



密码学

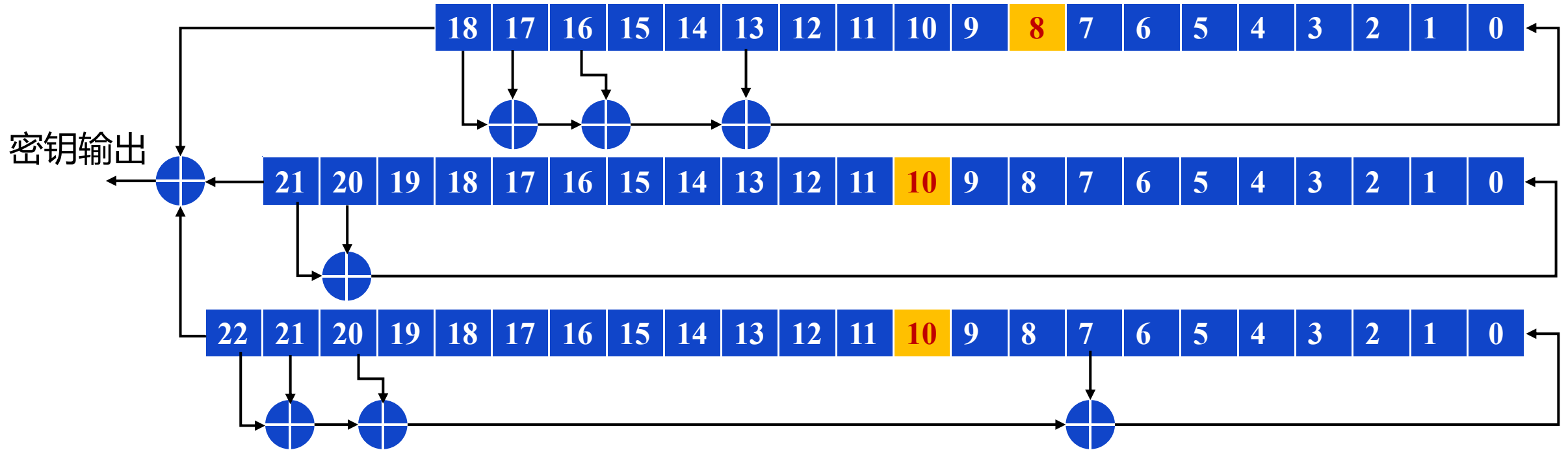
第四章 分组密码

网络空间安全学院

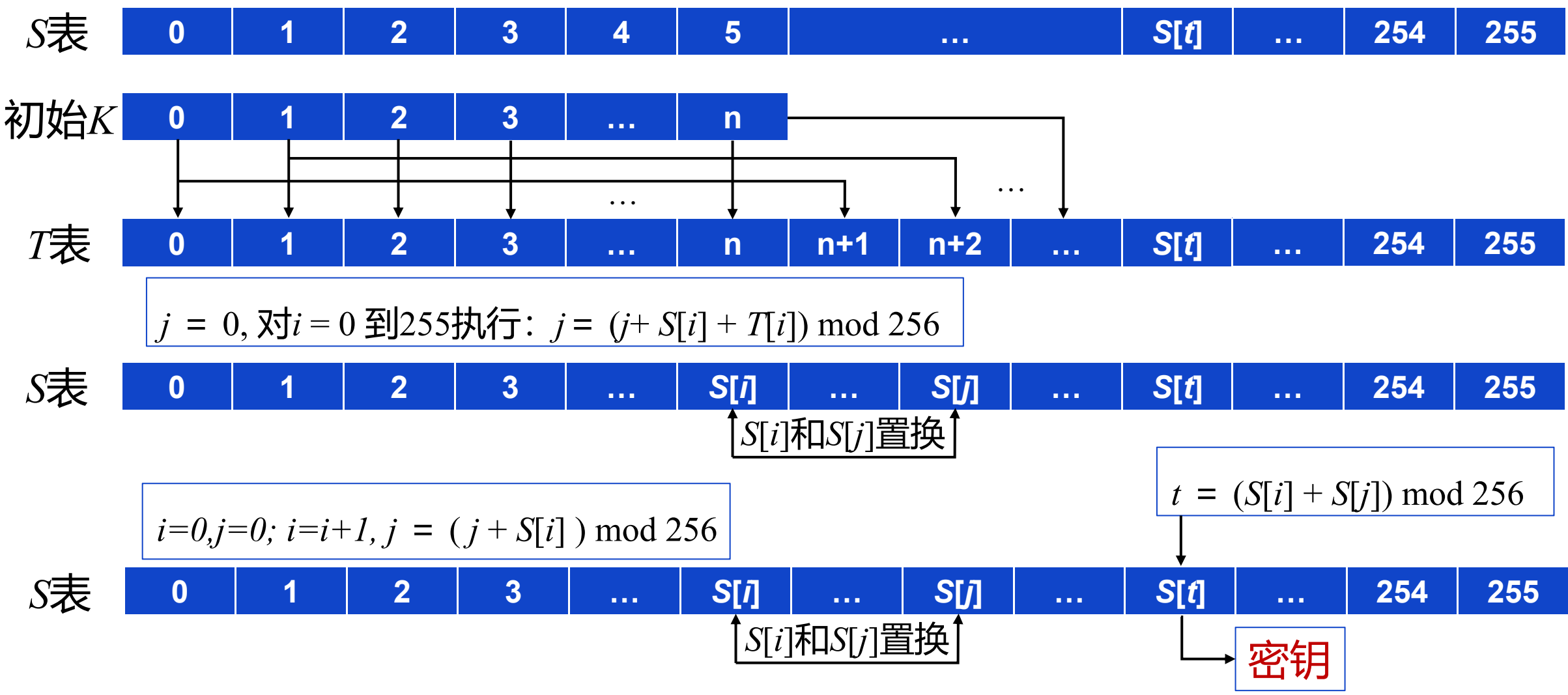
朱丹

zhudan@nwpu.edu.cn

- ✦ A5算法的种子密钥为64位，作为三个线性反馈移位寄存器的初始状态
- ✦ A5算法的输入是22位长的帧序号和64位长的密钥，输出为228位的流密钥序列
- ✦ 每个时钟周期产生一个比特的密钥，多周期形成密钥流（时钟脉冲控制移位）

