
西安電子科技大學
XIDIAN UNIVERSITY

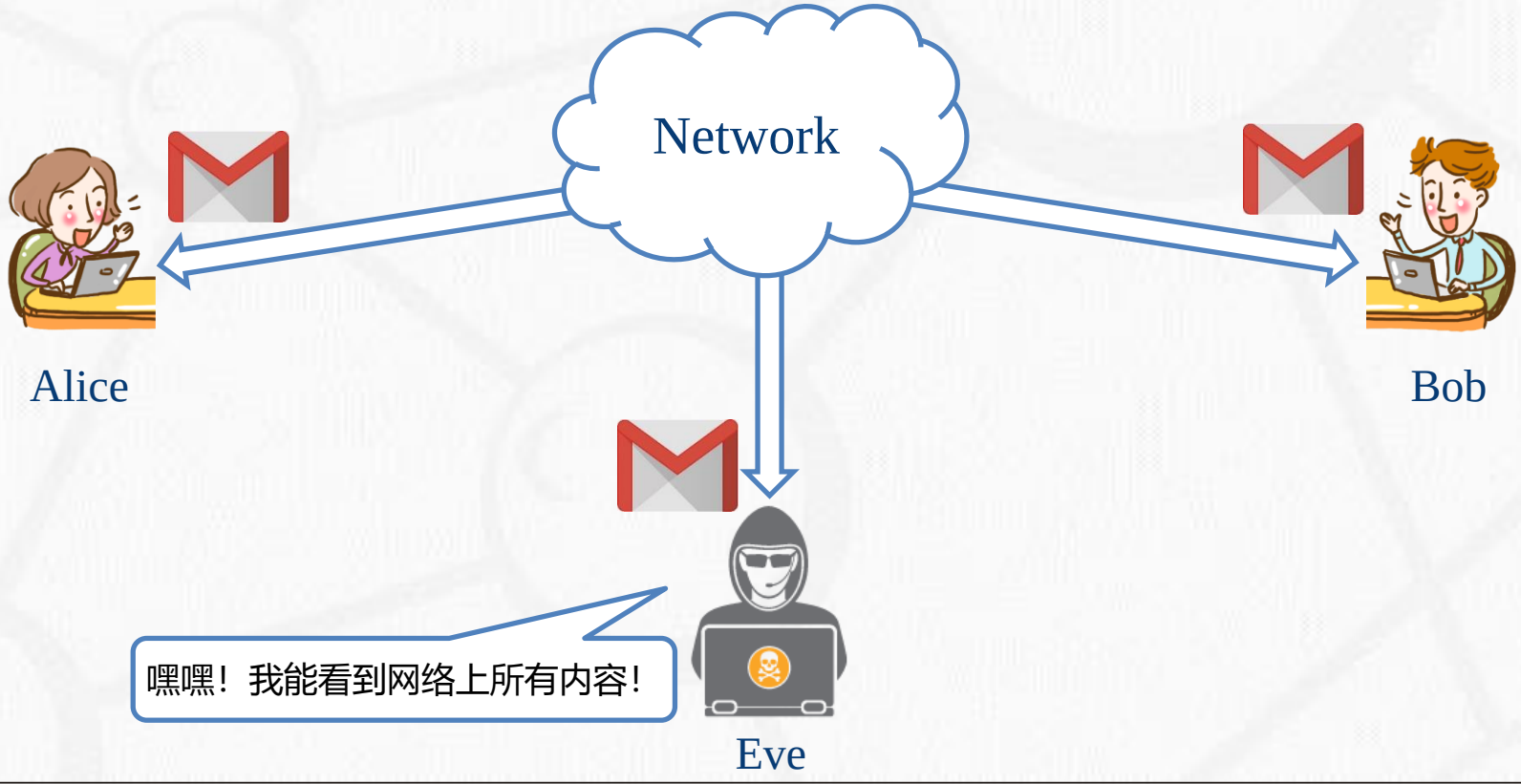
分组、扩散、混淆

# §2.1.3 对称密码算法

# §2.1.3 对称密码算法 - 引子(1)

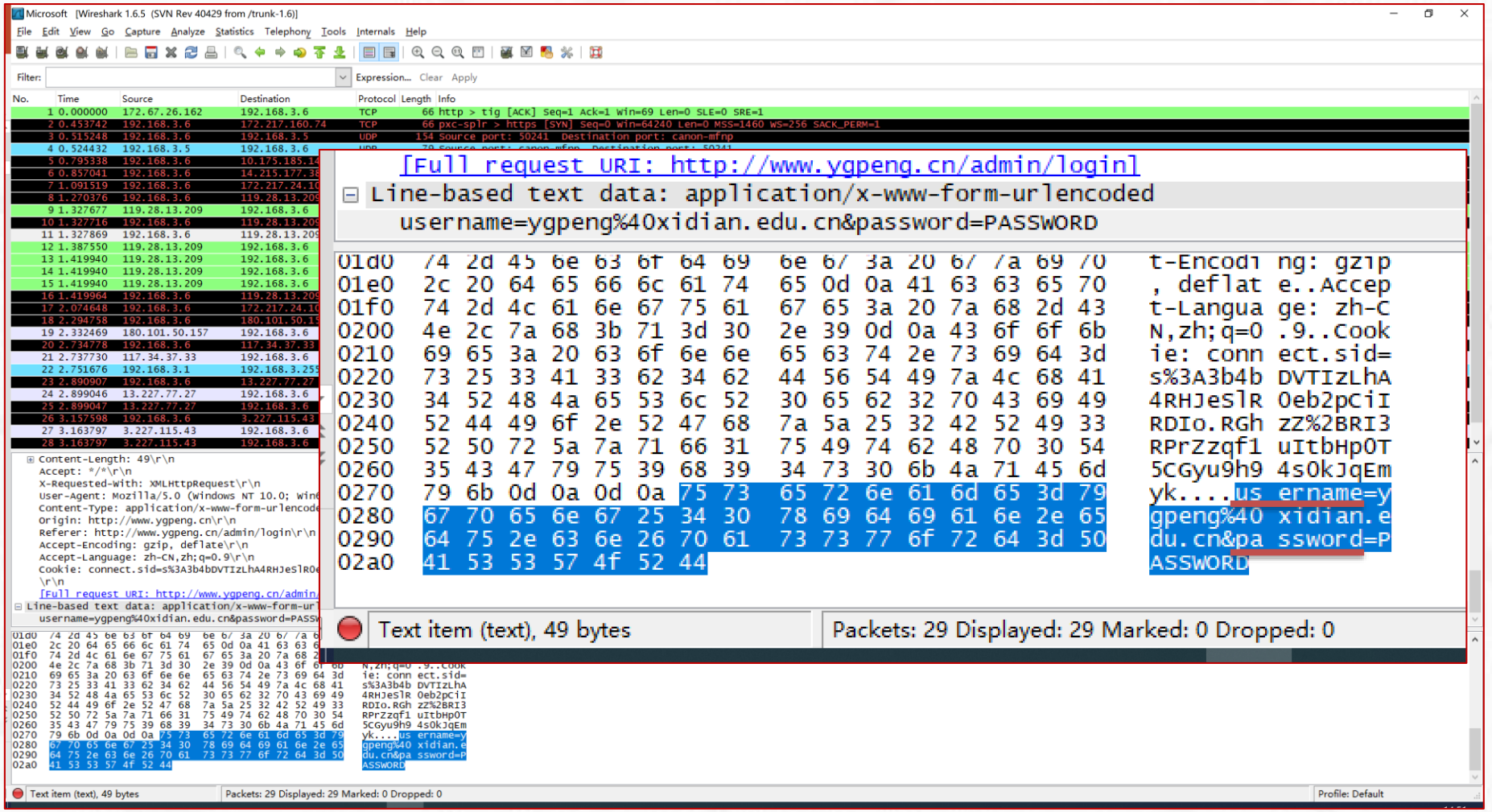
## 我们从审视一下这个故事

- Alice有一条消息$m$(存放在邮件中)需要通过公开网络发送给Bob，并且其他人都得不到。



# §2.1.3 对称密码算法 - 互联网监听的例子

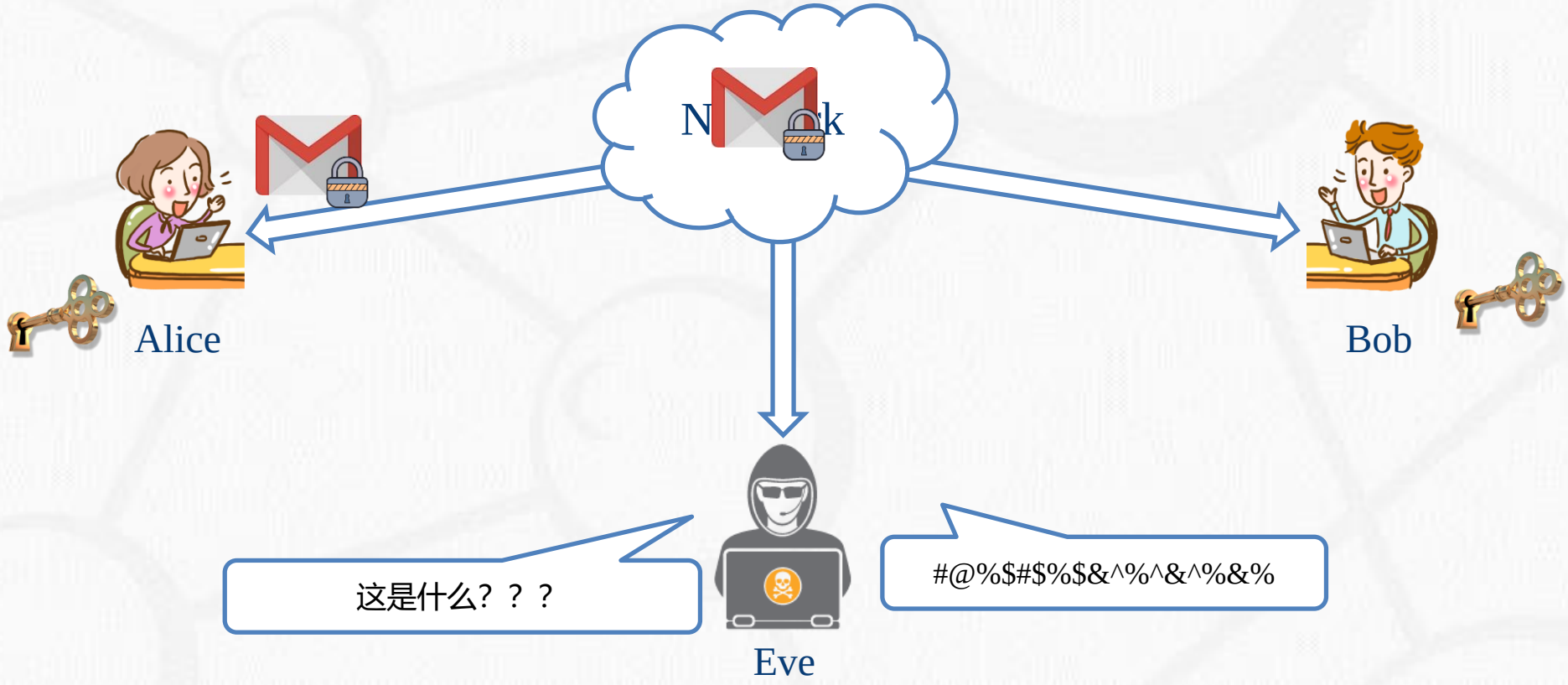
- 通过Wireshark监听得到的内容：



# §2.1.3 对称密码算法 - 引子(2)

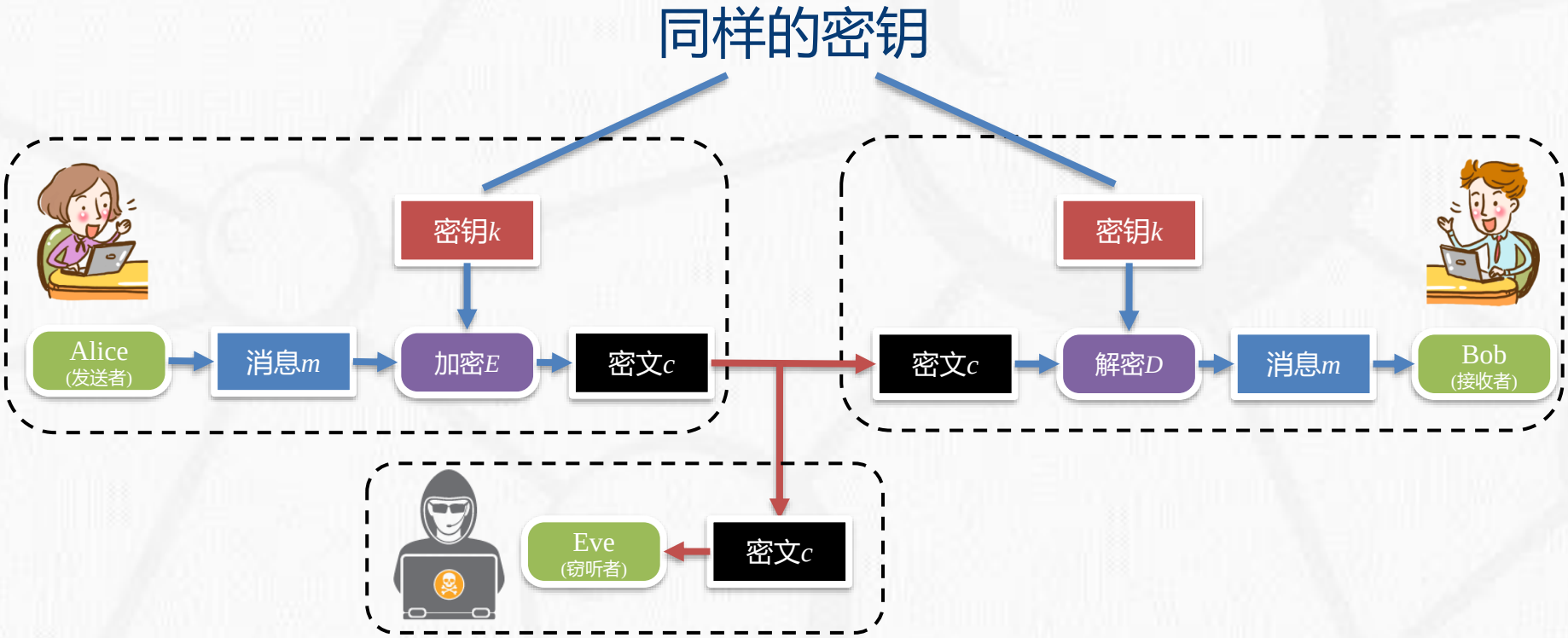
## 我们从审视一下这个故事

- Alice有一条消息$m$(存放在邮件中)需要通过公开网络发送给Bob，并且其他人都得不到。



# §2.1.3 对称密码算法 - 模型

- 对称加密模型：



# §2.1.3 对称密码算法 - 如何加密?

- 从文字到比特序列
  - 编码(Encoding)：将现实世界中的东西映射为比特序列的操作成为编码。
    - 例：ASCII码
    - $C \rightarrow 0100\ 0011$
    - $r \rightarrow 0111\ 0010$
    - $y \rightarrow 0111\ 1001$
    - $p \rightarrow 0111\ 0000$
    - $t \rightarrow 0111\ 0100$
    - $o \rightarrow 0110\ 1111$

| ASCII 值 | 控制字符 | ASCII 值 | 控制字符 | ASCII 值 | 控制字符 | ASCII 值 | 控制字符 |
|---|---|---|---|---|---|---|---|
| 0 | NUT | 32 | (space) | 64 | @ | 96 | 、 |
| 1 | SOH | 33 | ! | 65 | A | 97 | a |
| 2 | STX | 34 | ” | 66 | B | 98 | b |
| 3 | ETX | 35 | # | 67 | C | 99 | c |
| 4 | EOT | 36 | $ | 68 | D | 100 | d |
| 5 | ENQ | 37 | % | 69 | E | 101 | e |
| 6 | ACK | 38 | & | 70 | F | 102 | f |
| 7 | BEL | 39 | , | 71 | G | 103 | g |
| 8 | BS | 40 | ( | 72 | H | 104 | h |
| 9 | HT | 41 | ) | 73 | I | 105 | i |
| 10 | LF | 42 | * | 74 | J | 106 | j |
| 11 | VT | 43 | + | 75 | K | 107 | k |
| 12 | FF | 44 | , | 76 | L | 108 | l |
| 13 | CR | 45 | - | 77 | M | 109 | m |
| 14 | SO | 46 | . | 78 | N | 110 | n |
| 15 | SI | 47 | / | 79 | O | 111 | o |
| 16 | DLE | 48 | 0 | 80 | P | 112 | p |
| 17 | DCI | 49 | 1 | 81 | Q | 113 | q |
| 18 | DC2 | 50 | 2 | 82 | R | 114 | r |
| 19 | DC3 | 51 | 3 | 83 | X | 115 | s |
| 20 | DC4 | 52 | 4 | 84 | T | 116 | t |
| 21 | NAK | 53 | 5 | 85 | U | 117 | u |
| 22 | SYN | 54 | 6 | 86 | V | 118 | v |
| 23 | TB | 55 | 7 | 87 | W | 119 | w |
| 24 | CAN | 56 | 8 | 88 | X | 120 | x |
| 25 | EM | 57 | 9 | 89 | Y | 121 | y |
| 26 | SUB | 58 | : | 90 | Z | 122 | z |
| 27 | ESC | 59 | ; | 91 | [ | 123 | { |
| 28 | FS | 60 | < | 92 | / | 124 | \| |
| 29 | GS | 61 | = | 93 | ] | 125 | } |
| 30 | RS | 62 | > | 94 | ^ | 126 | ~ |
| 31 | US | 63 | ? | 95 | — | 127 | DEL |

部分ASCII码表

文本：Crypto

查表 映射

明文：编码值(数值)

# §2.1.3 对称密码算法 - 一个简单的加密

- 异或(XOR, exclusive)：计算机内数值的一个基本操作。
  - 两次异或，结果还原。
    - $b \oplus b' \oplus b' = b$

