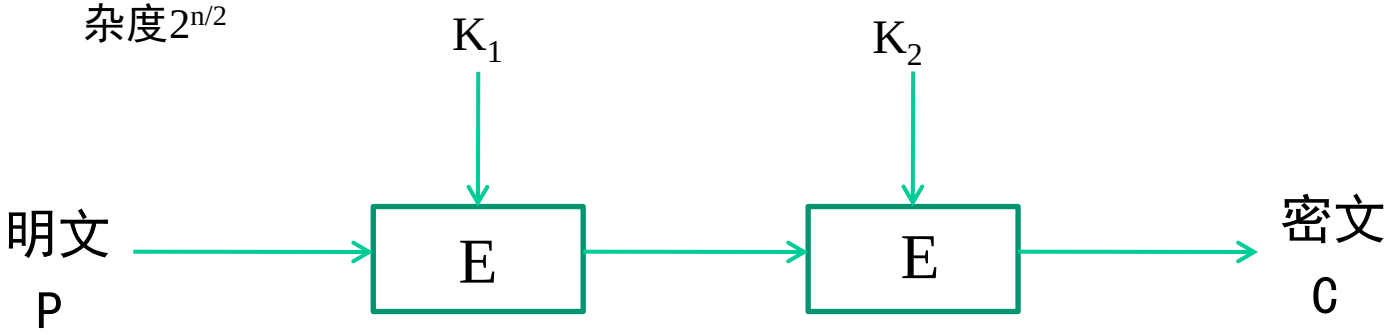
14	17	11	24	1	5	3	28	15	6	21	10
23	19	12	4	26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40	51	45	33	48
44	49	39	56	34	53	46	42	50	36	29	32

多重加密

- DES：56比特密钥长度
  - 在今天的计算能力下，不再安全
  - 如何增加密钥长度?
- 多重加密
  - Double-DES
  - Triple-DES

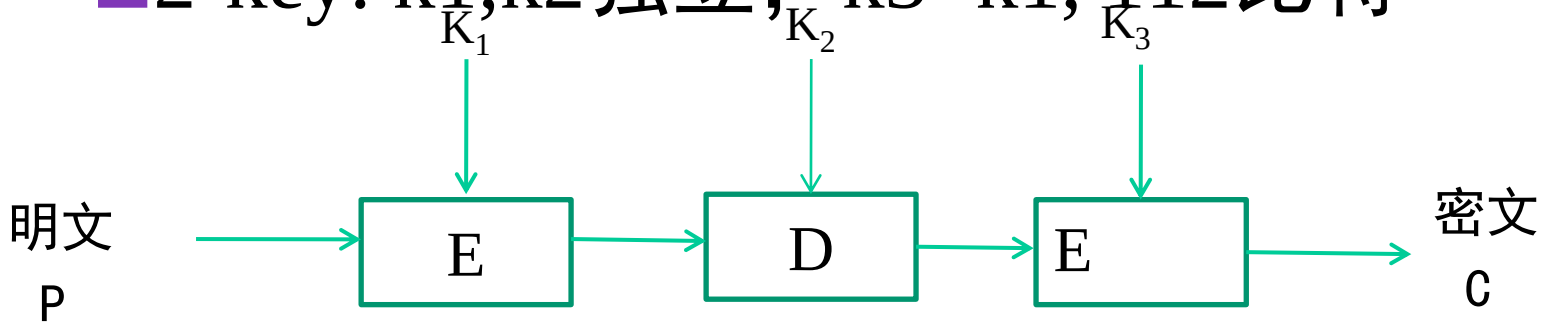
多重加密

- Double-DES
  - 密钥长度112比特
  - 加密： $C=E_{k2}(E_{k1}(P))$
  - 不能够抵抗中间相遇攻击：时间2^{57}, 存储2^{56}
    - 构造两个集合$I=\{E_{k1}(P)\}$, $J=\{D_{k2}(C)\}$, 利用生日攻击寻找碰撞， 时间、空间复杂度$2^{n/2}$