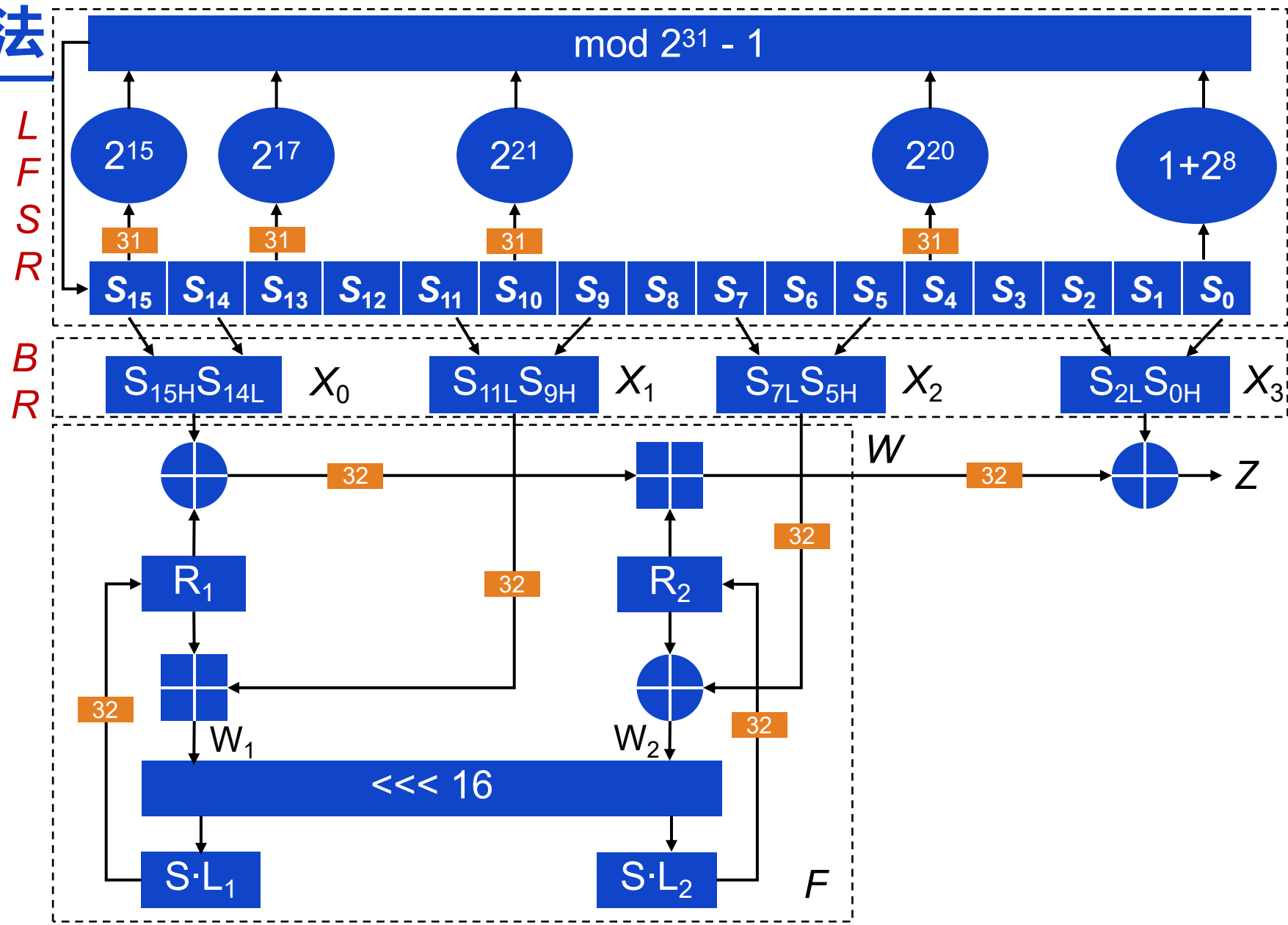


- ✦ 密钥长度为64比特，长度较短，安全性不足
- ✦ 线性移位寄存器输出的线性组合



知识回顾—ZUC密码算法

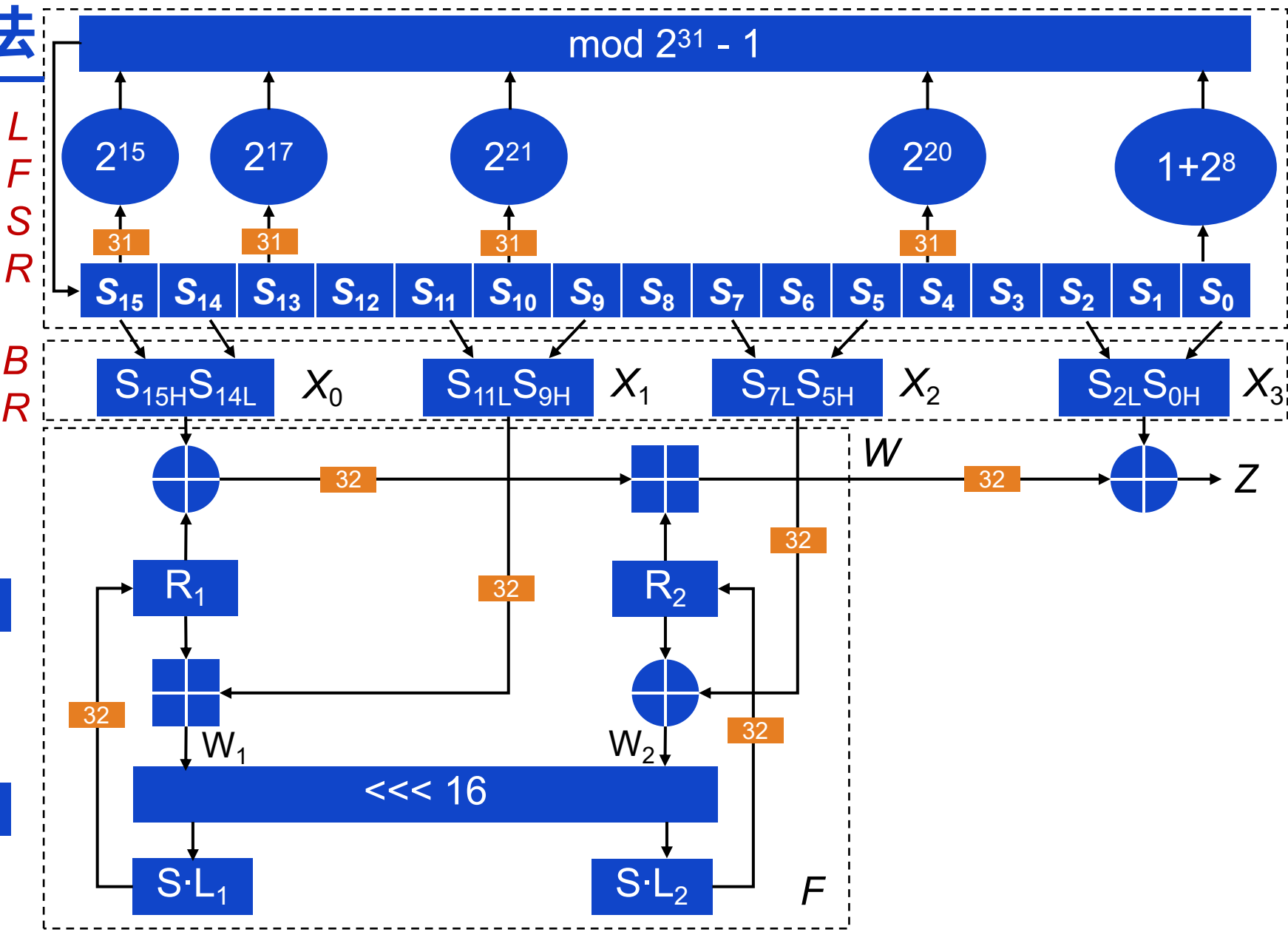
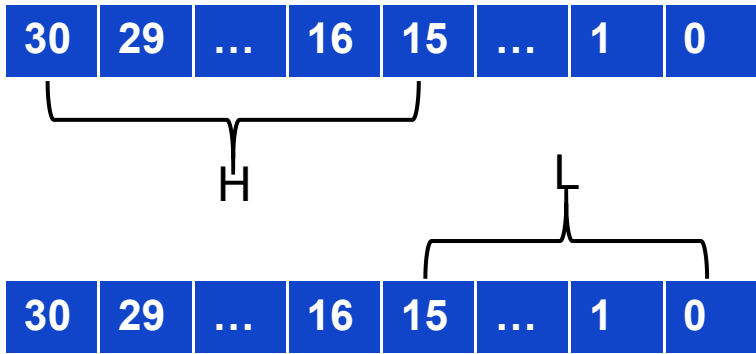
-  模2加
-  mod 2³²加
-  循环左移



✎ 掌握ZUC的基本运算类型
✎ 注意：有限域发生了变化，寄存器的宽度也发生了变化（31比特）

知识回顾—ZUC密码算法

- 三层结构
- LFSR
- 比特重组
- 非线性F函数



✎ F函数是ZUC密码算法中唯一的非线性部件，对算法安全性至关重要

✎ 模2加

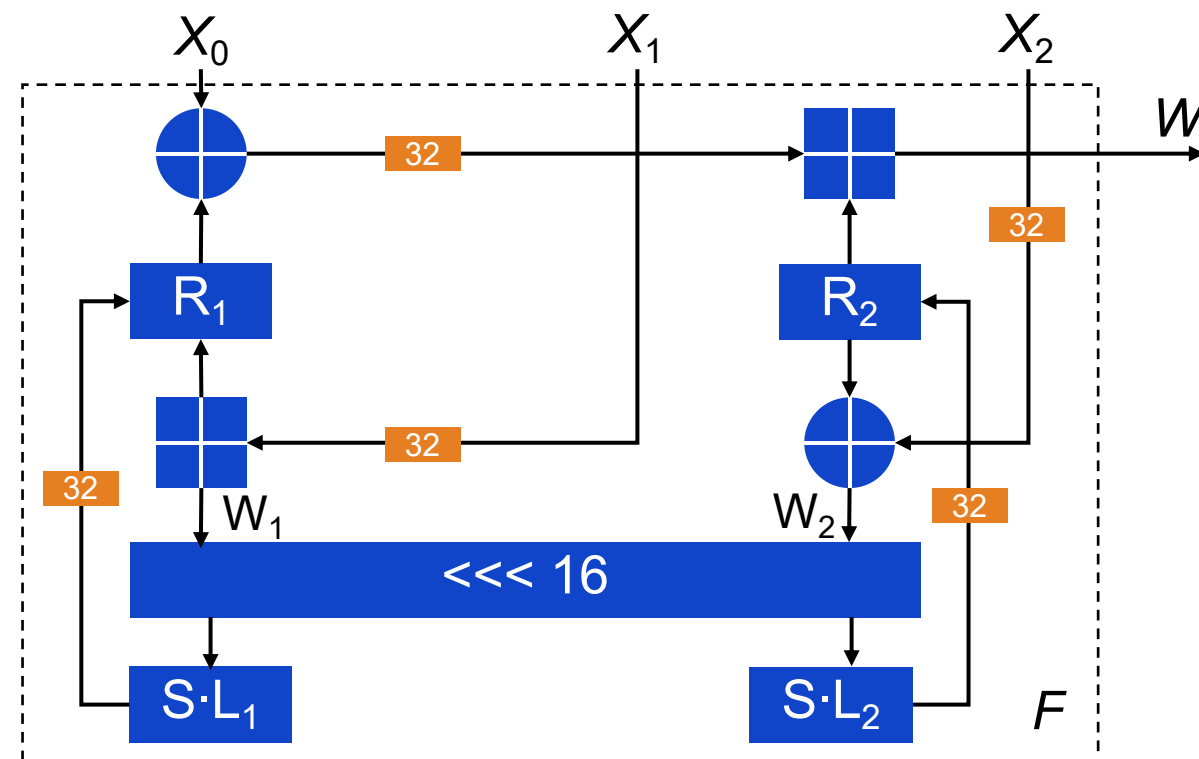
✎ $\text{mod } 2^{32}$

✎ 循环左移16位

✎ S盒置换：4个S盒 (8*8)

$$S = (S_0, S_1, S_0, S_1)$$

✎ L_1 和 L_2



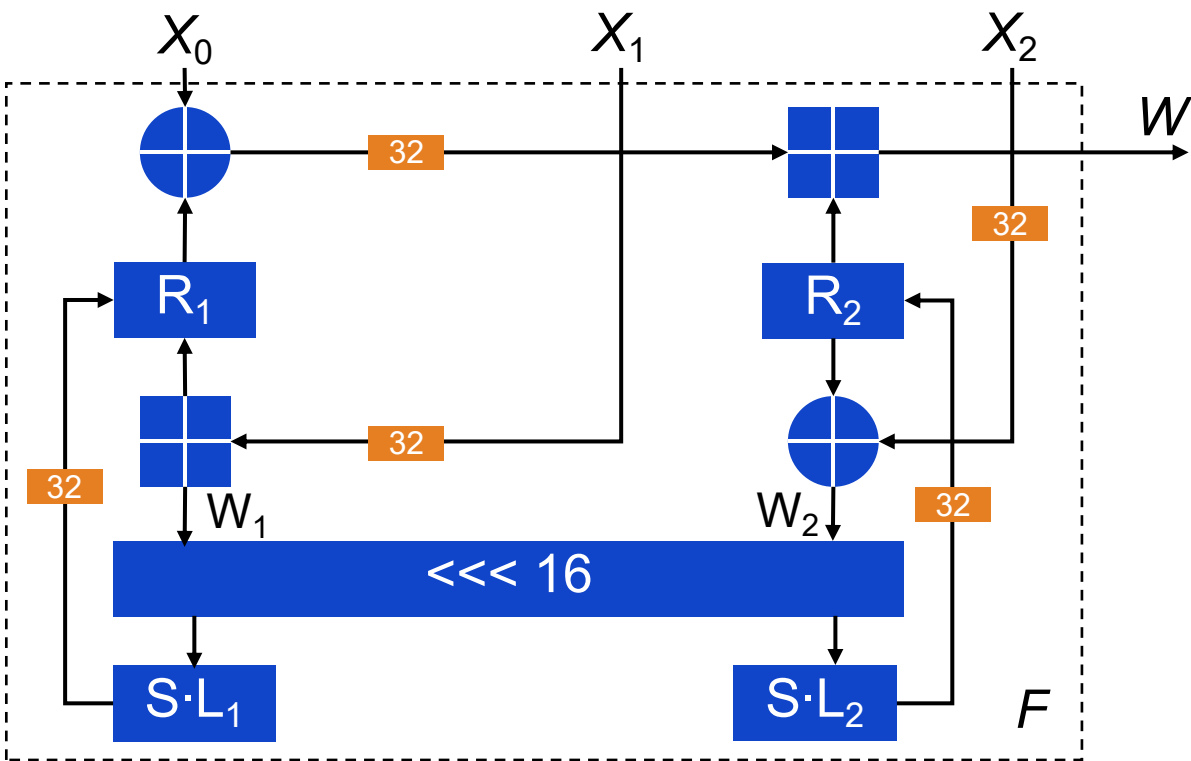
$$L_1(X) = X \oplus (X \lll 2) \oplus (X \lll 10) \oplus (X \lll 18) \oplus (X \lll 24)$$

$$L_2(X) = X \oplus (X \lll 8) \oplus (X \lll 14) \oplus (X \lll 22) \oplus (X \lll 30)$$

✎ 理解F函数对应ZUC安全性的重要性，F函数是非线性部件，其中包含非线性S盒替代

✎ ZUC使用的S盒规模；此外，ZUC使用了两种不同的S盒， S_0 和 S_1

F函数是ZUC密码算法中唯一的非线性部件，对算法安全性至关重要



输入：
密钥 k : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
初始向量 iv : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
输出：
 z_1 : 27bede74
 z_2 : 018082da
初始化：
线性反馈移位寄存器初态：






i	S_{0+i}	S_{1+i}	S_{2+i}	S_{3+i}	S_{4+i}	S_{5+i}	S_{6+i}	S_{7+i}
0	0044d700	0026bc00	00626b00	00135e00	00578900	0035e200	00713500	0009af00
8	004d7800	002f1300	006bc400	001af100	005e2600	003c4d00	00789a00	0047ac00
i	X_0	X_1	X_2	X_3	R_1	R_2	W	S_{15}
0	008f9a00	f100005e	af00006b	6b000089	67822141	62a3a55f	008f9a00	4563cb1b
1	8ac7ac00	260000d7	780000e2	5e00004d	474a2e7e	119e94bb	4fe932a0	28652a0f
2	50cacb1b	4d000035	13000013	890000c4	c29687a5	e9b6eb51	291f7a20	7464f744