0 XOR 1 = 1

1 XOR 1 = 0

0 XOR 0 = 0
0 XOR 1 = 1
1 XOR 0 = 1
1 XOR 1 = 0

异或操作规则

  - 对于比特序列，“两次异或，结果还原”同样成立。
    - $A \oplus B \oplus B = A$
    - $A: 0100\ 1100,\quad B: 1010\ 1010$
    - $C = A \oplus B = 1110\ 0110,\quad C \oplus B = 01001100 = A$

# §2.1.3 对称密码算法 - 一次一密

- 一次一密(one-time pad)：又称一次性密码本。
  - 绝对不会被破译的密码。
  - 原理：将明文与一串等长的随机比特序列进行异或运算。

| | | | | | | |
|---|---|---|---|---|---|---|
| 明文: | 0100 0011 | 0111 0010 | 0111 1001 | 0111 0000 | 0111 0100 | 0110 1111 |
| 密钥: | 0101 0011 | 0110 0101 | 0110 0011 | 0111 0010 | 0110 0101 | 0111 0100 |
| 密文: | 0001 0000 | 0001 0111 | 0001 1010 | 0000 0010 | 0001 0001 | 0001 1011 |

  - 解密：

| | | | | | | |
|---|---|---|---|---|---|---|
| 密文: | 0001 0000 | 0001 0111 | 0001 1010 | 0000 0010 | 0001 0001 | 0001 1011 |
| 密钥: | 0101 0011 | 0110 0101 | 0110 0011 | 0111 0010 | 0110 0101 | 0111 0100 |
| 明文: | 0100 0011 | 0111 0010 | 0111 1001 | 0111 0000 | 0111 0100 | 0110 1111 |

  - 若破解，需要尝试$n \times 2^8$个密钥进行解密。
  - 1949年由Shannon给出理论证明：无条件安全(unconditionally secure)。

# 主观题 1分

设置

XIDIAN UNIVERSITY

一次一密方案在实际应用中存在什么问题?

正常使用主观题需2.0以上版本雨课堂

作答

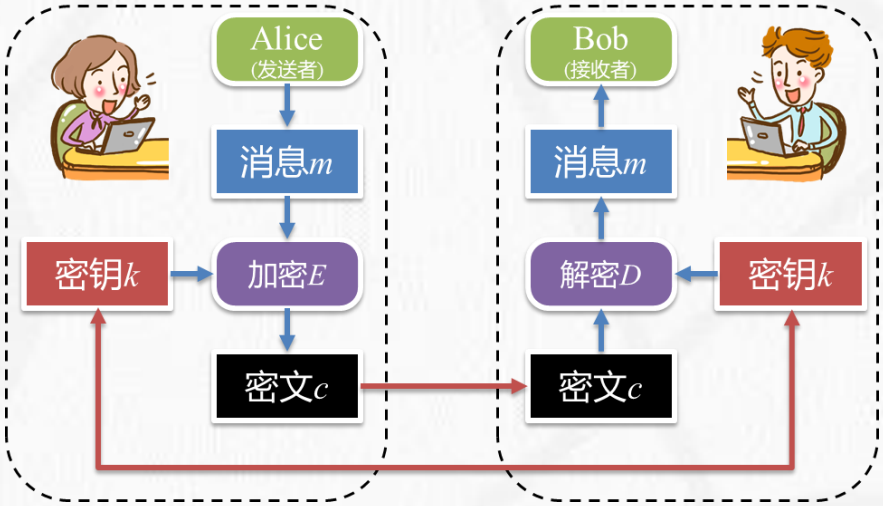
# §2.1.3 对称密码算法 - 一次一密的缺陷

- 一次一密的显著缺陷：
  - 长密钥：密钥的长度与明文长度一致；
  - 难管理：密钥难以管理。

**一次一密并不实用！**

# §2.1.3 对称密码算法 - 定义

- 对称加密算法(Symmetric encryption algorithm)又称为私钥加密密码：
  - 加密密钥能够从解密密钥推算出来，反过来也成立。
  - 在大多数对称算法中，加密/解密的密钥是相同的。
  - 这些算法也叫做秘密密钥算法或单钥算法，它要求发送者和接收者在安全通信之前，商定一个密钥。
  - 对称算法的安全性依赖于密钥，泄漏密钥就意味着任何人都能对消息进行加/解密。
  - 只要通信需要保密，密钥就必须保密。



# §2.1.3 对称密码算法 - DES的由来(1)

- 数据加密标准(Data Encryption Standard)


National Bureau of Standards

  - DES是全世界首个通用的标准算法。
  - 20世纪70年代初期，非军事性的密码处于一种无序状态。1972年，美国国家标准局NBS开始一项保护计算机和通信数据的项目，其中一部分是要开发一个单独的标准保密算法。
  - 1973年5月15日，NBS公开征集标准加密算法，并公布了设计要求，未果。
  - 1974年8月，NBS第二次征集算法，收到一份可选方案：基于IBM在20世纪70年代初开发的LUCIFER算法的算法。

# §2.1.3 对称密码算法 - DES的由来(2)

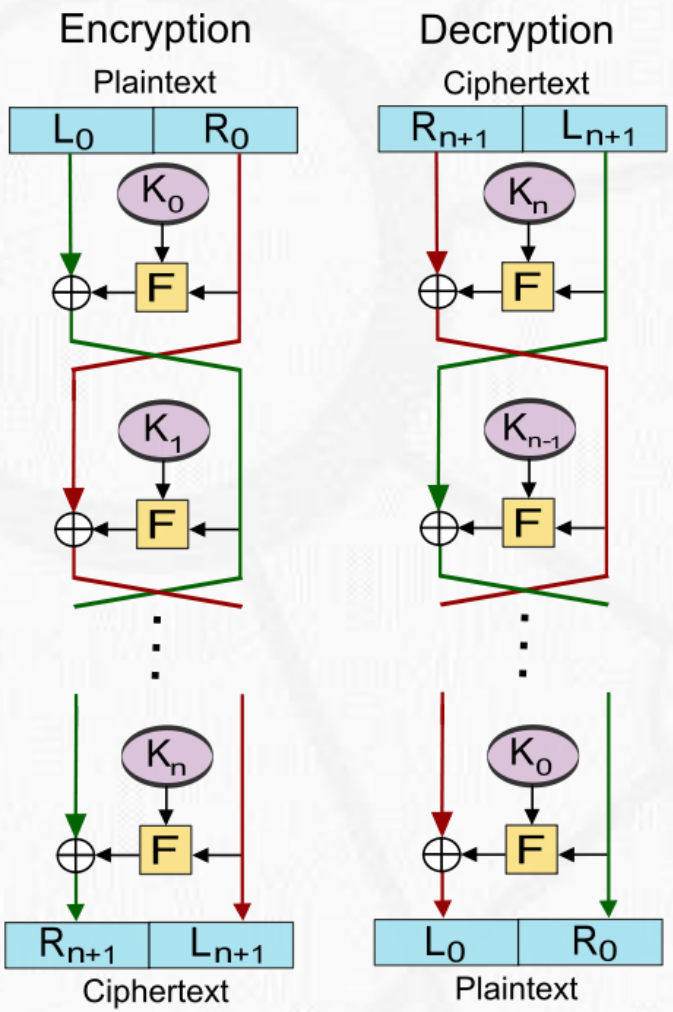
- 数据加密标准(Data Encryption Standard)
  - 1976年，NBS指派两个工作组来评价该标准。
  - DES在1976年11月23日被宣布为联邦标准，允许在非保密的政府通信中使用。标准的官方描述在1977年1月15日颁布，6个月后生效。
  - 1984年9美国总统签署145号国家安全决策令(NSDD)，命令NSA着手发展新的加密标准，用于政府系统非机密数据和私人企事业单位。
  - NSA宣布每隔5年重新审议DES是否继续作为联邦标准，1988年(FIPS46-1)，1993年(FIPS46-2)，1998年不再重新批准DES为联邦标准。

# §2.1.3 对称密码算法 - NBS公开征集标准加密算法的要求

- 算法必须提供高度的安全性
- 算法必须有详细的说明，并易于理解
- 算法的安全性必须取决于密钥，不依赖于算法本身的保密
- 算法必须必须对所有用户都适用
- 算法必须适用于各种应用
- 算法必须能用电子设备经济地实现
- 算法必须使用效率高
- 算法必须能被证实有效
- 算法必须是可移植的