多重加密

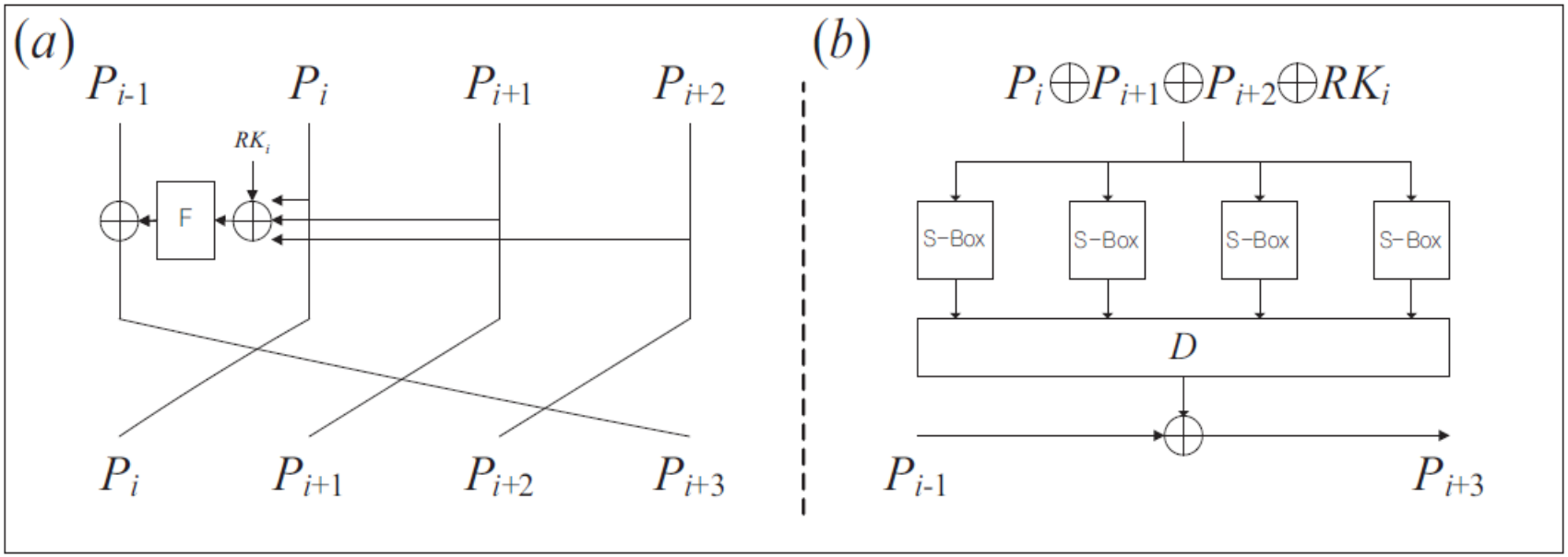
- Triple-DES
  - 加密： $C=E_{k3}(D_{k2}(E_{k1}(P)))$
  - 解密： $P=D_{k1}(E_{k2}(D_{k3}(C)))$
- 密钥选择
  - 3-key: k1,k2,k3 互相独立，168比特
  - 2-key: k1,k2独立，k3=k1, 112比特



SM4算法

- 我国WAPI无线网络标准中使用的加密算法
- 2012年国密标准，2016年国家标准
- 32轮，迭代的非平衡Feistel结构
- 密钥长度和分组长度都是128比特

SM4算法-轮函数



$$P_{i+3} = P_{i-1} \oplus D(S(P_i \oplus P_{i+1} \oplus P_{i+2} \oplus RK_i))$$

$$D(x) = x \oplus (x <<< 2) \oplus (x <<< 10) \oplus (x <<< 18) \oplus (x <<< 24)$$

SM4算法-S盒

	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xa	0xb	0xc	0xd	0xe	0xf
0x0	d6	90	e9	fe	cc	e1	3d	b7	16	b6	14	c2	28	fb	2c	05
0x1	2b	67	9a	76	2a	be	04	c3	aa	44	13	26	49	86	06	99
0x2	9c	42	50	f4	91	ef	98	7a	33	54	0b	43	ed	cf	ac	62
0x3	e4	b3	1c	a9	c9	08	e8	95	80	df	94	fa	75	8f	3f	a6
0x4	47	07	a7	fc	f3	73	17	ba	83	59	3c	19	e6	85	4f	a8
0x5	68	6b	81	b2	71	64	da	8b	f8	eb	0f	4b	70	56	9d	35
0x6	1e	24	0e	5e	63	58	d1	a2	25	22	7c	3b	01	21	78	87
0x7	d4	00	46	57	9f	d3	27	52	4c	36	02	e7	a0	c4	c8	9e
0x8	ea	bf	8a	de	40	c7	38	b5	a3	f7	f2	ce	f9	61	15	a1
0x9	e0	ae	5d	a4	9b	34	1a	55	ad	93	32	30	f5	8c	b1	e3
0xa	1d	f6	e2	2e	82	66	ca	60	c0	29	23	ab	0d	53	4e	6f
0xb	d5	db	37	45	de	fd	8e	2f	03	ff	6a	72	6d	6c	5b	51
0xc	8d	1b	af	92	bb	dd	bc	7f	11	d9	5c	41	1f	10	5a	d8
0xd	0a	c1	31	88	a5	cd	7b	bd	2d	74	d0	12	b8	e5	b4	b0
0xe	89	69	97	4a	0c	96	77	7e	65	b9	f1	09	c5	6e	c6	84
0xf	18	f0	7d	ec	3a	dc	4d	20	79	ee	5f	3e	d7	cb	39	48

例：S(0x01)=0x90

SM4算法-密钥扩展算法

- 同轮函数相似
- 128比特的主密钥(MK_0,MK_1,MK_2,MK_3)

$$(K_0, K_1, K_2, K_3) = (MK_0 \oplus T_0, MK_1 \oplus T_1, MK_2 \oplus T_2, MK_3 \oplus T_3,)$$

$$RK_j = K_{j+3} = K_{j-1} \oplus D'(S(K_j \oplus K_{j+1} \oplus K_{j+2} \oplus CK_j)$$

$$D'(x) = x \oplus (x <<< 13) \oplus (x <<< 23)$$

$$CK_j = ((28 \bullet j), (28 \bullet j + 7), (28 \bullet j + 14), (28 \bullet j + 21))$$

$$T_0 = 0xa3b1bac6, T_1 = 0x56aa3350, T_2 = 0x677d9197, T_3 = 0xb27022dc.$$

谢谢！