理解F函数对应ZUC安全性的重要性，F函数是非线性部件，其中包含非线性S盒替代
通过例子理解字节重组是如何进行的

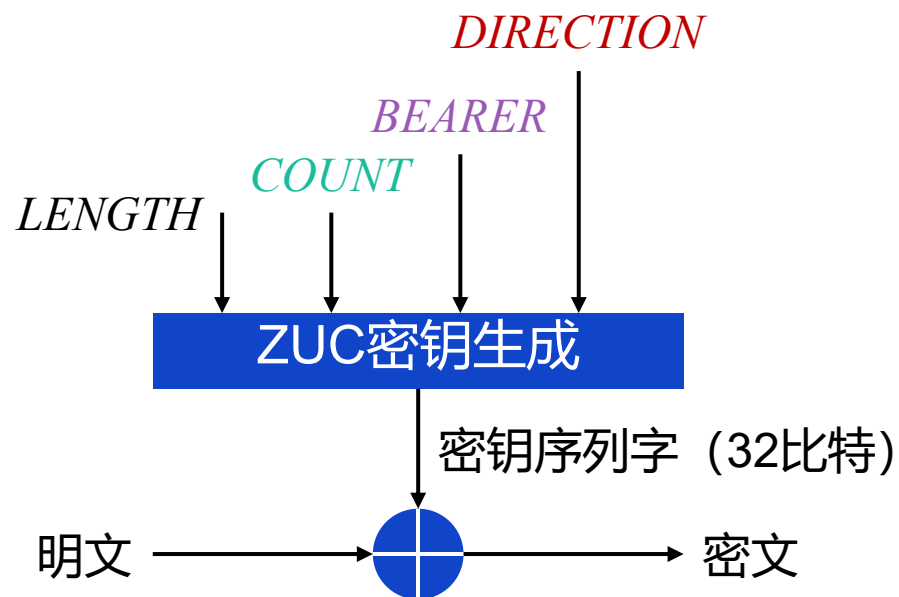
- ✎ S1: 将128位初始密钥 KEY 和128位初始向量 IV 装载到LFSR的 S_0 至 S_{15}
 - ✎ 128位初始密钥 KEY 分为16个字节 $KEY = k_0 \parallel k_1 \parallel \dots \parallel k_{15}$
 - ✎ 128位初始向量 IV 分为16个字节 $IV = iv_0 \parallel iv_1 \parallel \dots \parallel iv_{15}$
 - ✎ 240比特的常量 D (教材P106), 分为16个15比特的常量 $D = d_0 \parallel d_1 \parallel \dots \parallel d_{15}$

$$S_i = k_i \parallel d_i \parallel iv_i$$

ZUC算法运行流程

-  S1: 将128位初始密钥 KEY 和128位初始向量 IV 装载到LFSR的 S_0 至 S_{15}
-  S2: 置F函数中的两个32比特寄存器 R_1 和 R_2 为0
-  S3: 以初始化模式运行32次
-  S4: 以工作模式运行一次, 并将输出 W 舍弃
-  S5: 进入密钥产生阶段, 每时钟节拍产生32位密钥字 Z

- 祖冲之序列密码的机密性算法128-EEA3算法结构，最主要的还是密钥生成



- 通过如下规则产生 iv :

$$iv_0 = COUNT[0], iv_1 = COUNT[1]$$

$$iv_2 = COUNT[2], iv_3 = COUNT[3]$$

$$iv_4 = BEARER \parallel DIRECTION \parallel 00$$

$$iv_5 = iv_6 = iv_7 = 00000000$$

$$iv_{j+8} = iv_j, j = 8, 9 \dots 15$$

章节安排

Outline



分组密码概述



分组密码工作模式

章节安排

Outline



分组密码概述



分组密码工作模式



- ✎ 对称和非对称密码算法是以密钥来区分的
- ✎ 对称密码算法包括分组密码算法和序列密码算法

- ✎ 分组密码研究起始于20世纪70年代
- ✎ 1973年DES的颁布实施揭开了商用密码研究的序幕
- ✎ 典型的分组密码算法
 - ✎ DES, 3-DES
 - ✎ IDEA, AES, SM4
 - ✎ 轻量级分组密码LED、SIMON、PRESENT等

- ✎ 将明文按规定的长度**分组**
- ✎ 密文的一个比特与**整个明文分组**相关
- ✎ 实质是较**复杂的单表代替密码**

密钥 $k = (k_1, k_2, \dots, k_r)$

密钥 $k = (k_1, k_2, \dots, k_r)$

$m = (m_1, m_2, \dots, m_n)$

$c = (c_1, c_2, \dots, c_n)$

$m = (m_1, m_2, \dots, m_n)$



✎ 理解分组的含义，明文按照规定的长度分组；图中体现分组密码属于对称密码

✎ 理解分组密码的本质：复杂的单表替代

- ✎ 分组长度 n 要足够大，以抗报文的穷举攻击
- ✎ 密钥空间要足够大，以抗密钥的穷举攻击
- ✎ 由密钥确定的置换算法要足够复杂，以抗报文的统计分析

将简单且易于实现的密码系统进行组合，构成较复杂，且密钥空间较大的密码系统

- ✎ **概率加权和**：以一定的**概率**随机地从**多个加密子系统**中选择一个用于加密当前的明文
- ✎ 设有 r 个加密子系统 E_1, E_2, \dots, E_r , 相应的被选用的概率为 P_1, P_2, \dots, P_r , 其中 $\sum_{i=1}^r P_i = 1$

$$E = P_1 E_1 + P_2 E_2 + \dots + P_r E_r$$

- ✎ **乘积密码**：Shannon提出的一种强化密码的方法
- ✎ 定义：对于 $M = C$ 的两个密码 $S_1 = (M, M, K_1, E_1, D_1)$ 和 $S_2 = (M, M, K_2, E_2, D_2)$ ，定义 $S_1 \times S_2 = (M, M, K_1 \times K_2, E, D)$ ，对于密钥 (k_1, k_2) ，其加解密过程分别为：

$$c = E_{k_2}(E_{k_1}(m))$$

$$m = D_{k_1}(D_{k_2}(c))$$

- ✎ **乘积密码**：Shannon提出的一种强化密码的方法
- ✎ 例如，设有两个子系统 E_1 , E_2 ，乘积密码先以 E_1 对明文进行加密后得到中间加密结果，然后再以 E_2 对该中间结果进行加密，得到最终加密结果，表示为

$$E = E_2 \cdot E_1$$

利用这种方法，可将简单易实现的密码组合称复杂的更安全的密码

✎ 定义：如果密码 S 和自身做乘积之后得到的 S^2 同于 S ，则把 S 称为**幂等密码**

✎ 幂等密码的乘积不会增强安全性

✎ 例如，加法密码的乘积仍然是加法密码

✎ 例如，乘法密码的乘积仍然是乘法密码

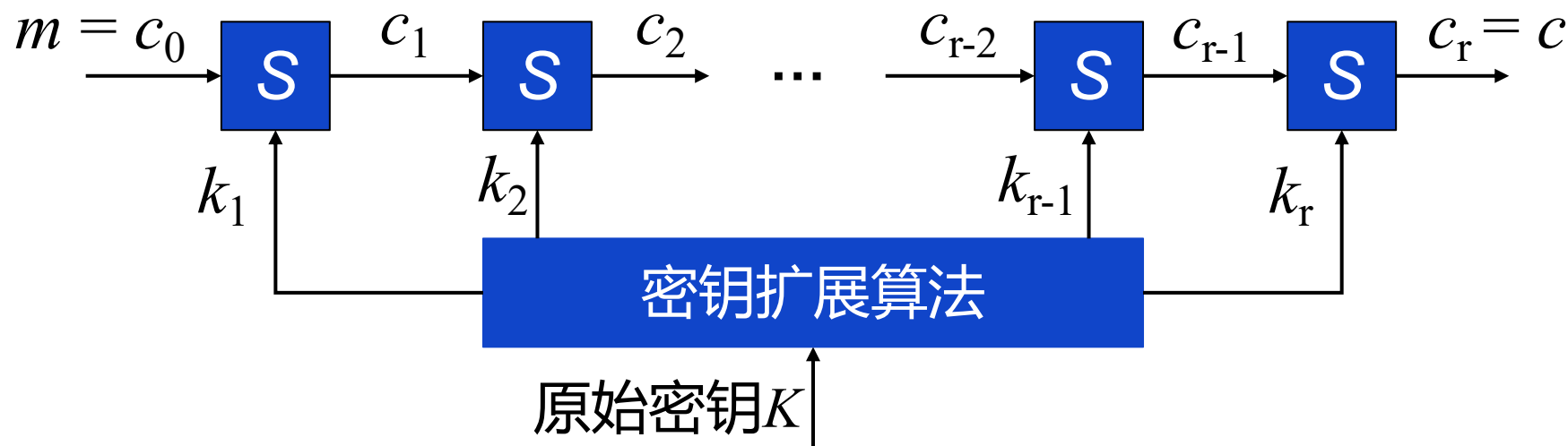
$$\begin{aligned} f(a_i) &= b_i = a_j \\ j &= i + k \bmod n \end{aligned}$$

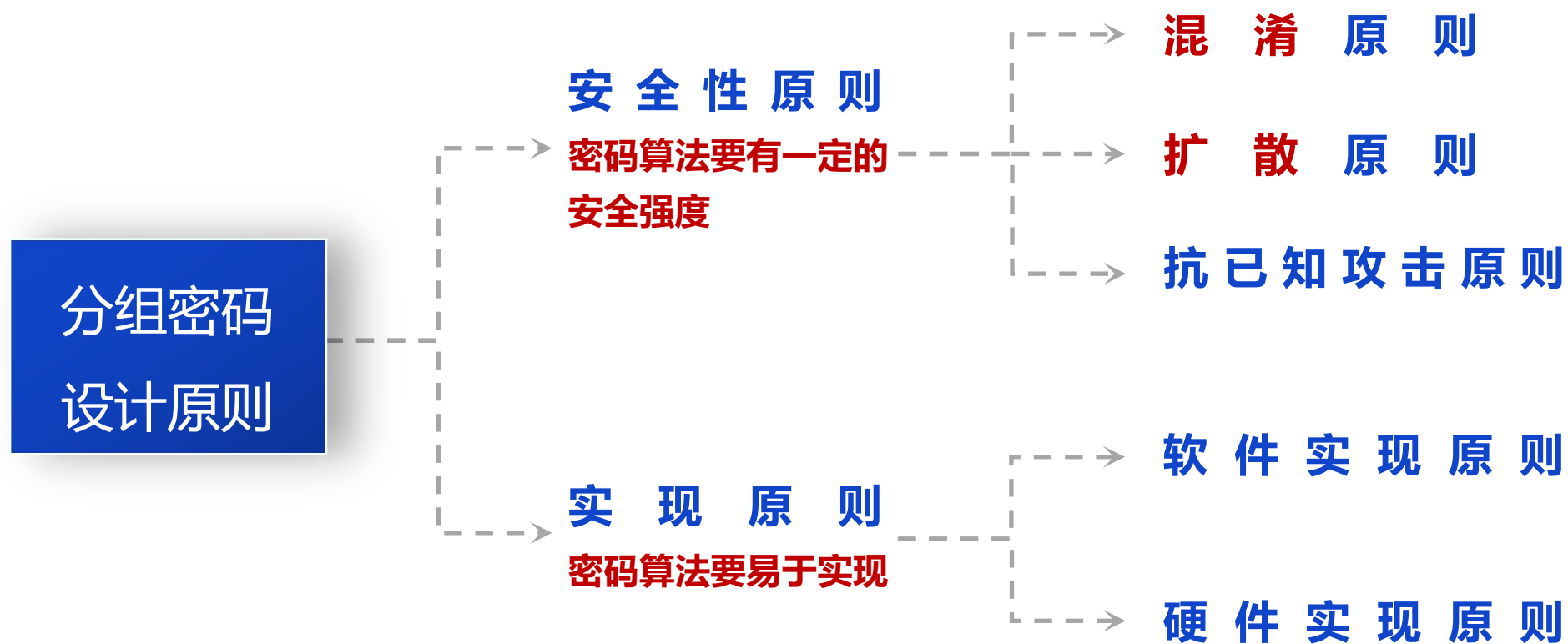
其中， $a_i \in A$ ， k 是满足 $0 < k < n$ 的正整数