# §2.1.3 对称密码算法 - 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密码结构，由此可获得代换密码。
  - 它可以看作是diffusion(扩散)和confusion(混乱)的具体实现，很多分组密码方案本质上都是基于Feistel结构。
  - 明文分组分为：$L_0$和$R_0$，数据的这两部分通过$n$次循环处理后，再结合起来生成密文分组
  - 每$i$次循环都以上一循环产生的$L_{i-1}$和$R_{i-1}$和$K$产生的子密钥$K_i$作为输入。



# §2.1.3 对称密码算法 - 扩散与混淆

- 扩散和混淆是由Shannon提出的设计密码系统的两个基本方法，目的是抗击敌手对密码系统的统计分析。
  - 如果敌手知道明文的某些统计特性，如消息中不同字母出现的频率、可能出现的特定单词或短语，而且这些统计特性以某种方式在密文中反映出来，那么敌手就有可能得出加密密钥或其一部分，或者得出包含加密密钥的一个可能的密钥集合。
- 在Shannon称之为理想密码的密码系统中，密文的所有统计特性都与所使用的密钥独立。
- 扩散和混淆成功地实现了对称密码的本质属性，因而成为设计现代分组密码的基础。

# §2.1.3 对称密码算法 - 扩散

- 扩散：就是将明文的统计特性散布到密文中去，实现方式是使得明文的每一位影响密文中多位的值。
  - 使明文和密文之间的统计关系尽可能复杂，以使敌手无法得到密钥。
- 例：对英文消息$M = m_1 m_2 \ldots m_n$的加密操作

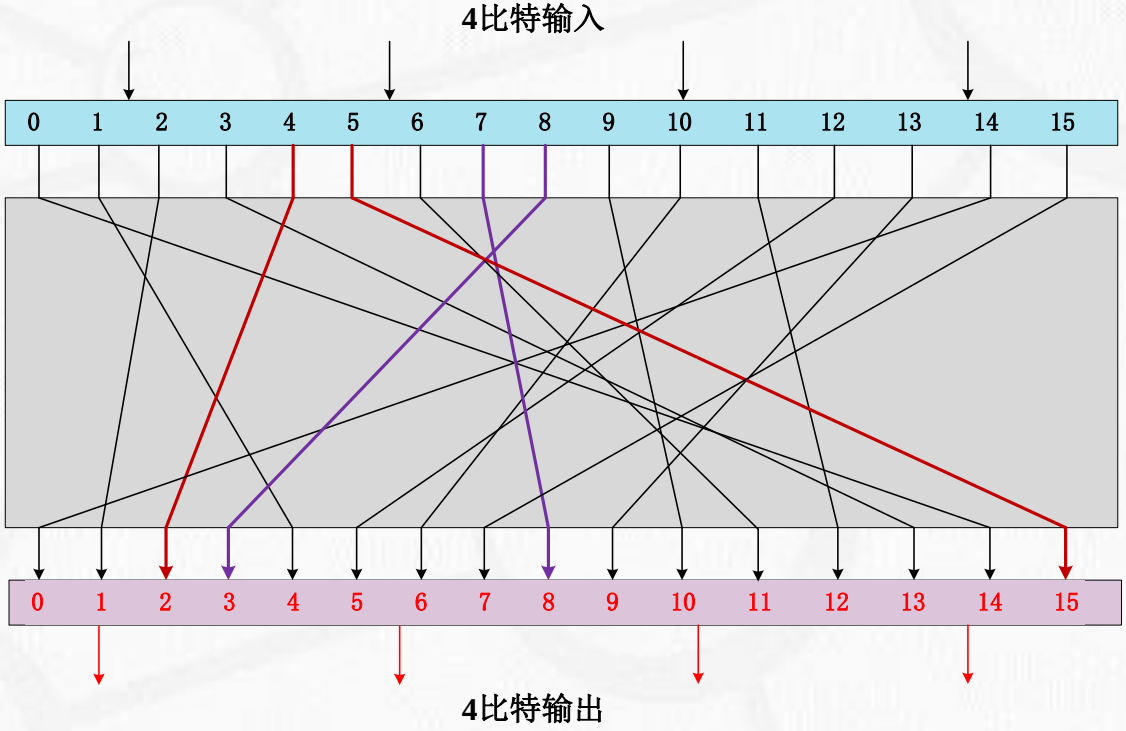
$$y_n = chr\left(\sum_{i=1}^{k} ord(m_{n+i}) (\mathrm{mod} 26)\right)$$

$k$个明文影响一个密文！！！

  - 其中$ord(m_i)$是求字母$m_i$对应的序号，$chr(i)$是求序号$i$对应的字母。
  - 这时明文的统计特性将被散布到密文中，因而每一字母在密文中出现的频率比在明文中出现的频率更接近于相等，双字母及多字母出现的频率也更接近于相等。

# §2.1.3 对称密码算法 - 混淆

- 混淆：使密文和密钥之间的统计关系尽可能复杂，以使敌手无法得到密钥。
  - 使用复杂的代换算法可以得到预期的混淆效果，而简单的线性代换函数得到的混淆效果则不够理想。

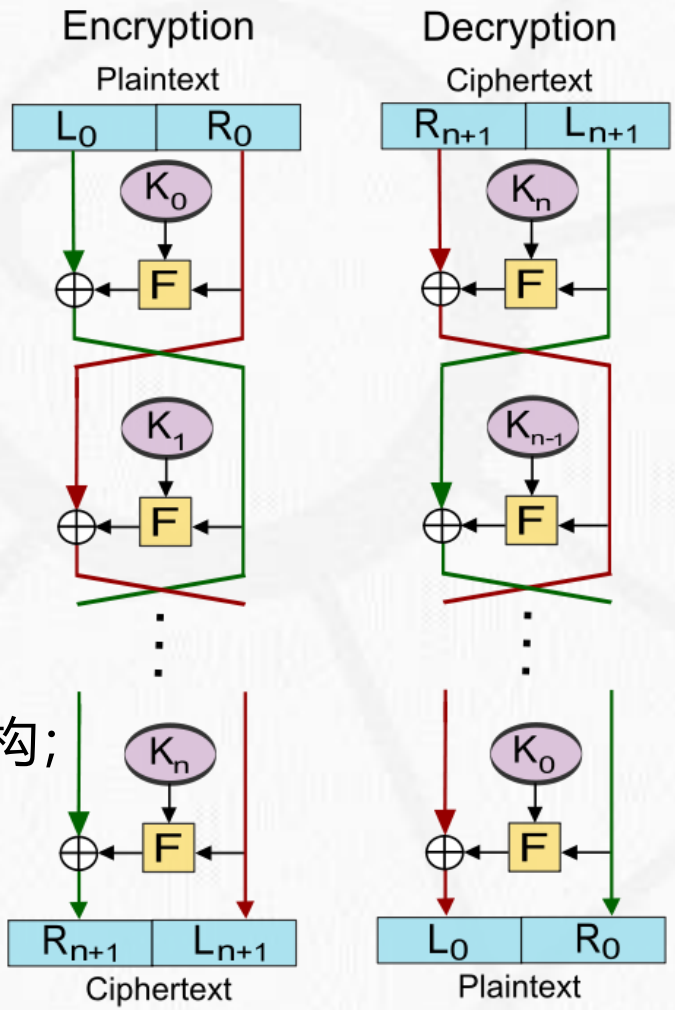



混淆距离

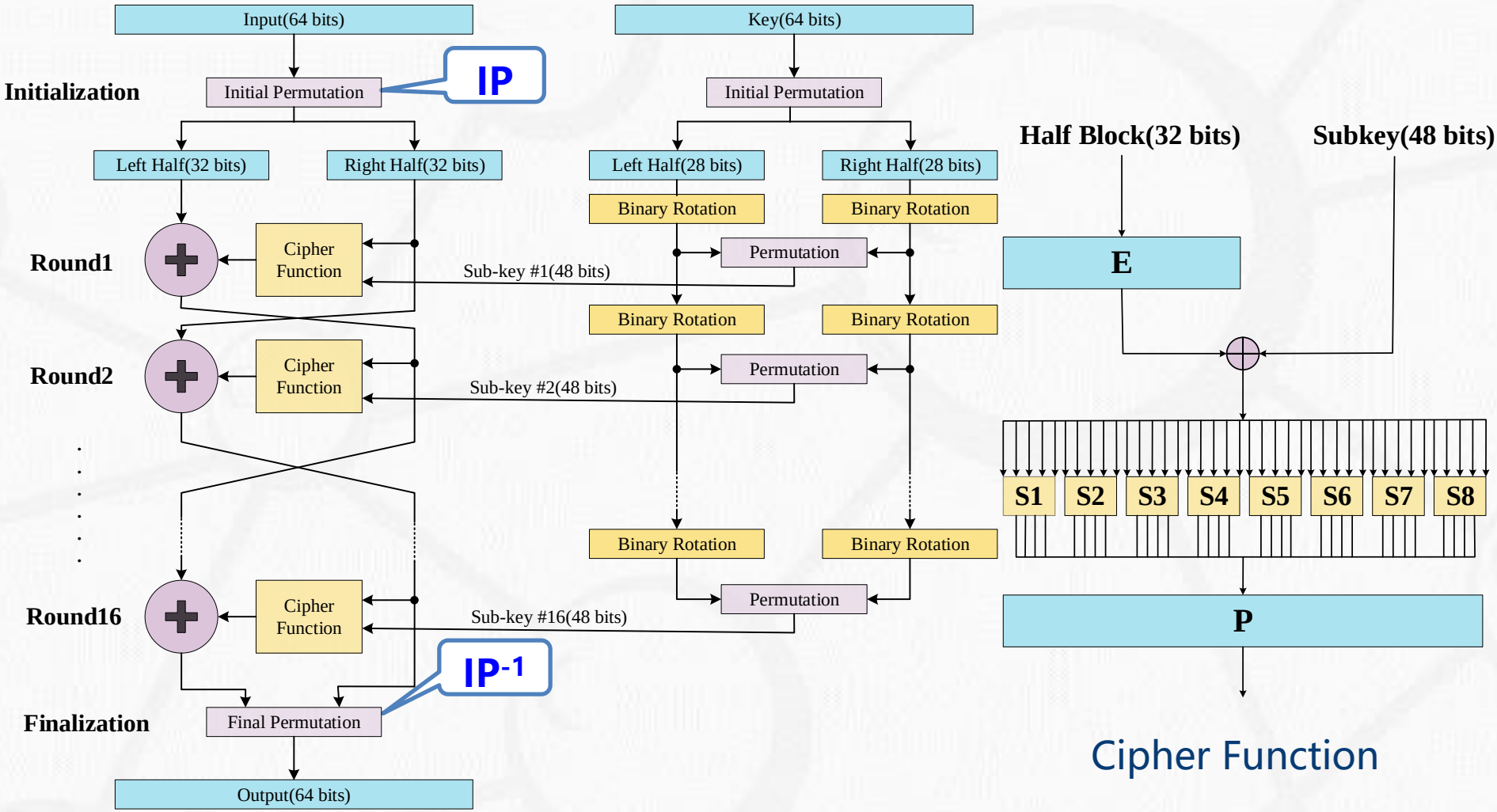
混淆大小关系

# §2.1.3 对称密码算法 - Feistel密码结构示例

- 输入：
  - 长为$2w$比特的明文分组；
  - 密钥$K$。
- 输出：
  - 长为$2w$比特的密文分组。
- 特点：
  - 循环函数对每次循环都有相同的通用结构；
  - 解密过程与加密过程基本相同。

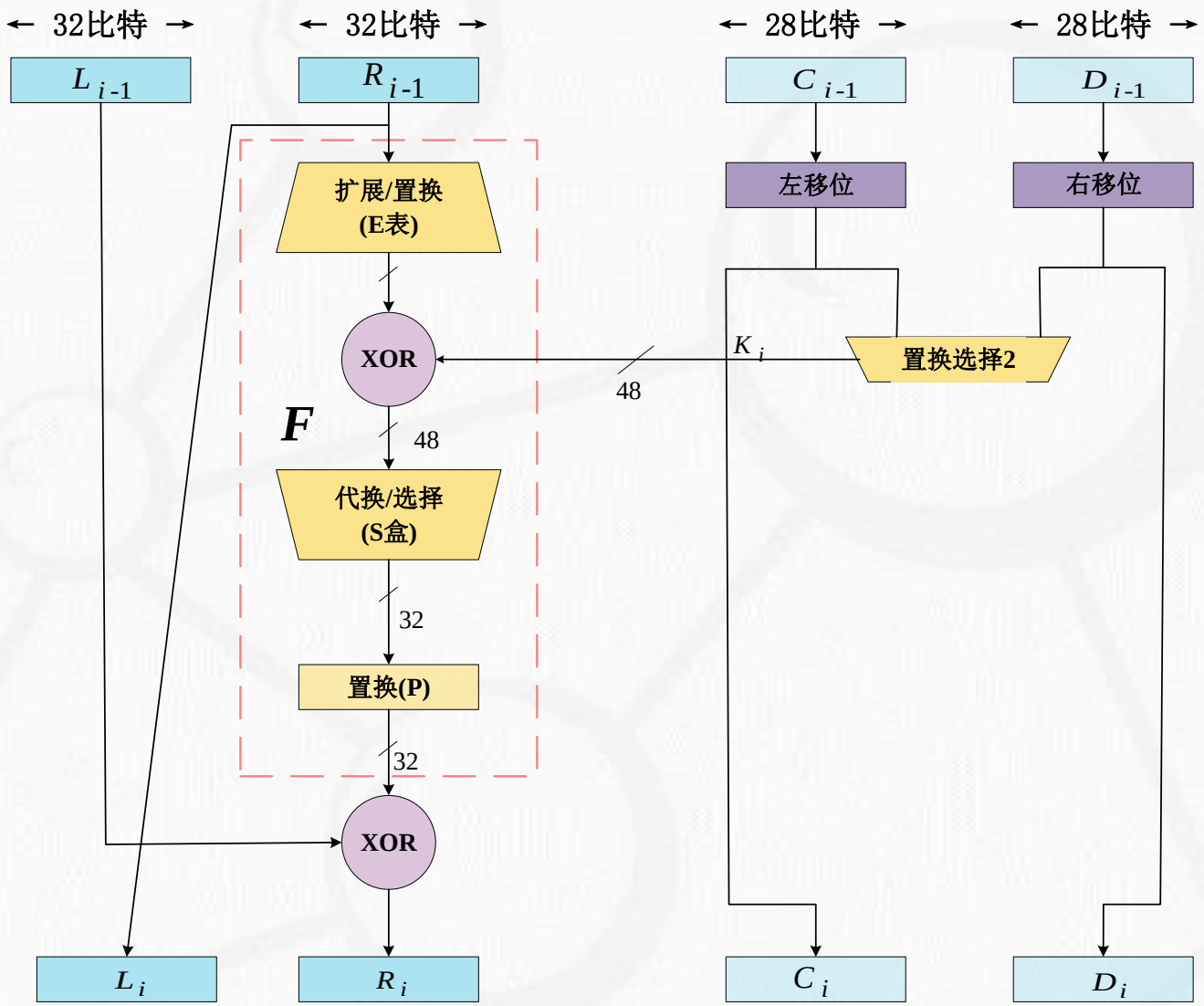


# §2.1.3 对称密码算法 - DES的整体结构




DES加密结构

# §2.1.3 对称密码算法 - DES的轮结构




DES算法的轮结构

# §2.1.3 对称密码算法 - DES的初始置换IP(1)

## Permutation Tables for DES

### 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 |

### 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 |

# §2.1.3 对称密码算法 - DES的初始置换IP(2)

**IP置换与IP逆置换互为逆置换**

原始输入矩阵M

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

IP

$M_{20} \rightarrow M'_{14}$

IP置换后矩阵 M '

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

$IP^{-1}$

$M'_{14} \rightarrow M''_{20}$

$IP^{-1}$置换后矩阵 M"

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

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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($IP^{-1}$)

# §2.1.3 对称密码算法 - DES的扩展函数

- 扩展置置换目标是IP置换后获得的右半部分R0，将32位(4位×8组)输入扩展为48位(6位×8组)输出，其规则如下表：扩展的数据是从相邻两组分别取靠近的一位，4位变为6位。靠近32位的位为1。表中第二行的4取自上组中的末位，9取自下组中的首位。

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

## Expansion Permutation(E)

输入32位

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

输出48位

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

# §2.1.3 对称密码算法 - DES的S盒

**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 |

代表行

代表列

首尾两位（10）

101100

中间四位（0110）

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

$S_1$

使用S盒的一个例子

得到2（0010），即
101100->0010

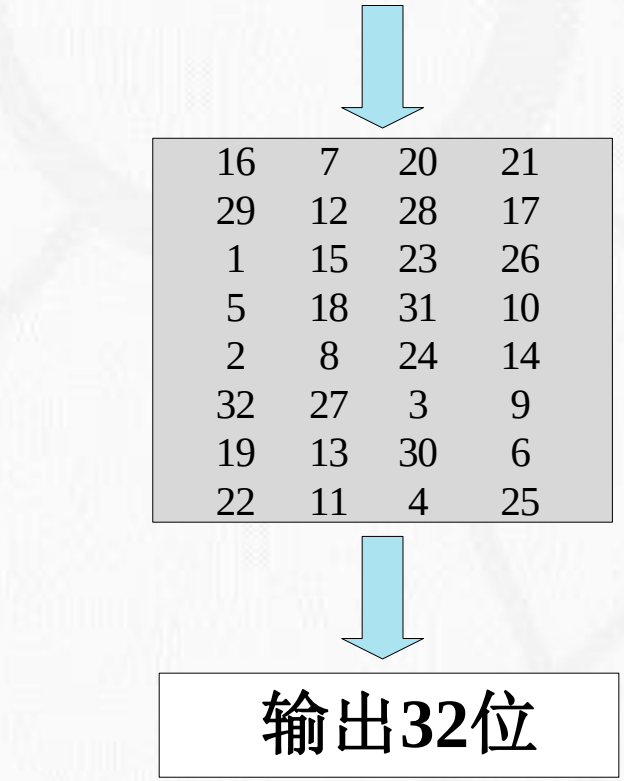
# §2.1.3 对称密码算法 - DES的P置换

- S盒代替运算的32位输出按照P盒进行置换。该置换把输入的每位映射到输出位，任何一位不能被映射两次，也不能被略去，映射规则如下表：

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

**Permutation Function(P)**

**输入32位（来自S盒）**



# §2.1.3 对称密码算法 - DES的子密钥生成

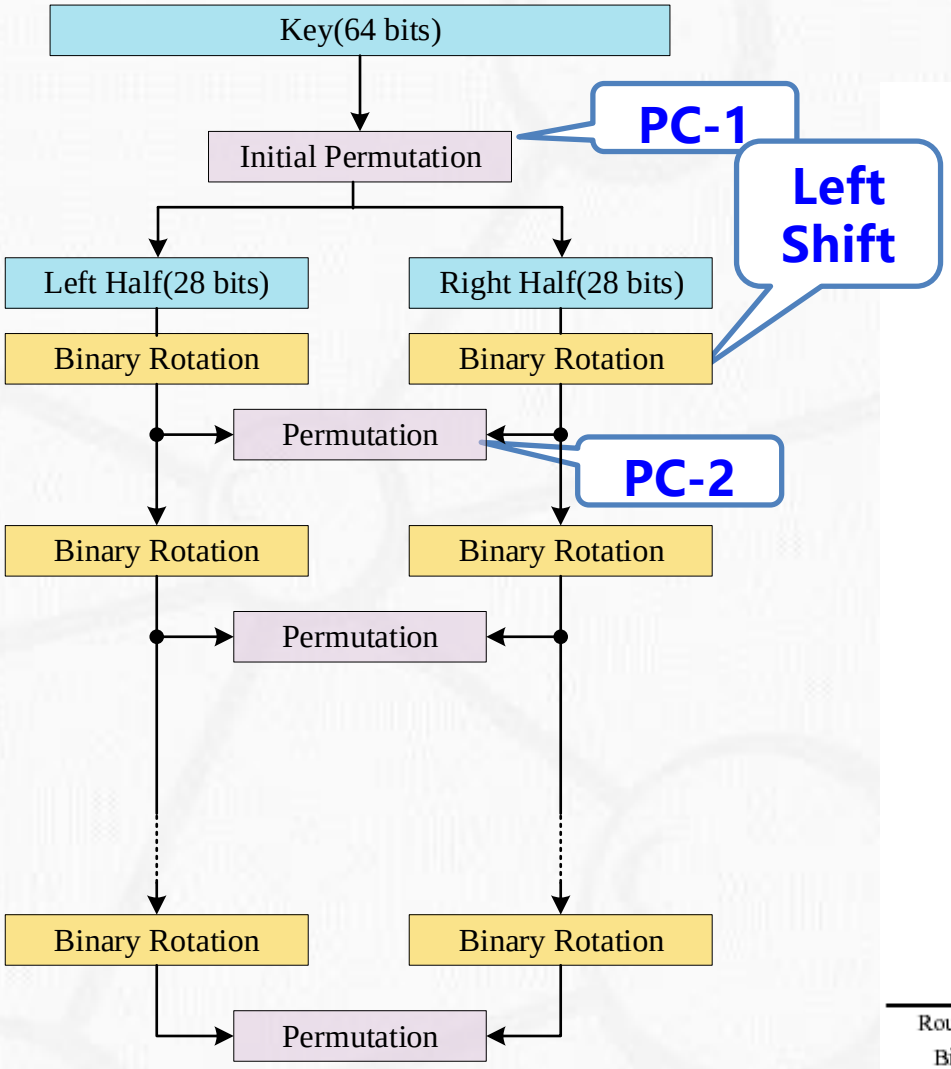



**Table 3.4 Tables Used for DES Key Schedule Calculation**

**(a) 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 |

**(b) 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 |

**(c) 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 |

# §2.1.3 对称密码算法 - DES的子密钥生成示例

**64位密钥**

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 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 | 41 | 52 | 53 | 54 | 55 | 56 |
| 57 | 58 | 59 | 60 | 61 | 62 | 63 | 64 |

**56位密钥**

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

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

**C0(28位子密钥)**

| | | | | | | |
|---|---|---|---|---|---|---|
| 63 | 55 | 47 | 39 | 31 | 33 | 15 |
| 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 |
| 21 | 13 | 5 | 28 | 20 | 12 | 4 |

**D0(28位子密钥)**

PC-1

**C0(28位子密钥)**

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

**D0(28位子密钥)**

| | | | | | | |
|---|---|---|---|---|---|---|
| 63 | 55 | 47 | 39 | 31 | 33 | 15 |
| 7 | 62 | 54 | 46 | 38 | 30 | 22 |
| 14 | 6 | 61 | 53 | 45 | 37 | 29 |
| 21 | 13 | 5 | 28 | 20 | 12 | 4 |

循环左移

循环左移

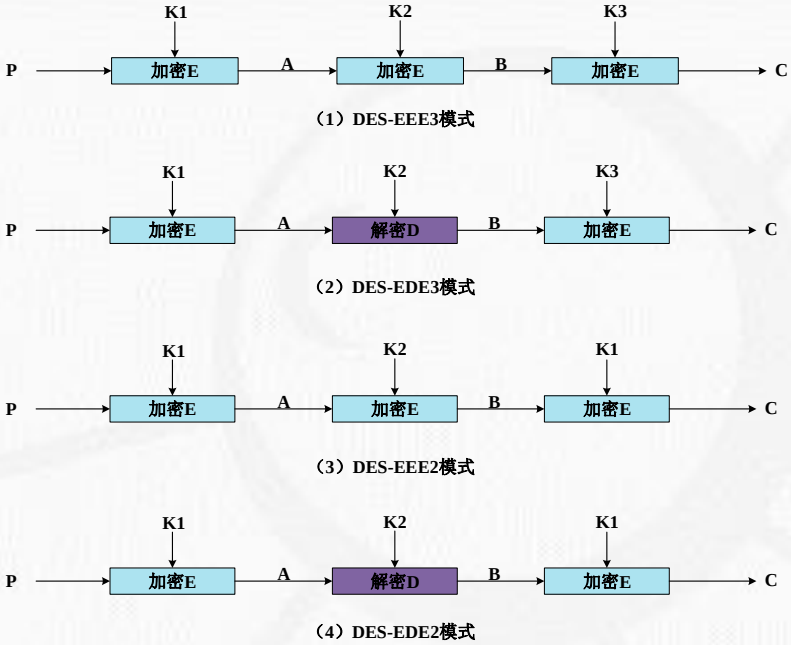
| | | | | | |
|---|---|---|---|---|---|
| 14 | 17 | 11 | 24 | 1 | 5 |
| 3 | 28 | 15 | 6 | 21 | 10 |
| 23 | 19 | 12 | 4 | 26 | 8 |
| 16 | 7 | 27 | 20 | 13 | 2 |
| 41 | 52 | 31 | 37 | 47 | 55 |
| 30 | 40 | 51 | 45 | 33 | 48 |
| 44 | 49 | 39 | 56 | 34 | 53 |
| 46 | 42 | 50 | 36 | 29 | 32 |

**48位密钥**

PC-2

# §2.1.3 对称密码算法 - DES存在的问题

- DES的问题：
  - 密钥过短；
  - 1998年已不作为标准。




- 3DES的优缺点：
  - 密钥长度增加到112位或168位，可以有效克服穷举搜索攻击；具备继续使用现有的DES实现的可能。
  - 处理速度相对较慢，特别是对于软件实现。
  - 明文分组的长度仍为64位，就效率和安全性而言，与密钥的增长不相匹配。

# §2.1.3 对称密码算法 - AES出现的背景(1)

- DES 不再安全:
  - 1977, US$20 million, 1 day, Diffie and Hellman;
  - 1993, US$1 million, 7 hours, Wiener;
  - 1998, US$ 250000, 56 hours, EEF;
- 3DES 效率较低:
  - 3次DES算法执行;
  - 数据分组为64b.