$$\begin{aligned} f(a_i) &= b_i = a_j \\ j &= i * k \bmod n \end{aligned}$$

其中， $a_i \in A$ ， k 是满足 $0 < k < n$ 的正整数，且 k 与 n 互素

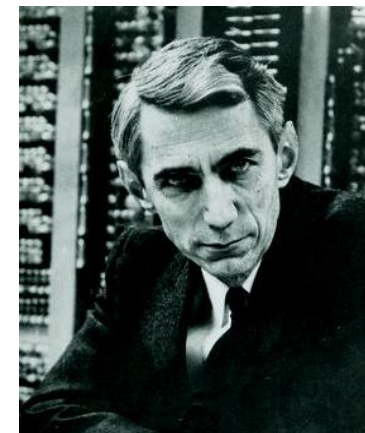
- 定义：对于非幂等的密码体制 S ，将自身做 n 次乘积得到的密码 S^n 称为 S 的 n 重迭代密码
 - 迭代密码可以通过简单密码得到高强度密码
 - 迭代密码是现代分组密码和杂凑函数的核心设计思想
- 迭代型分组密码将原始密钥经密钥扩展算法得到多个轮密钥，每一轮使用一个轮密钥





4.1.3(1) 分组密码设计原则

- ✦ 混淆 (Confusion) 原则：要求所设计的密码应该是**密钥和密文之间的依赖关系尽可能复杂**，以至于无法被攻击者利用
- ✦ 扩散 (Diffusion) 原则：**明文的每个比特位影响密文尽可能多的比特位**，输入微小的变化导致输出多位变化



Claude E. Shannon

在密码学当中，**混淆** (confusion) 与**扩散** (diffusion) 是设计密码学算法的两种主要方法。这样的定义最早出现在克劳德·香农1945年的论文《密码学的数学理论》当中。

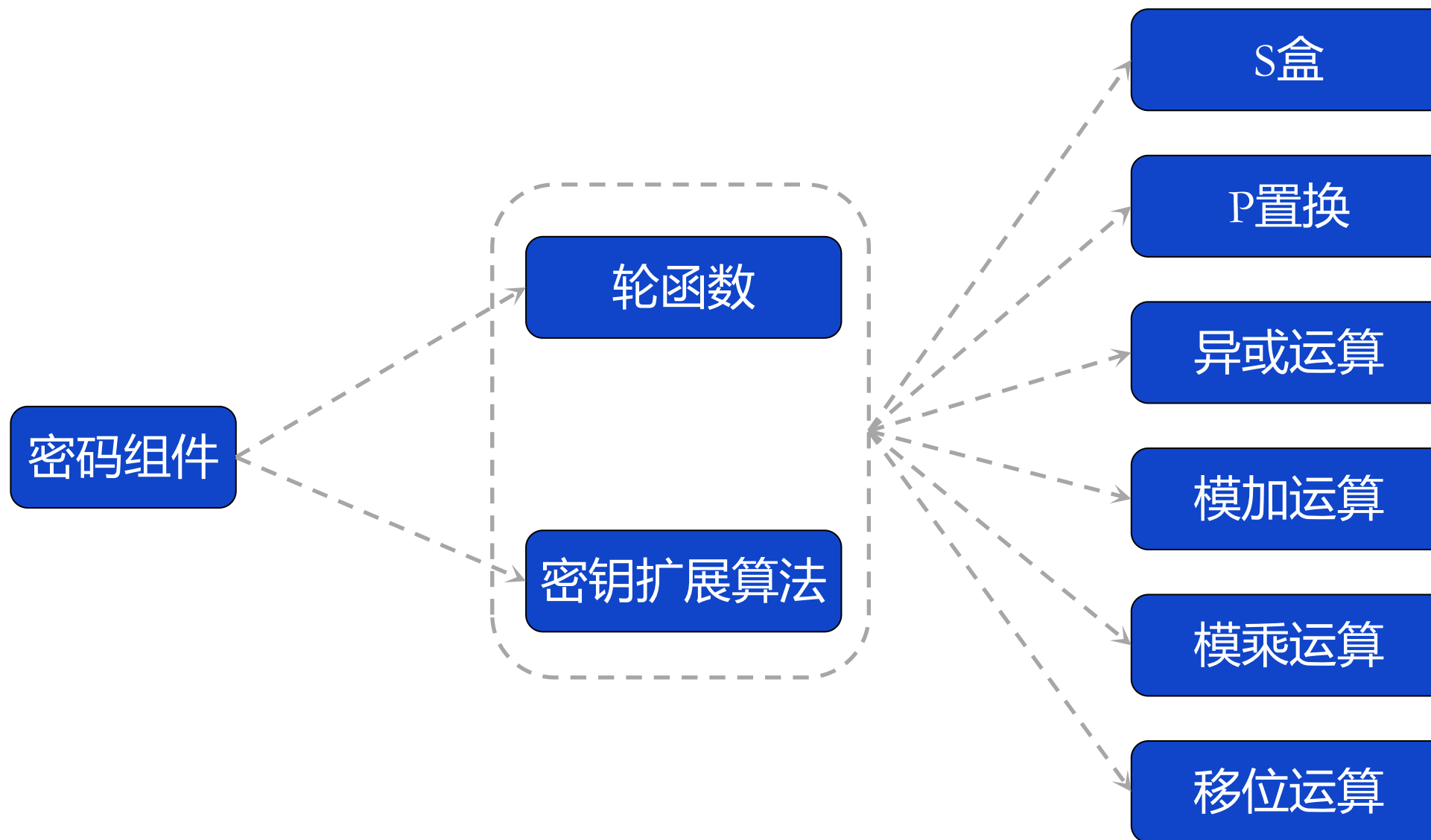
在克劳德·香农的定义之中，混淆主要是用来使密文和对称式加密方法中密钥的关系变得尽可能的复杂；扩散则主要是用来使用明文和密文的关系变得尽可能的复杂，明文中任何一点小更动都会使得密文有很大的差异。

混淆用于掩盖**密钥与密文之间的关系**。这可以挫败通过研究密文以获取冗余度和统计模式的企图。做到这一点最容易的方法是“**代替**”。

扩散通过将明文冗余度分散到密文中使之分散开来。即将**单个明文比特的影响尽可能扩大到更多的密文比特中去**。产生扩散最简单的方法是**换位 (置换)**。

- ✦ 理解混淆和扩散的含义，准确掌握混淆和扩散描述的是谁和谁之间的关系
- ✦ 在唯密文攻击模型下：掌握密文能够获得尽量少的关于密钥和明文的信息，通过互信息来度量

- ✎ 软件实现原则：密码算法应尽可能使用子块和简单运算
 - ✎ 例如，模加运算、移位运算或异或运算
 - ✎ 基本运算同时易于在8位，16位，32位计算平台上实现
- ✎ 硬件实现原则：密码算法应尽量保证加密和解密的相似性
 - ✎ 加密和解密的过程应该仅仅是密钥的使用方式不同
 - ✎ 使用同样的电子元器件既可用于加密又可用于解密



- ✦ 密码算法盒中提供混淆用的非线性部件
- ✦ 分组密码的安全强度，特别是抗差分密码分析和线性密码分析的能力，与S盒紧密相关
- ✦ 通常，S盒规模越大，密码算法的非线性程度越高，密码算法的混淆性能就越好
- ✦ S盒规模越大，密码算法实现的效率就越低

ZUC密码的非线性函数F中S盒的规模是？



- ✎ 代替-置换结构的分组密码算法中，P置换一般位于S盒之后
- ✎ 将S盒的混淆效应扩散开来
- ✎ 进一步提高算法的混淆程度
- ✎ P置换一般设计为线性置换

- ✎ 轮函数是指迭代分组密码算法中单轮加密的非线性函数
- ✎ 轮函数设计时，主要考虑如何快速实现密钥和明文的混淆和扩散
- ✎ 设计原则：
 - ✎ 安全性：抗现有各种攻击
 - ✎ 速度：轮函数的复杂性和轮数决定了算法的加解密速度
 - ✎ 灵活性：密码算法便于在多计算平台上实现

- ✦ 为迭代分组密码的各轮生成轮密钥（子密钥）
- ✦ 轮函数的功能是在轮密钥控制下实现的
- ✦ 轮密钥生成原则：
 - ✦ 密钥与密文独立
 - ✦ 轮密钥比特之间的统计关系是难以计算的
 - ✦ 没有弱密钥
 - ✦ 结构尽量简单，便于实现
 - ✦ 种子密钥对轮密钥每比特的影响要均衡