# §2.1.3 对称密码算法 - AES出现的背景(2)

- 1997年1月2日，美国国家标准和技术协会（NIST）宣布征集一个新的对称密钥分组密码算法作为取代DES的新的加密标准，该标准被命名为高级加密标准（Advanced encryption standard）。
- NIST对AES算法的要求简述如下：
  - 必须是一个非保密的，公开的，对称密钥加密算法
  - 算法必须支持128比特分组长度，128、192、256比特的密钥长度，强度应该相当于三重DES，但比三重DES更有效。
- 2000年10月，NIST宣布选中由比利时密码学家Daemen和Rijmen共同设计的Rijndael算法作为AES

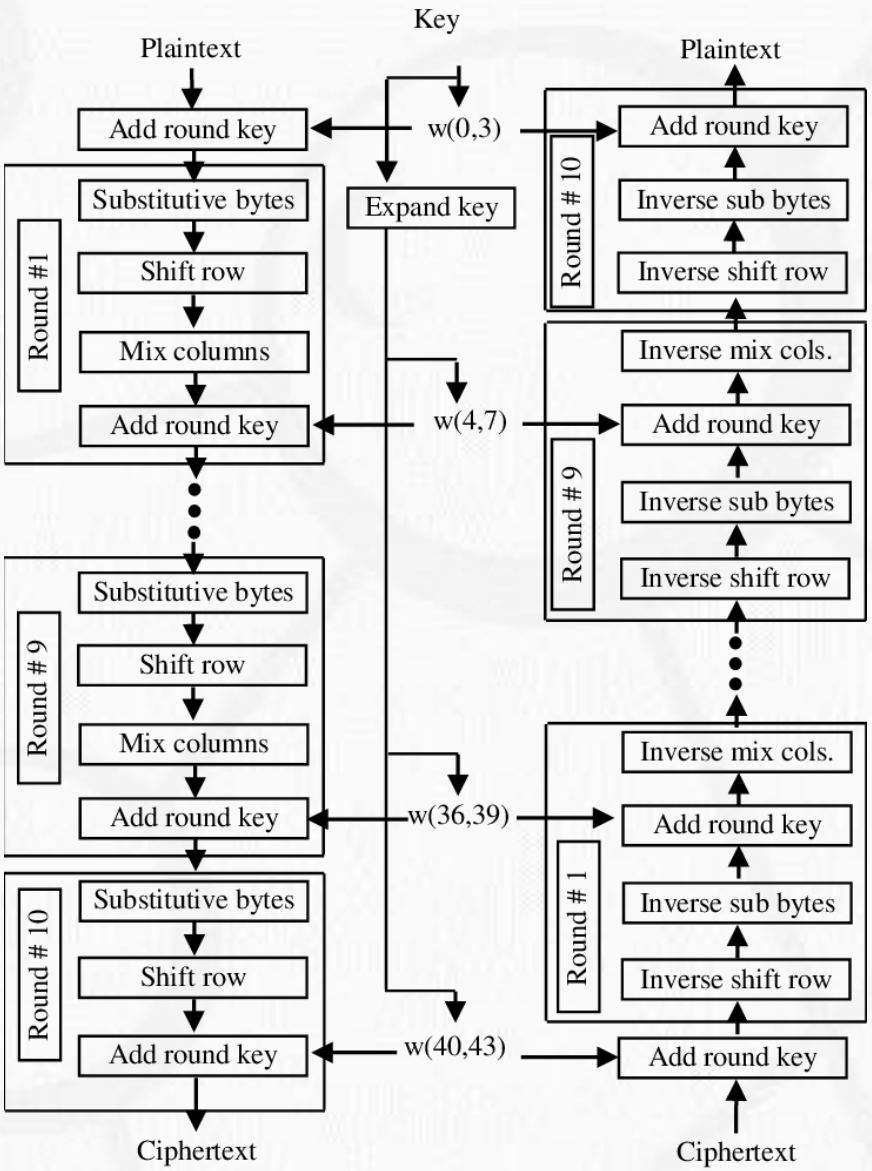
# §2.1.3 对称密码算法 - AES概述

- 分组长度为128位，密钥长度分别被指定为128位，192b位，256b位，这里以128位分组为例进行介绍.
- 特性:
  - 对多有已知的攻击具有免疫性;
  - 在各种平台上,其执行速度快且代码紧凑;
  - 设计简单,软硬件均容易实现.
- 输入为32B，输出也为32B。
- 利用有限域GF($2^8$)。

| 密钥长度B | 16 | 24 | 32 |
|---|---|---|---|
| 分组长度B | 16 | 16 | 16 |
| 轮数 | 10 | 12 | 14 |
| 每轮密钥长度B | 16 | 16 | 16 |

# §2.1.3 对称密码算法 - AES的整体结构

- 主要组成部分:
  - 轮转密钥加;
  - 字节代换;
  - 行位移;
  - 列混淆;
  - 扩展密钥;



# §2.1.3 对称密码算法 - AES的演示动画

**用一个动画使大家初探AES**

- AES的特点
  - 按字节进行计算，效率远高于DES(按位进行计算)；
  - 所有细节均有严格的数学推理；
  - 解密算法与加密算法不一致。

# §2.1.3 对称密码算法 - SM4的由来

- SM4分组密码算法是国家密码管理局于2006年1月6日公布的无线局域网产品使用的密码算法，是国内官方公布的第一个商用密码算法。

- SM4是一个分组密码算法，分组长度和密钥长度均为128比特。加密算法与密钥扩展算法都采用32轮非线性迭代结构。

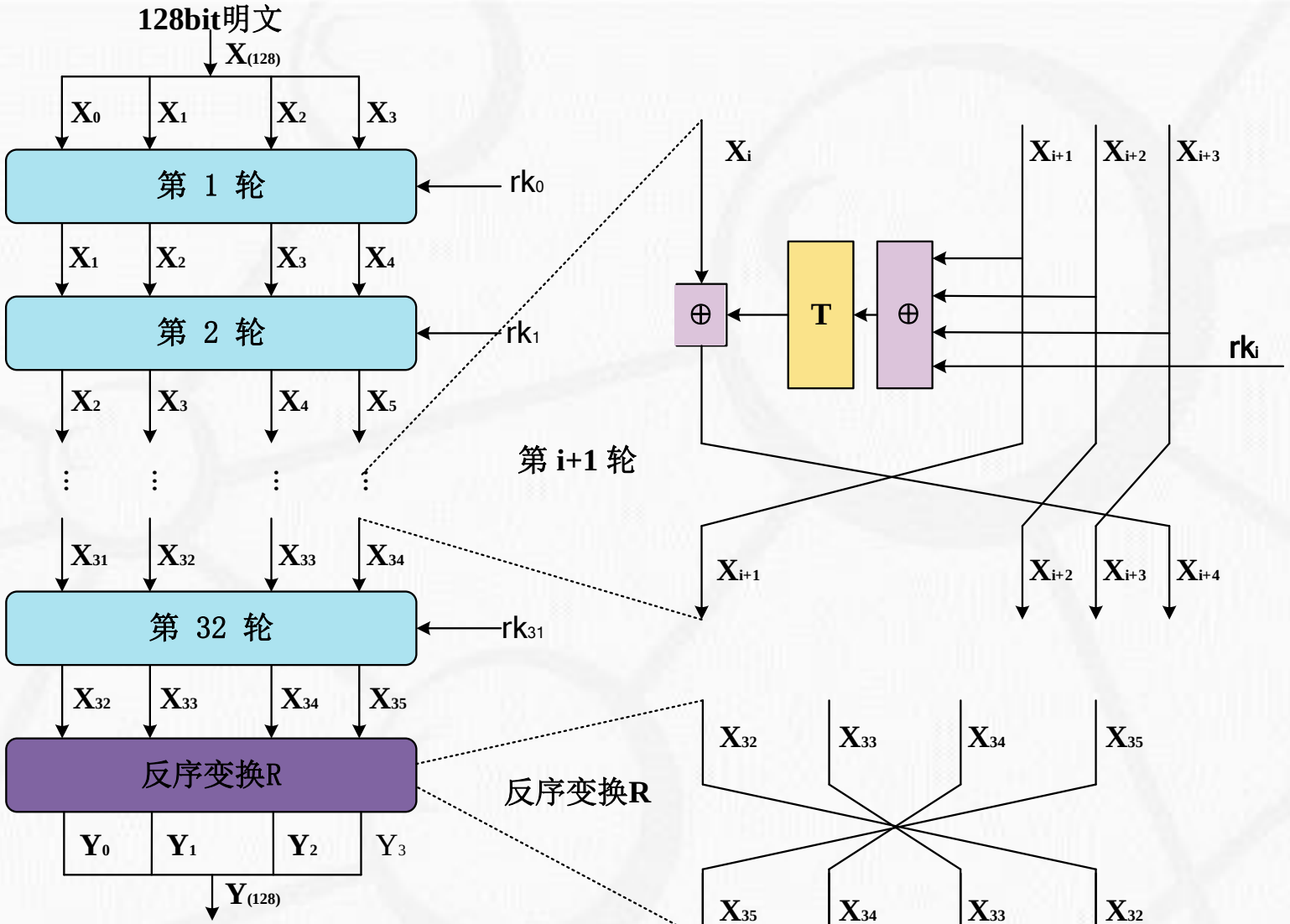
- 它的解密算法与加密算法的结构相同，只是轮密钥的使用顺序相反，解密轮密钥是加密轮密钥的逆序。

# §2.1.3 对称密码算法 - SM4术语说明

- $Z_2^e$表示e-比特的向量集，$Z_2^8$ 中的元素成为字节，$Z_2^{32}$中的元素称为字 。
- S盒是一个固定的8比特输入8比特输出的置换，记为$Sbox(.)$
- SM4中采用了两个基本运算：$\oplus$，32比特异或;$<<< i$,32比特循环左移$i$位。
- SM4的密钥长度为128比特，表示为

  $MK = (MK_0, MK_1, MK_2, MK_3)$,其中，$MK_i, i = 0,1,2,3$ 为字。
- 轮密钥为$(rk_0, rk_0, \ldots, rk_{31})$，$rk_i$为字。轮密钥由加密密钥通过密钥扩展算法生成。
- $FK = (FK_0, FK_1, FK_2, FK_3)$为系统参数。
- $CK = (CK_0, CK_1, \ldots, CK_{31})$为固定参数，用于密钥扩展算法。

# §2.1.3 对称密码算法 - SM4加密算法整体结构

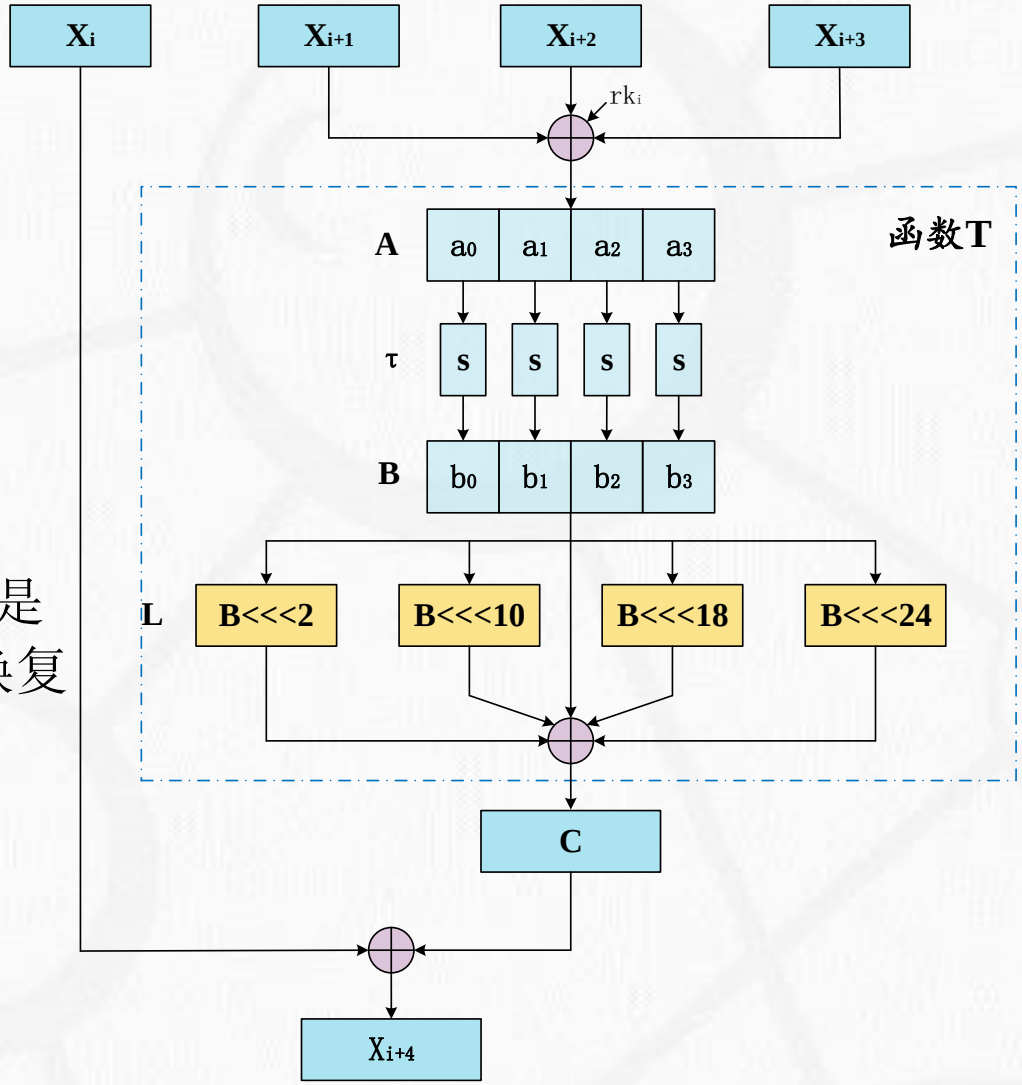



SM4加密结构

# §2.1.3 对称密码算法 - SM4的轮函数

- 设输入为$(X_i, X_{i+1}, X_{i+2}, X_{i+3}) \in (Z_2^{32})^4$，
  轮密钥为$rk_i \in Z_2^{32}$,则轮函数为：
  $$X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i) = X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i), i = 0,1,\ldots,31$$

- 其中$T: Z_2^{32} \to Z_2^{32}$称为合成置换，是一个由非线性变换盒一个线性变换复合而成的可逆变换，即
  $$T(.) = L(\tau(.))$$




SM4轮函数

# §2.1.3 对称密码算法 - SM4的S盒

- 非线性变换τ中所使用的S盒是一个具有很好密码学特性的、由8比特输入产生8比特输出的置换
- 在设计原理上，SM4比AES的S盒设计多了一个仿射变换：

$$y = A(Ax + B)^{-1} + B$$

| x \ y | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | d6 | 90 | e9 | fe | cc | e1 | 3d | b7 | 16 | b6 | 14 | c2 | 28 | fb | 2c | 05 |
| 1 | 2b | 67 | 9a | 76 | 2a | be | 04 | c3 | aa | 44 | 13 | 26 | 49 | 86 | 06 | 99 |
| 2 | 9c | 42 | 50 | f4 | 91 | ef | 98 | 7a | 33 | 54 | 0b | 43 | ed | cf | ac | 62 |
| 3 | e4 | b3 | 1c | a9 | c9 | 08 | e8 | 95 | 80 | df | 94 | fa | 75 | 8f | 3f | a6 |
| 4 | 47 | 07 | a7 | fc | f3 | 73 | 17 | ba | 83 | 59 | 3c | 19 | e6 | 85 | 4f | a8 |
| 5 | 68 | 6b | 81 | b2 | 71 | 64 | da | 8b | F8 | eb | 0f | 4b | 70 | 56 | 9d | 35 |
| 6 | 1e | 24 | 0e | 5e | 63 | 58 | d1 | a2 | 25 | 22 | 7c | 3b | 01 | 21 | 78 | 87 |
| 7 | d4 | 00 | 46 | 57 | 9f | d3 | 27 | 52 | 4c | 36 | 02 | e7 | a0 | c4 | c8 | 9e |
| 8 | ea | bf | 8a | d2 | 40 | c7 | 38 | b5 | a3 | f7 | f2 | ce | f9 | 61 | 15 | a1 |
| 9 | e0 | ae | 5d | a4 | 9b | 34 | 1a | 55 | ad | 93 | 32 | 30 | f5 | 8c | b1 | e3 |
| a | 1d | f6 | e2 | 2e | 82 | 66 | ca | 60 | c0 | 29 | 23 | ab | 0d | 53 | 4e | 6f |
| b | d5 | db | 37 | 45 | de | fd | 8e | 2f | 03 | ff | 6a | 72 | 6d | 6c | 5b | 51 |
| c | 8d | 1b | af | 92 | bb | dd | bc | 7f | 11 | d9 | 5c | 41 | 1f | 10 | 5a | d8 |
| d | 0a | c1 | 31 | 88 | a5 | cd | 7b | bd | 2d | 74 | d0 | 12 | b8 | e5 | b4 | b0 |
| e | 89 | 69 | 97 | 4a | 0c | 96 | 77 | 7e | 65 | b9 | f1 | 09 | c5 | 6e | c6 | 84 |
| f | 18 | f0 | 7d | ec | 3a | dc | 4d | 20 | 79 | ee | 5f | 3e | D7 | cb | 39 | 48 |

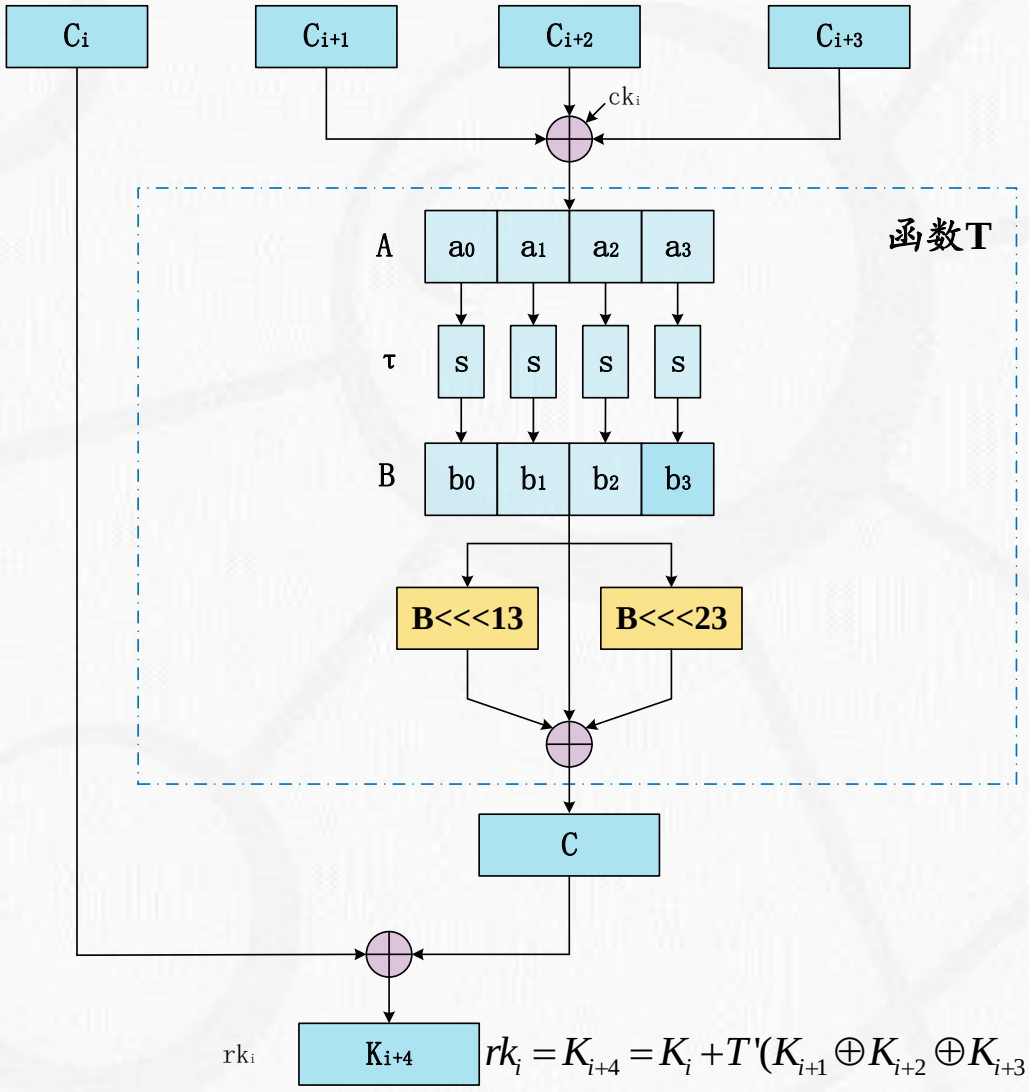
# 对称密码算法 - SM4的密钥扩展

- 设加密密钥
  $MK = (MK_0, MK_0, MK_0, MK_0)$,
  其中$MK_i$为字。

- 轮密钥的生成方法具体为：

$$(K_0, K_1, K_2, K_3) = (MK_0 \oplus FK_0, MK_1 \oplus FK_1, MK_2 \oplus FK_2, MK_3 \oplus FK_3)$$

$$rk_i = K_{i+4} = K_i \oplus T'(K_{i+1} \oplus K_{i+2} \oplus K_{i+3} \oplus CK_i)$$




SM4密钥扩展

# §2.1.3 对称密码算法 - 实际应用

- 分组密码算法名称的由来：
  - 每一种对称加密算法的明文输入均是规定长度的。
    - DES为64位，AES为128位，SM4为128位
  - 实际中明文的长度可以无限大，远大于对称加密算法输入的密文长度。

# 主观题 1分

设置

XIDIAN UNIVERSITY

## 如何进行大数据量的加密?

实际明文 M(长度远远大于128位)

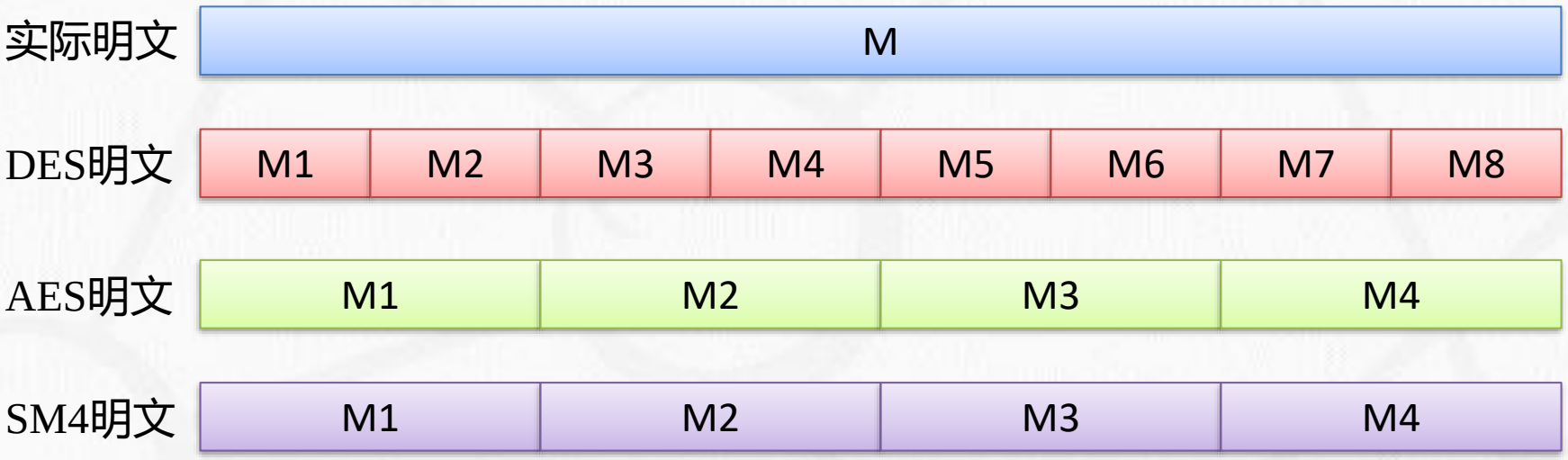
正常使用主观题需2.0以上版本雨课堂

作答

# §2.1.3 对称密码算法 - 实际应用

- 分组密码算法名称的由来：
  - 每一种对称加密算法的明文输入均是规定长度的。
    - DES为64位，AES为128位，SM4为128位
  - 实际中明文的长度可以无限大，远大于对称加密算法输入的密文长度。