✎ 非线性代替S层实现分组小块的混淆和扩散

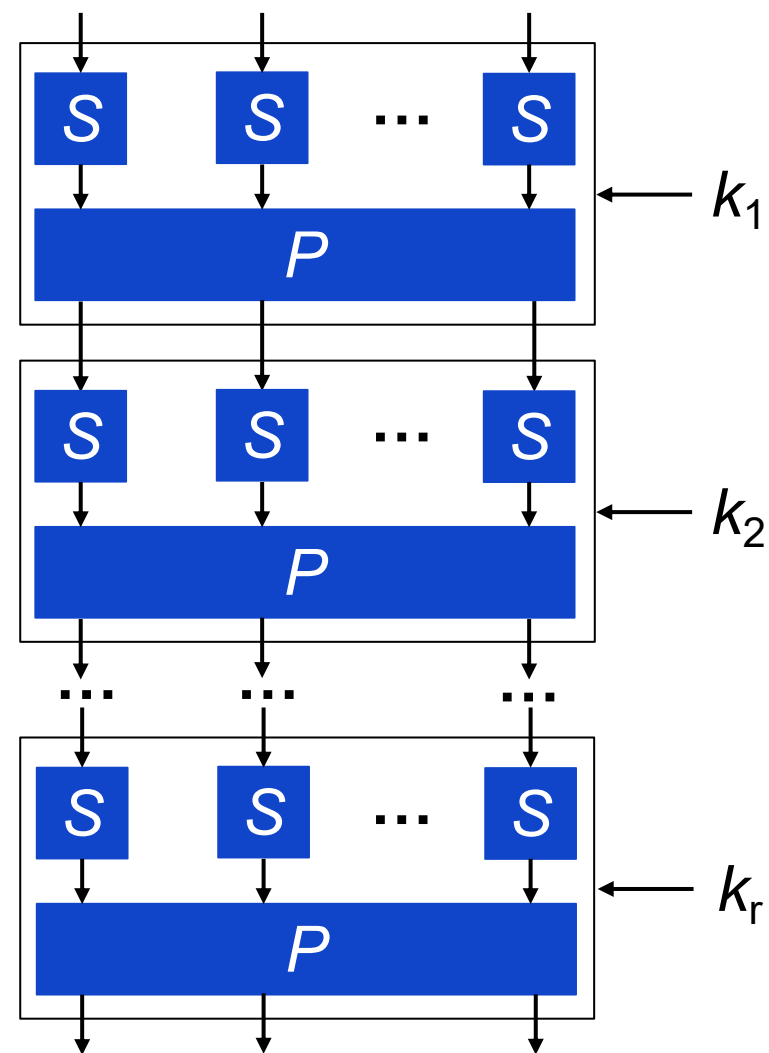
✎ 置换P层实现整体扩散

✎ S-P网络的特点

✎ 结构简单

✎ 扩散速度快

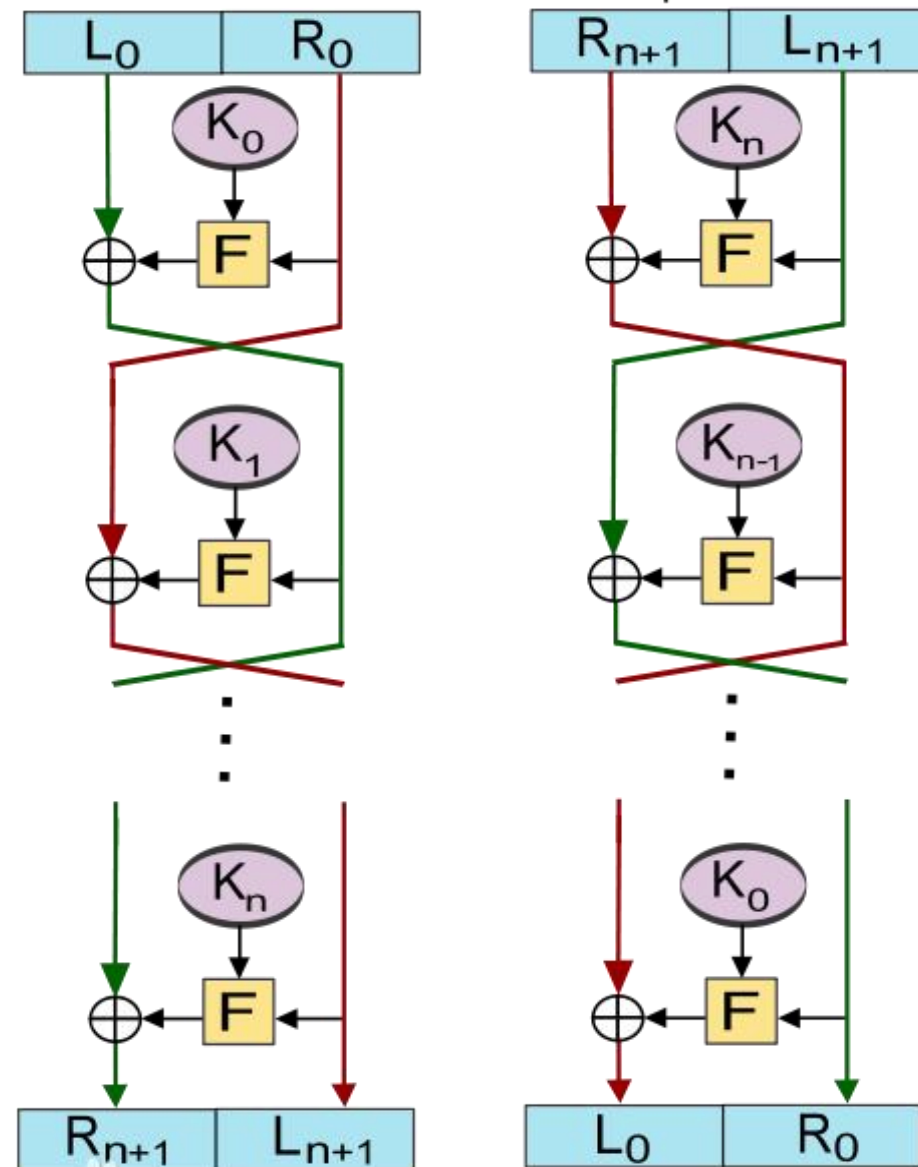
✎ 加解密结构不同



4.1.4(2) Feistel结构

- Feistel模型因在DES等分组密码中广泛应用而著名
- 长度为 $2w$ 的分组分为左右两半各 w 比特
- 右半部分的数据在轮密钥 k 的作用下进行 f 变换
- f 变换结果与左半部分异或，产生下一轮的右半部分
- 原右半部分直接作为下一轮的左半部分
- 最后一轮不做左右对换

$$\begin{cases} R_i = f(R_{i-1}, k_i) \oplus L_{i-1} \\ L_i = R_{i-1} \\ i = 1, 2, \dots, r \end{cases}$$



章节安排

Outline



分组密码概述



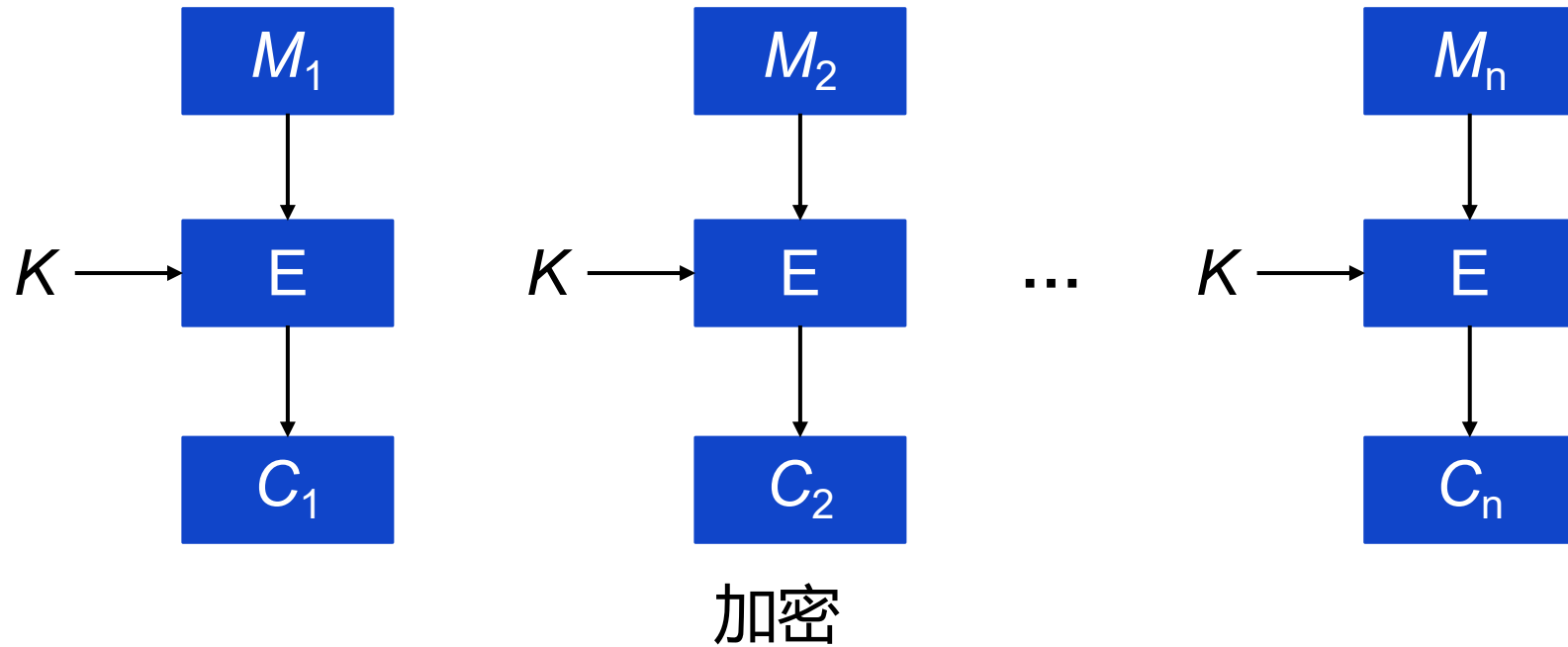
分组密码工作模式

加密模式		特 点
Electronic Code Book(ECB)	电子密码本模式	简单快速，可并行计算
Cipher Block Chaining(CBC)	密码分组链接模式	仅解密支持并行计算
Cipher Feedback Mode(CFB)	密文反馈模式	仅解密支持并行计算
Output Feedback Mode(OFB)	输出反馈模式	不支持并行运算
Counter (CTR)	计数器模式	支持并行计算

4.2.1(1) 电子密码本 (ECB) 模式

ECB mode

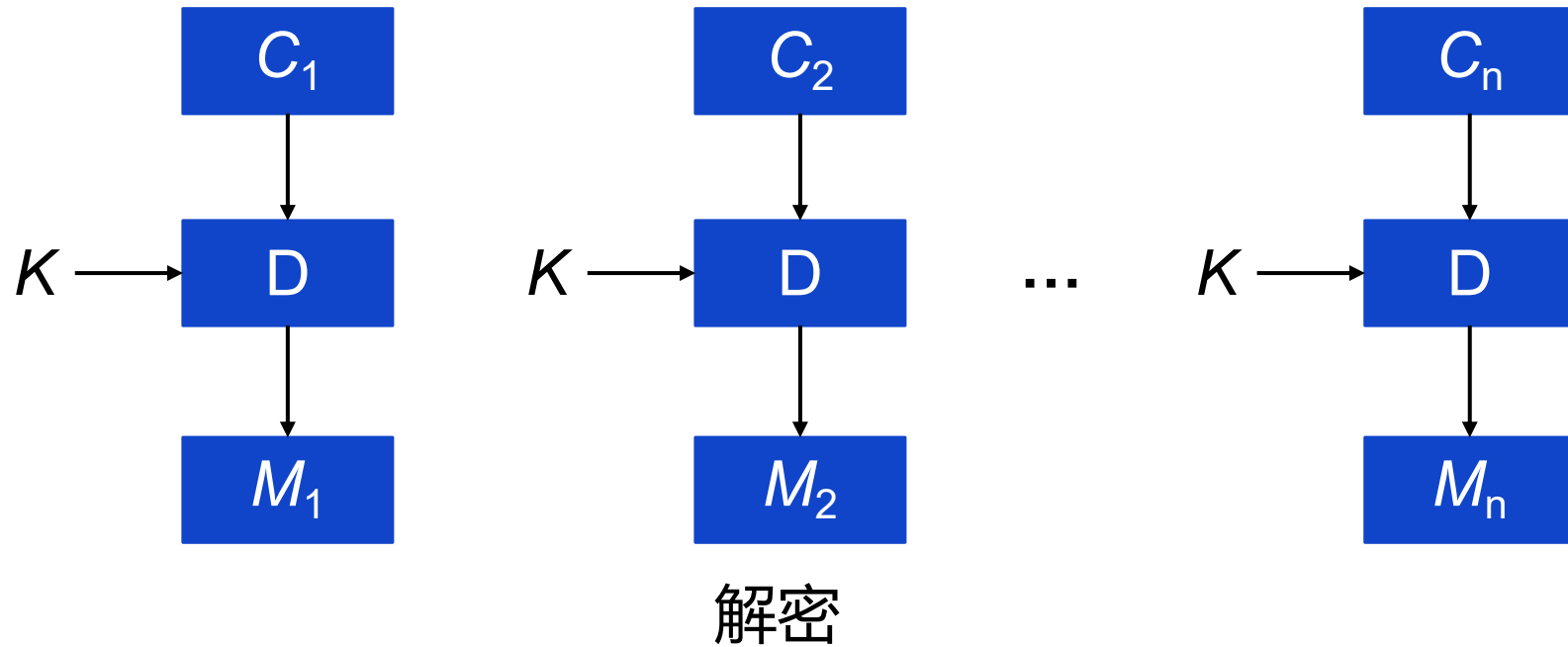
- ✎ 最早采用和**最简单的模式**，它将明文分成若干组，然后每组都用相同的密钥进行加密
- ✎ **相同的明文会产生相同的密文**
- ✎ 用途：传送短数据（如一个加密密钥）