- 将明文分组进行加密：



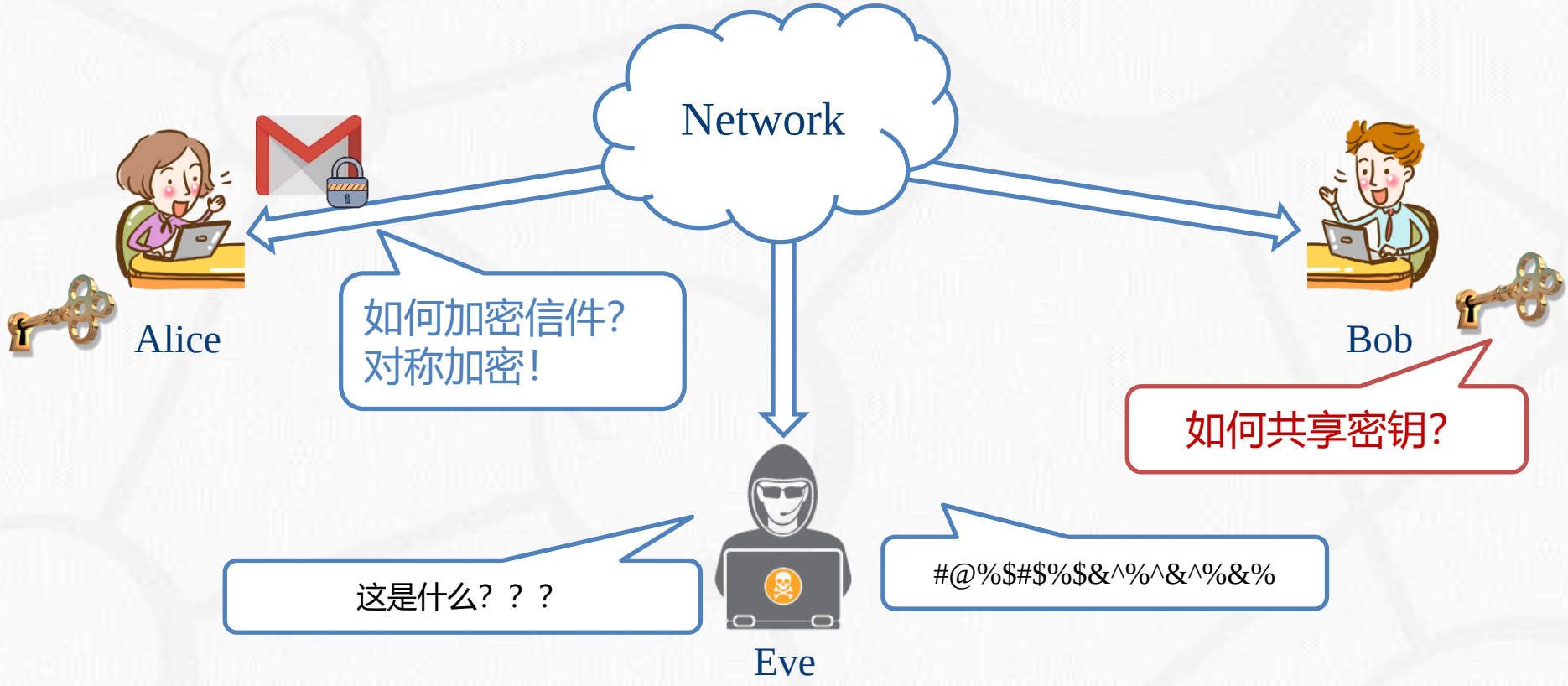

西安電子科技大學
XIDIAN UNIVERSITY

离散对数难解问题

# §2.1.4 D-H密钥交换协议

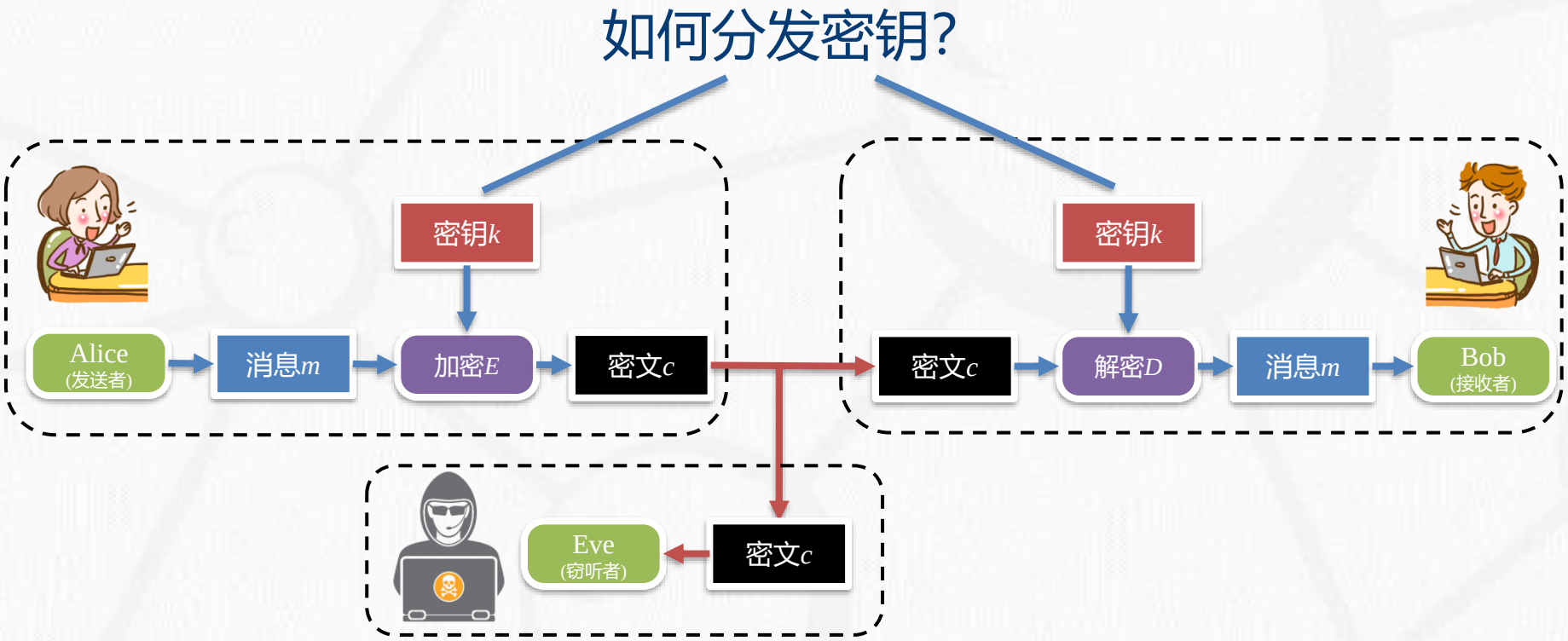
# §2.1.4 D-H密钥交换协议 - 回顾

- Alice有一条消息$m$(存放在邮件中)需要通过公开网络发送给Bob，并且其他人都得不到。
  - 对称密码体制并不能给出一个完美的解决方案。



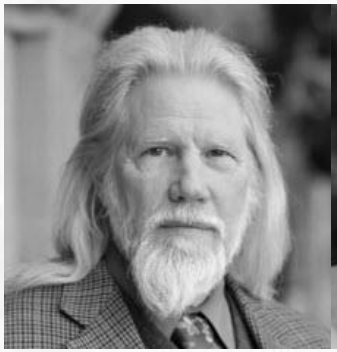
# §2.1.4 D-H密钥交换协议 - 密钥分发

- 如何使信源和信宿都得到相同的密钥?
  - 对称密码体制根本无法解决的一个问题。



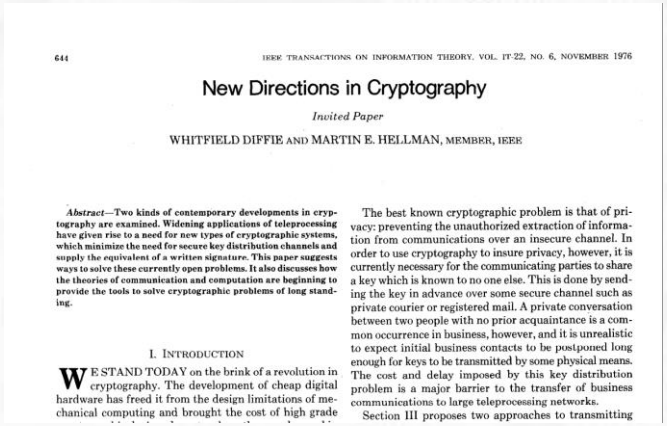
# §2.1.4 D-H密钥交换协议 - 现代密码的开端

- 1976年，Diffie和Hellman发表了“New Directions in Cryptography”[Diffie，1976]，提供了一个新的思想，即密码系统的加密密钥、解密密钥是可以不同的，由加密密钥和密文不能容易地求得解密密钥或明文，从而可以公开这种系统的加密算法和加密密钥可以公开，系统保密安全性完全依赖于秘密的解密密钥。



WHITFIELD DIFFIE　MARTIN HELLMAN

2015年图灵奖得主(D-H协议)



644　IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

New Directions in Cryptography

*Invited Paper*

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

*Abstract*—Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.

I. INTRODUCTION

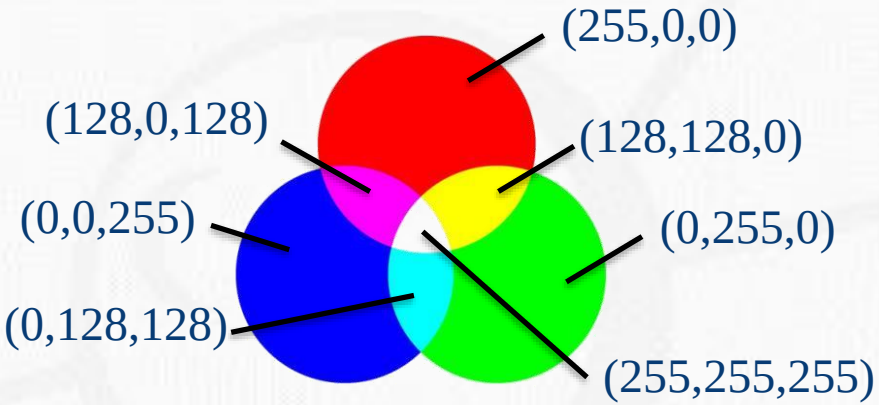
WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

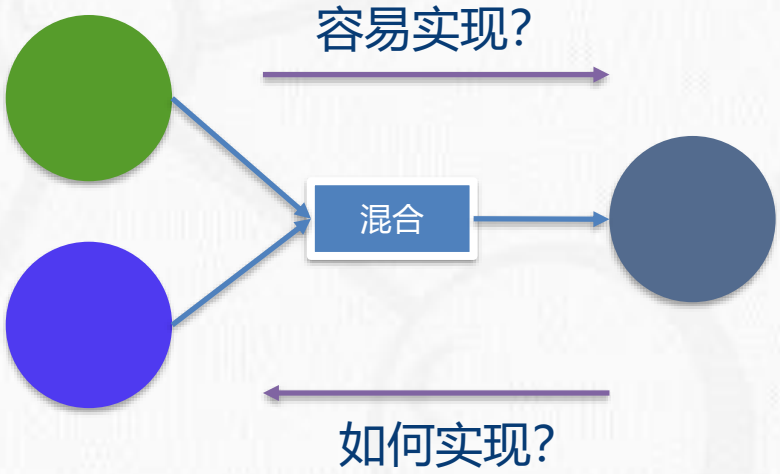
Section III proposes two approaches to transmitting