4.2.1(1) 电子密码本 (ECB) 模式

ECB mode

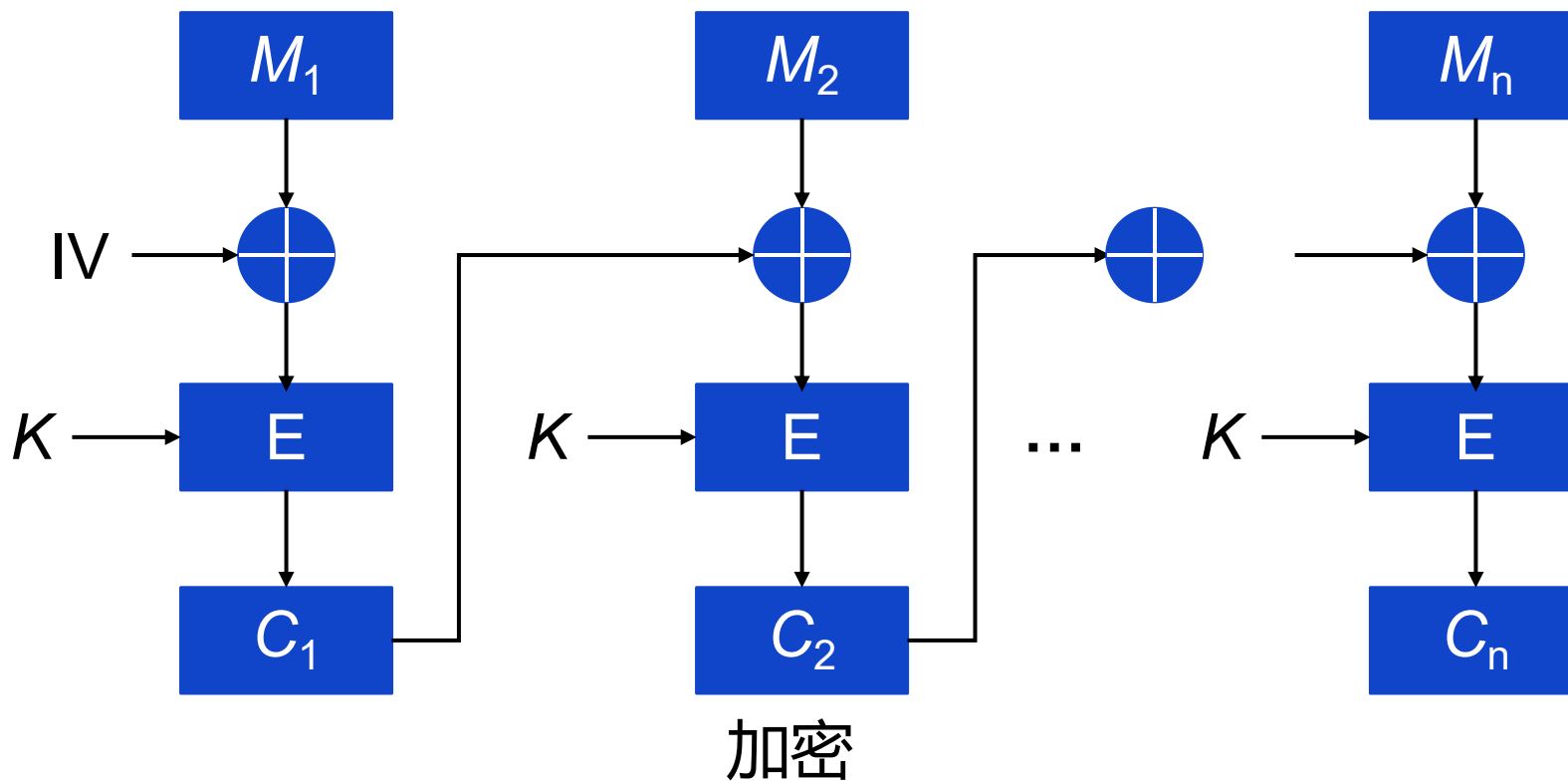
- ✎ 最早采用和**最简单的模式**，它将明文分成若干组，然后每组都用相同的密钥进行加密
- ✎ **相同的明文会产生相同的密文**
- ✎ 用途：传送短数据（如一个加密密钥）



4.2.1(2) 密码分组链接 (CBC) 模式

CBC mode

- ✎ 初始化向量IV参与计算第一组密文，然后与第二组明文异或后加密产生第二组密文
- ✎ 安全性较好，TLS、IPSec等协议的推荐模式，但不利于并行运算
- ✎ 用途：传送数据分组；认证

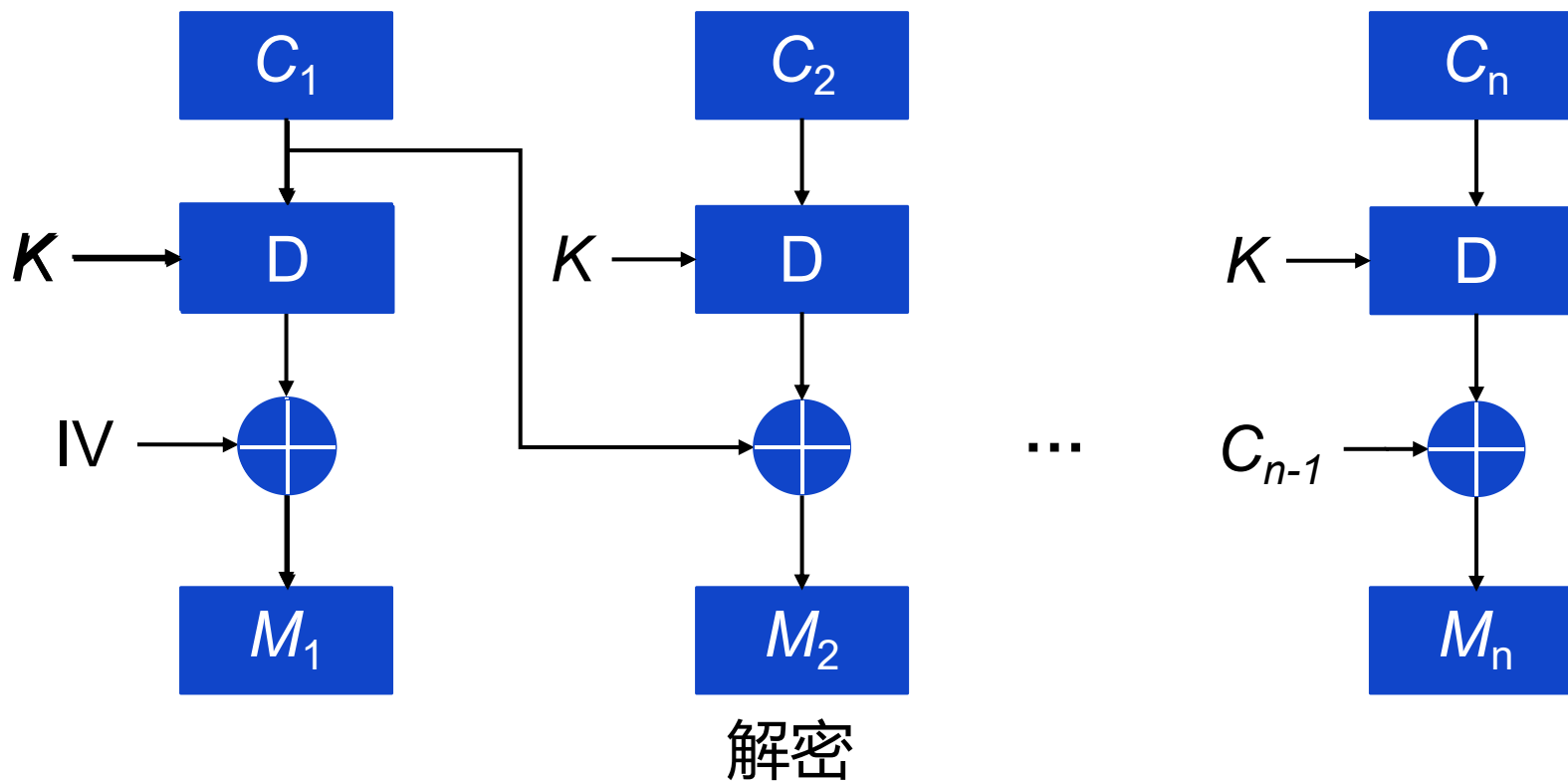


- ✎ 与ECB模式对照来理解CBC模式，增加了反馈
- ✎ 加密无法并行，解密可并行；相同明文的加密结果不同，安全性更高

4.2.1(2) 密码分组链接 (CBC) 模式

CBC mode

- ✎ 初始化向量IV参与计算第一组密文，然后与第二组明文异或后加密产生第二组密文
- ✎ 安全性较好，TLS、IPSec等协议的推荐模式，但不利于并行运算
- ✎ 用途：传送数据分组；认证