# §2.1.4 D-H密钥交换协议 - 秘密传输颜色

- 如何公开的传输秘密信息？
  - 来自三原色叠加的启发。
- 来自生活的例子：
  - 颜色叠加。




三原色示意图

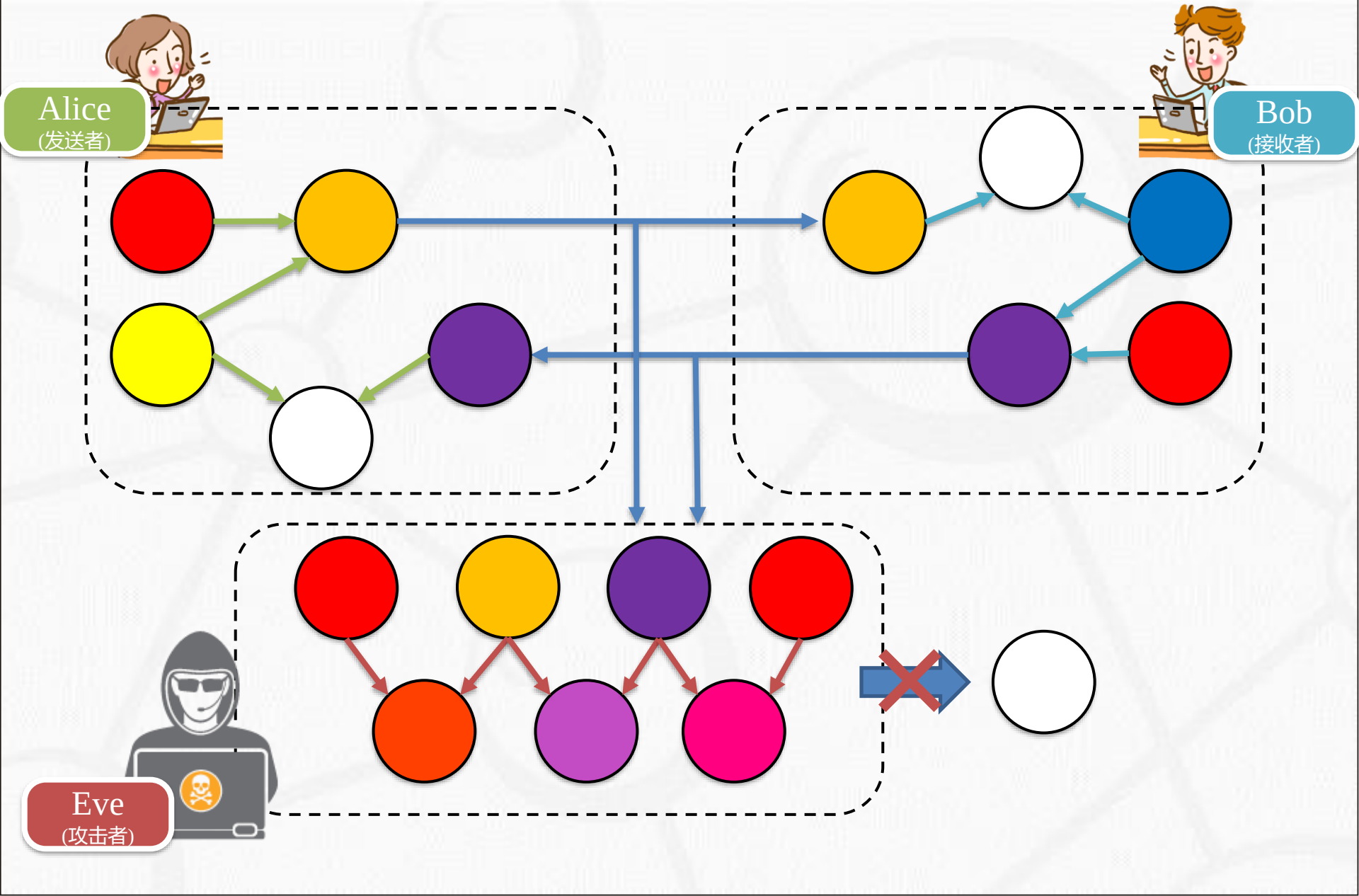




混合颜色的分解

- 现实中的单向函数。

# §2.1.4 D-H密钥交换协议 - 公开传输秘密信息



# §2.1.4 D-H密钥交换协议 - 离散对数文题

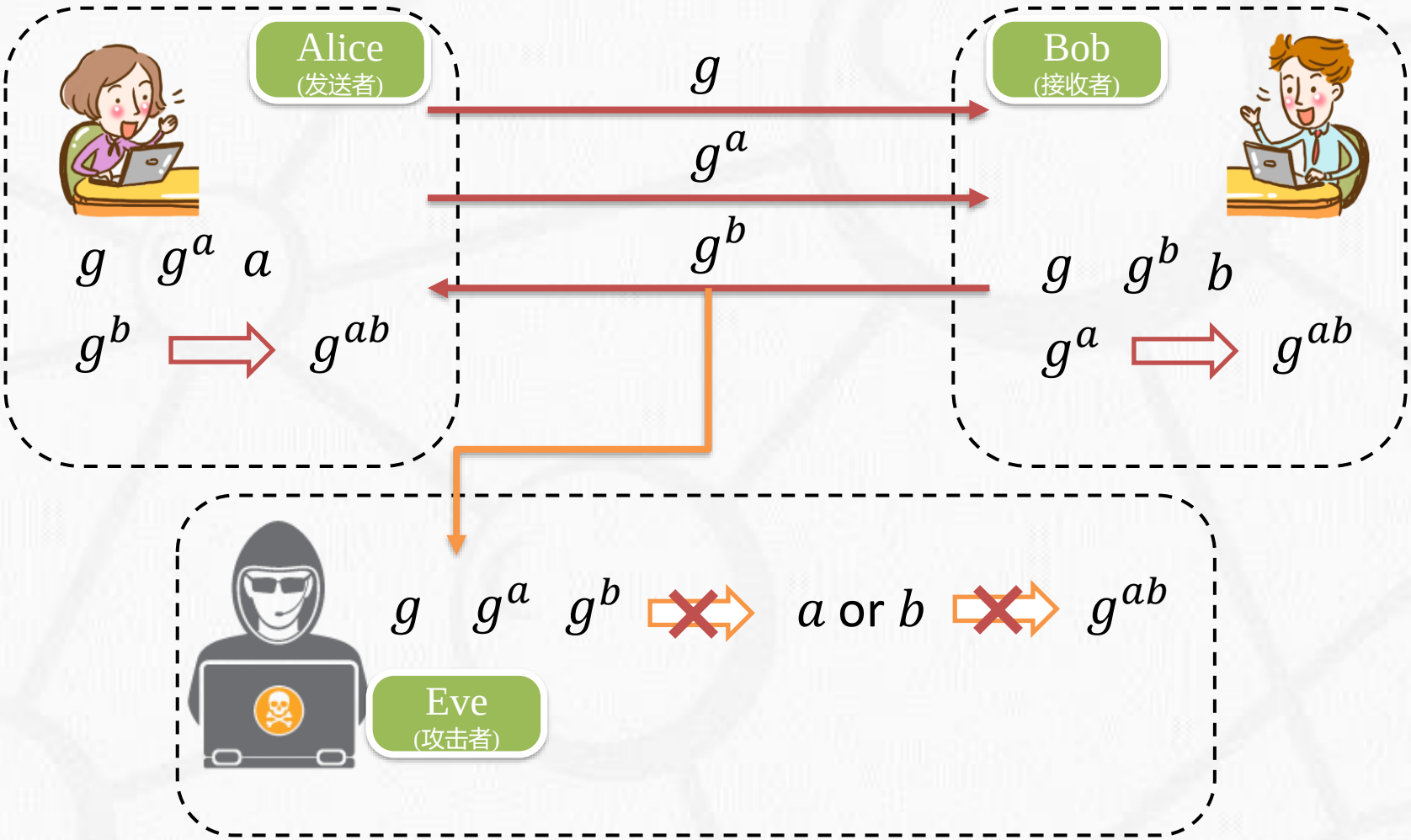
- 离散对数问题：
  - 给定一个阶为$q$乘法循环群$G$，其单位元为1。假设$B$是$G$的一个元，并且$g$是$G$的一个生成元。求解x，令$g^x = B$。


- 单向函数：
  - 函数$f(x) \to y$是一个单向函数(One-way function)，必须满足以下条件：
  - 由$y$求$x$极为困难；
  - 由$y$求$x$是易于实现的；
  - 其中，极为困难是对现有的计算资源和算法而言。

| 以2为基的$mod$ 11 循环群 | | | |
|---|---|---|---|
| $2^1$ | 2 | $2^7$ | 7 |
| $2^2$ | 4 | $2^8$ | 3 |
| $2^3$ | 8 | $2^9$ | 6 |
| $2^4$ | 5 | $2^{10}$ | 1 |
| $2^5$ | 10 | $2^{11}$ | 2 |
| $2^6$ | 9 | - | - |

# §2.1.4 D-H密钥交换协议 - 协议内容

- 生成元$g$和大素数$p$。以下默认为$mod\ q$运算。

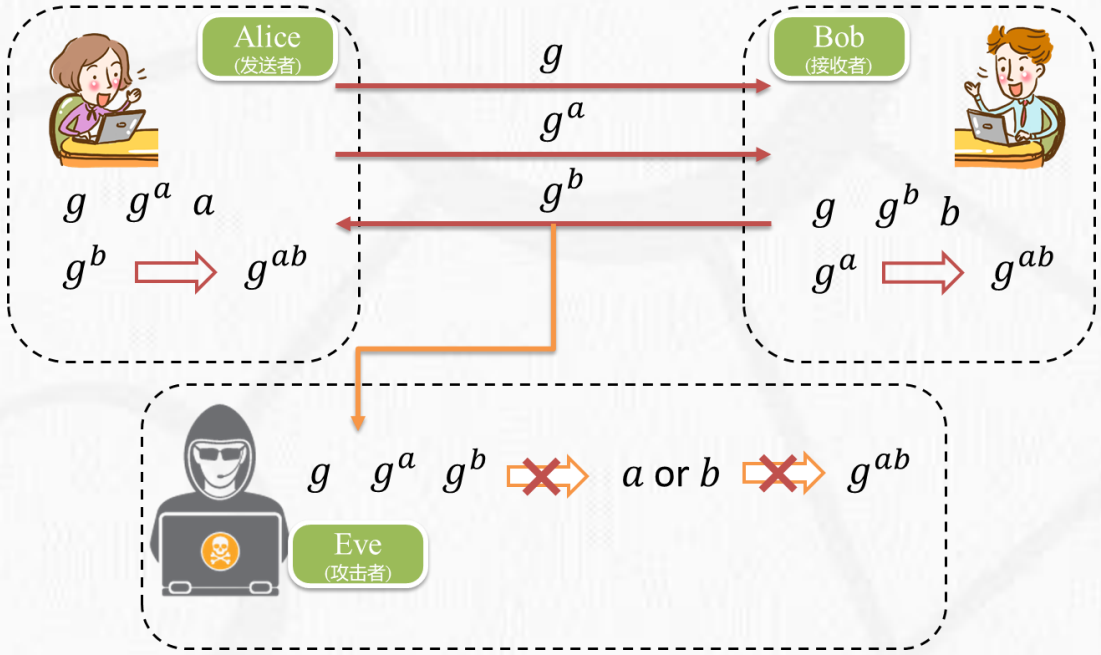


# 主观题 1分

设置

XIDIAN UNIVERSITY

- 通信方Alice和Bob希望交换会话密钥：
- 约定一个素数$q = 17$及生成元$g = 3$。
- 各自选择私钥：
  - $a = 15$，$b = 13$。




- 会话密钥是什么？

正常使用主观题需2.0以上版本雨课堂

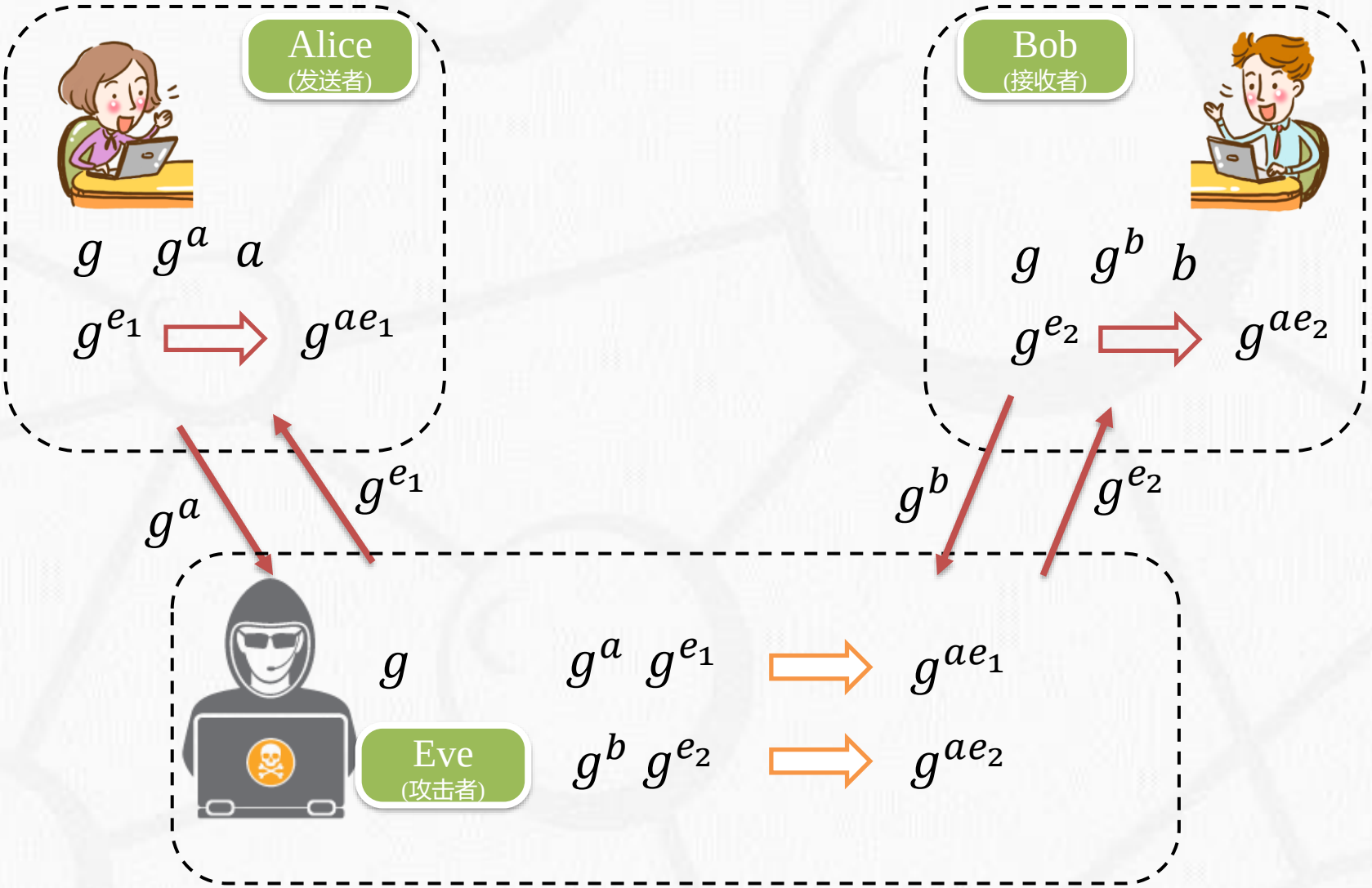
作答

# §2.1.4 D-H密钥交换协议 - 例子

- 通信方Alice和Bob希望交换会话密钥：
- 约定一个素数$q = 17$及生成元$g = 3$。
- 各自选择私钥：
  - $a = 15$，$b = 13$。
- 计算各自的公开信息：
  - $Y_a = g^a \ mod \ q = 3^{15} \ mod \ 17 = 6$；
  - $Y_{\mathrm{b}} = g^b \ mod \ q = 3^{13} \ mod \ 17 = 12$；
- 计算会话密钥：
  - $K_{a\mathrm{b}} = {Y_b}^a \ mod \ q = 12^{15} \ mod \ 17 = 10$；
  - $K_{a\mathrm{b}} = {Y_a}^b \ mod \ q = 6^{13} \ mod \ 17 = 10$；

# §2.1.4 D-H密钥交换协议 - 安全吗?

- 生成元$g$和大素数$p$。以下默认为$mod\ q$运算。



# §2.1.1 & §2.1.2 小结

- 内容回顾
  - Feistel结构、扩散和混淆的概念
  - DES、AES、SM4的基本结构和实现
  - D-H密钥交换协议
  - 中间人攻击

- 掌握
  - Feistel结构的工作原理
  - DES的工作原理
  - D-H密钥交换协议的计算流程
  - 中间人攻击的基本原理

Thanks!
Questions & Advices!