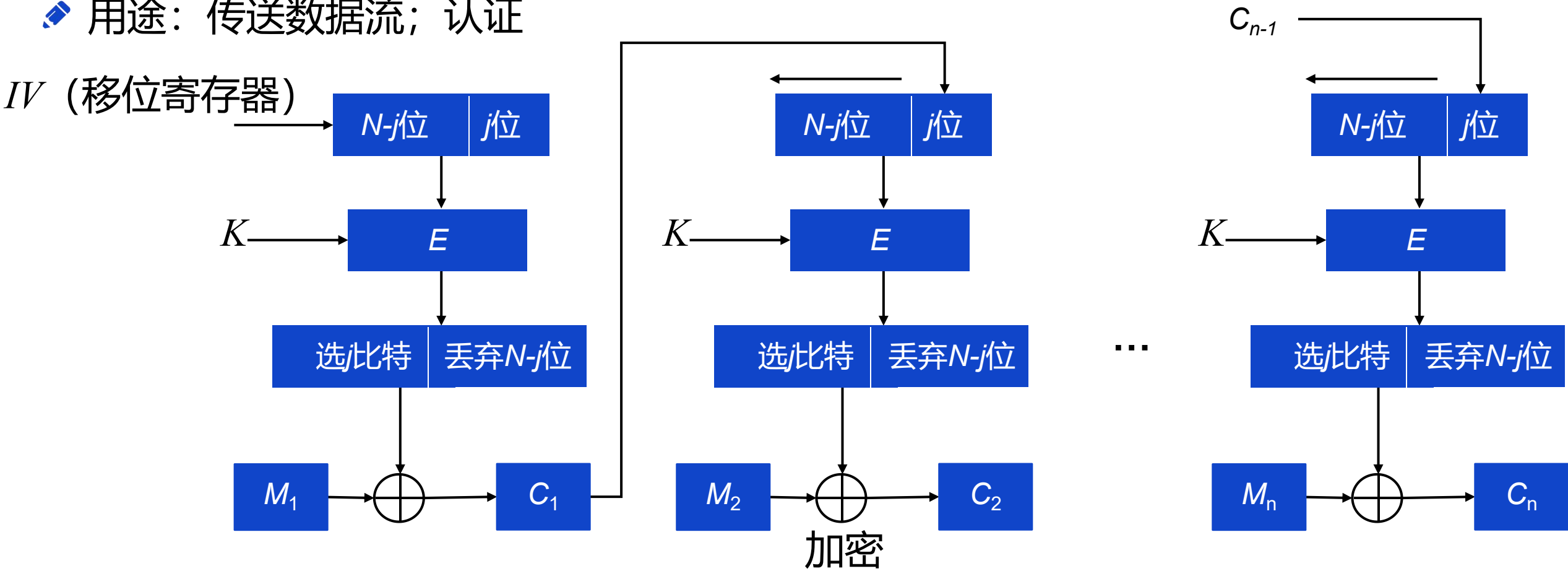


- ✎ 与ECB模式对照来理解CBC模式，增加了反馈
- ✎ 加密无法并行，解密可并行；相同明文的加密结果不同，安全性更高

4.2.1(3) 密文反馈 (CFB) 模式

CFB mode

- ✦ 初始化向量IV加密生成初始化密文，初始化密文与明文进行异或运算，产生密文 C_1
- ✦ 向量IV左移n位，最右侧填入密文 C 的最高j位
- ✦ 用途：传送数据流；认证

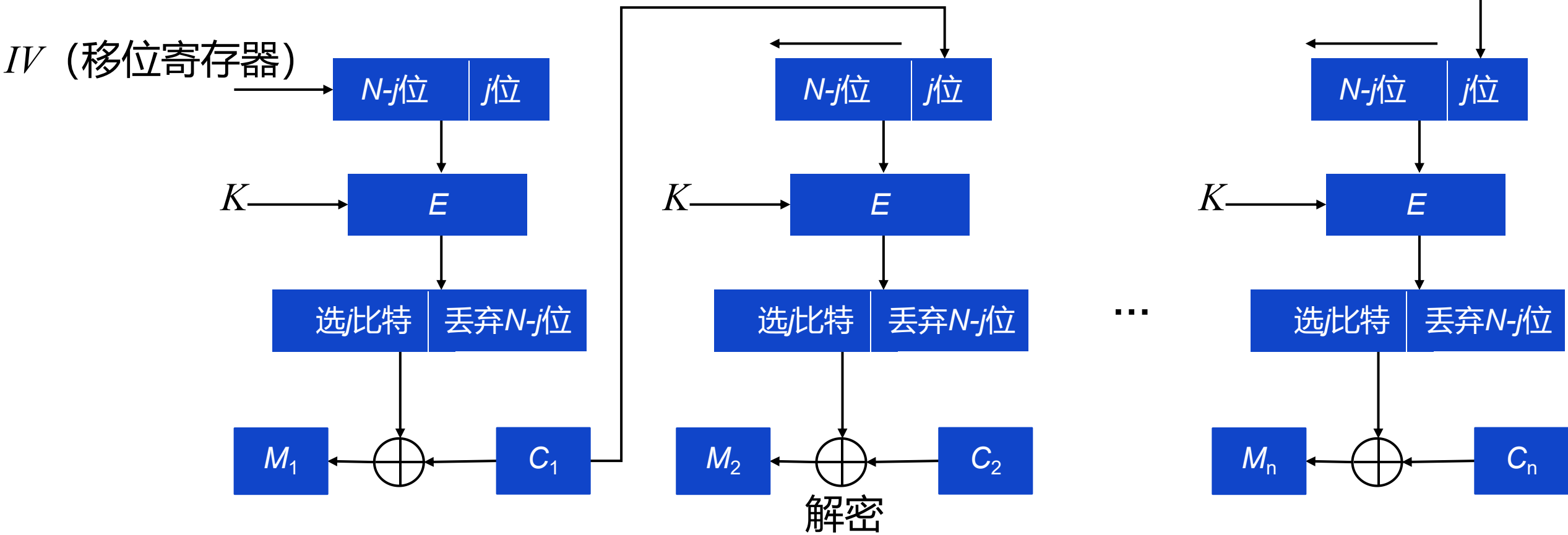


✦ 准确把握反馈位置，与OFB对照

4.2.1(3) 密文反馈 (CFB) 模式

CFB mode

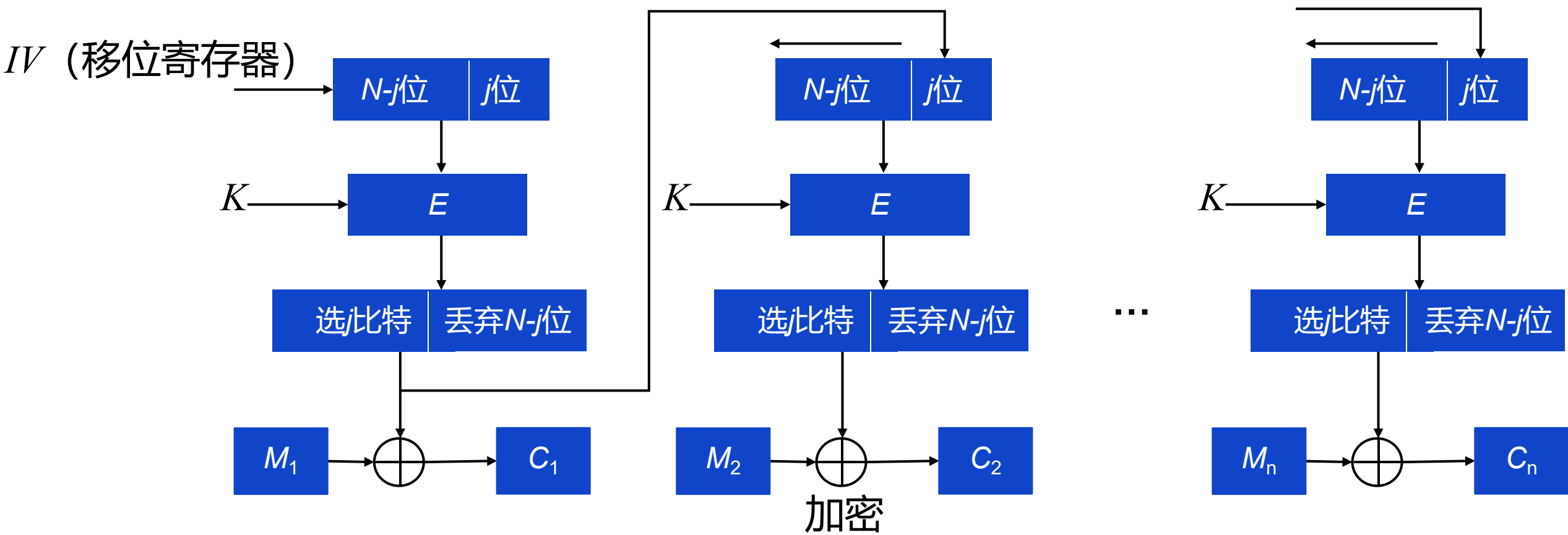
- ✦ 初始化向量IV加密生成初始化密文，初始化密文与明文进行异或运算，产生密文 C_1
- ✦ 向量IV左移n位，最右侧填入密文 C 的最高j位
- ✦ 用途：传送数据流；认证



✦ 准确把握反馈位置，与OFB对照

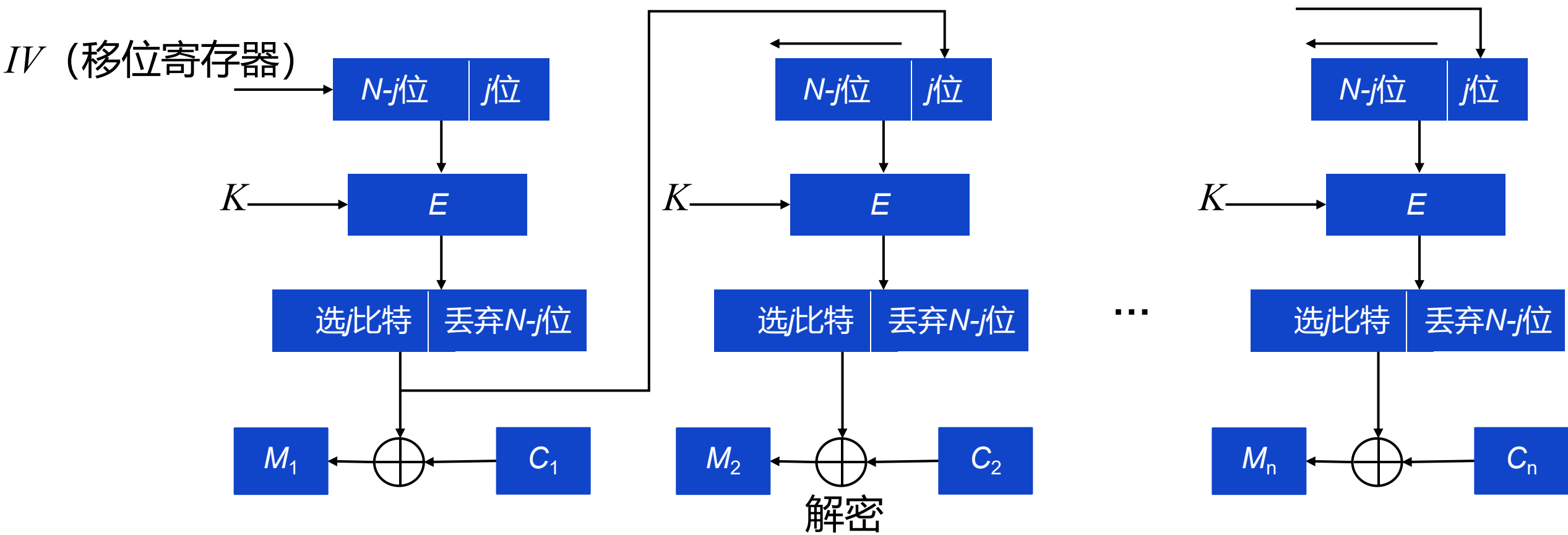
4.2.1(4) 输出反馈 (OFB) 模式

- 与CFB模式相似，差别是，CFB中密文填入加密过程下一阶段，而在OFB模式中，初始化向量加密后输出的最高j位反馈到移位寄存器
- 用途：有扰信道上（如卫星通信）传送数据流



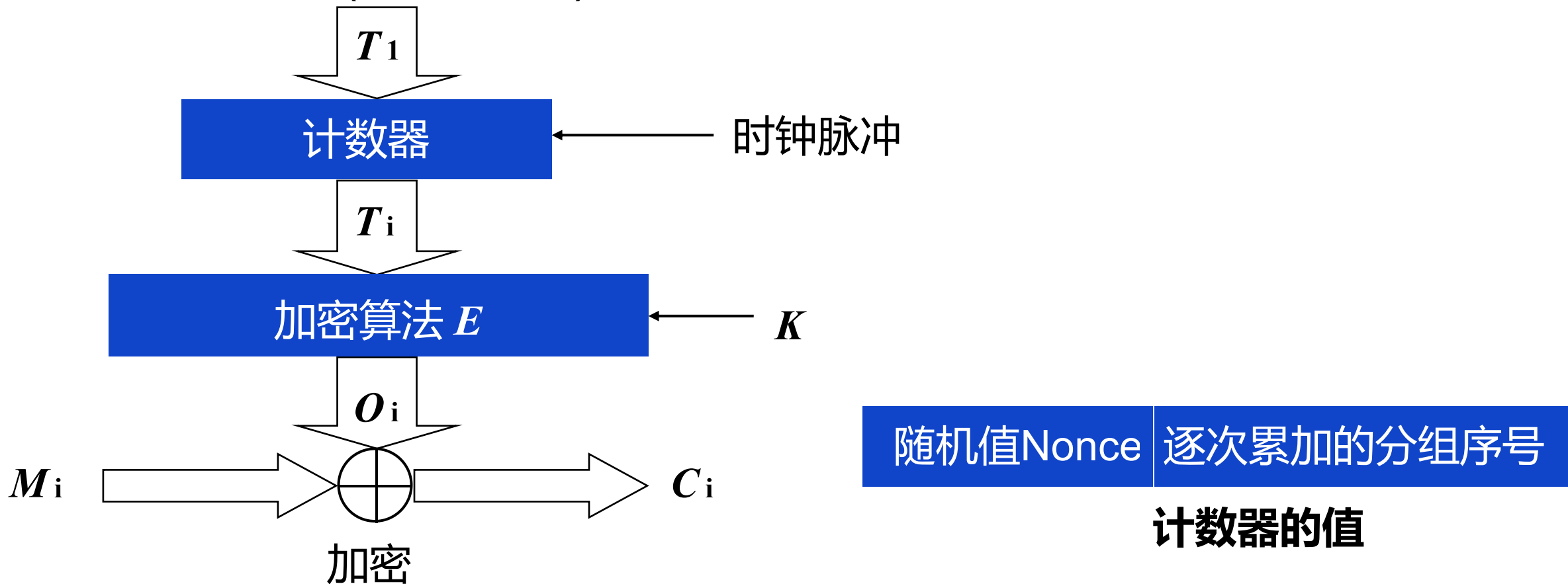
4.2.1(4) 输出反馈 (OFB) 模式

- 与CFB模式相似，差别是，CFB中密文填入加密过程下一阶段，而在OFB模式中，初始化向量加密后输出的最高j位反馈到移位寄存器
- 用途：有扰信道上（如卫星通信）传送数据流



4.2.1(5) 计数器 (CTR) 模式

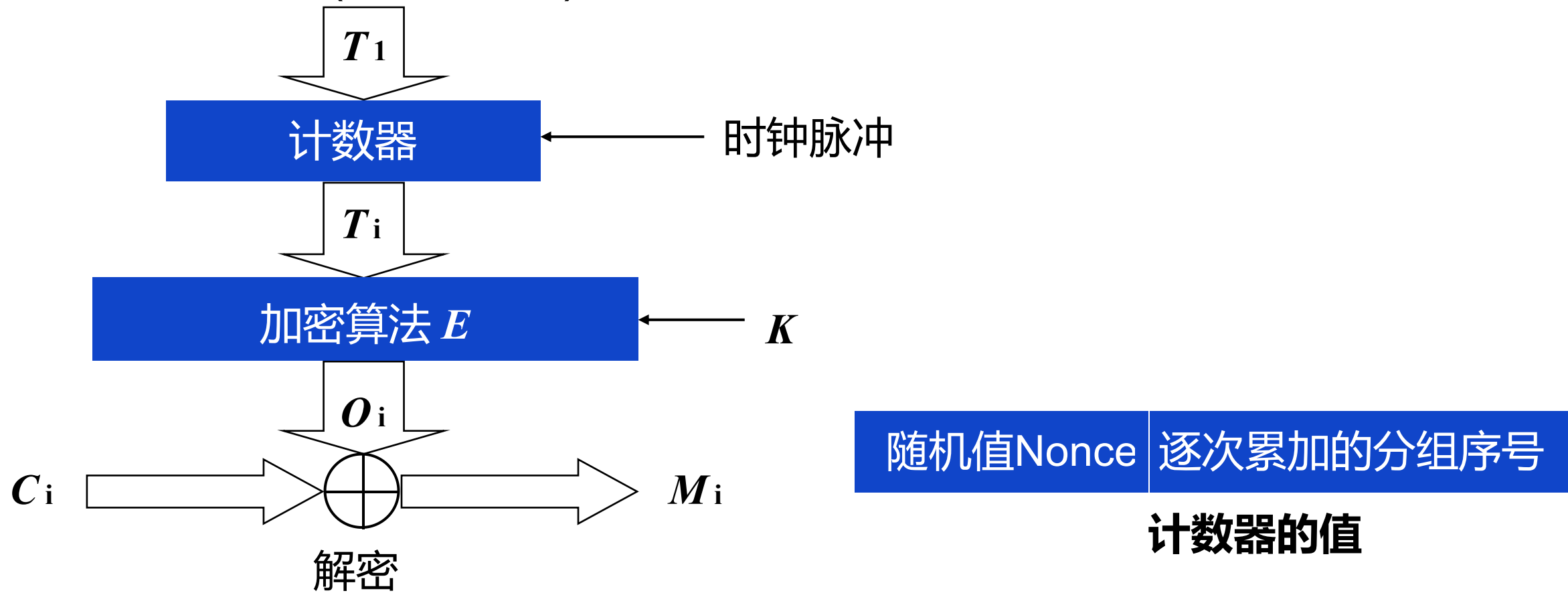
- ✎ CTR模式中，有一个自增的算子，算子用密钥K加密之后和明文异或得到密文，**相当于一次一密**；简单快速，安全可靠，而且可以并行加密
- ✎ 用途：有扰信道上（如卫星通信）传送数据流



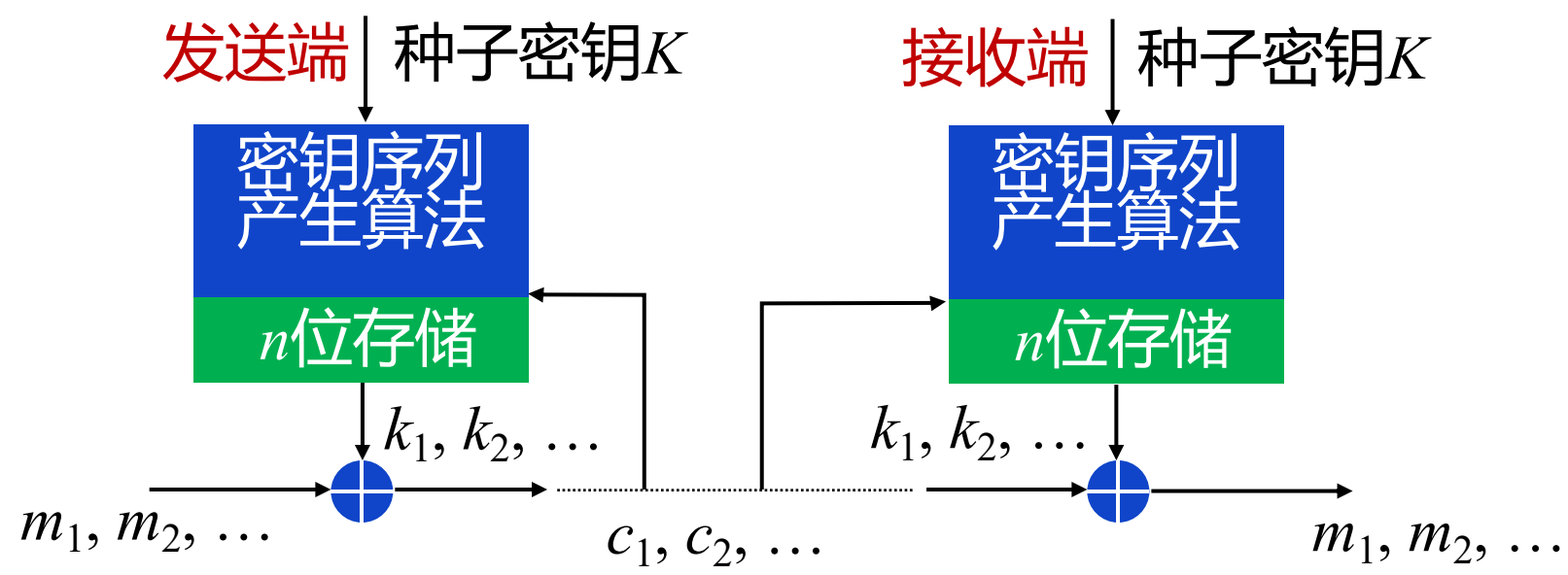
✎ 计数器模式最接近 “一次一密”，可并行，效率也很高

4.2.1(5) 计数器 (CTR) 模式

- ✎ CTR模式中，有一个自增的算子，算子用密钥K加密之后和明文异或得到密文，**相当于一次一密**；简单快速，安全可靠，而且可以并行加密
- ✎ 用途：有扰信道上（如卫星通信）传送数据流



✎ 计数器模式最接近 “一次一密”，可并行，效率也很高



混淆 (Confusion) 与扩散 (Diffusion)

S-P网络和Feistel结构

分组密码的工作模式

在密码学当中，**混淆** (*confusion*) 与**扩散** (*diffusion*) 是设计密码学算法的两种主要方法。这样的定义最早出现在克劳德·香农1945年的论文《密码学的数学理论》当中。

在克劳德·香农的定义之中，混淆主要是用来使密文和对称式加密方法中密钥的关系变得尽可能的复杂；扩散则主要是用来使用明文和密文的关系变得尽可能的复杂，明文中任何一点小更动都会使得密文有很大的差异。

混淆用于掩盖明文与密文之间的关系。这可以挫败通过研究密文以获取冗余度和统计模式的企图。做到这一点最容易的方法是“代替”。

扩散通过将明文冗余度分散到密文中使之分散开来。即将单个明文或密钥位的影响尽可能扩大到更多的密文去。产生扩散最简单的方法是换位（置换）。

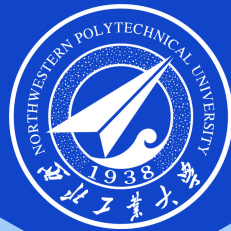
课后阅读

Homework

扩散(diffusion)和混淆(confusion)是C. E. Shannon提出的设计密码体制的两种基本方法，其目的是为了抵抗对手对密码体制的统计分析。在分组密码的设计中，充分利用扩散和混淆，可以有效地抵抗对手从密文的统计特性推测明文或密钥。扩散和混淆是现代分组密码的设计基础。

所谓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。当然，理想的情况是让明文中的每一位影响密文中的所有位，或者说让密文中的每一位受明文中所有位的影响。

所谓混淆就是将密文与密钥之间的统计关系变得尽可能复杂，使得对手即使获取了关于密文的一些统计特性，也无法推测密钥。使用复杂的非线性代替变换可以达到比较好的混淆效果，而简单的线性代替变换得到的混淆效果则不理想。可以用“揉面团”来形象地比喻扩散和混淆。当然，这个“揉面团”的过程应该是可逆的。乘积和迭代有助于实现扩散和混淆。选择某些较简单的受密钥控制的密码变换，通过乘积和迭代可以取得比较好的扩散和混淆的效果。



感谢聆听!

THANK YOU FOR YOUR ATTENTION!