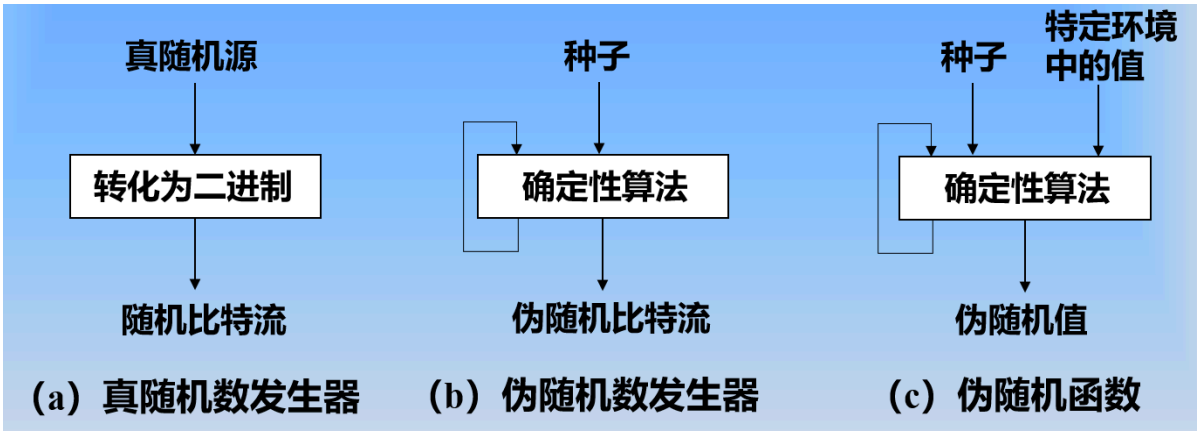
随机数与流密码

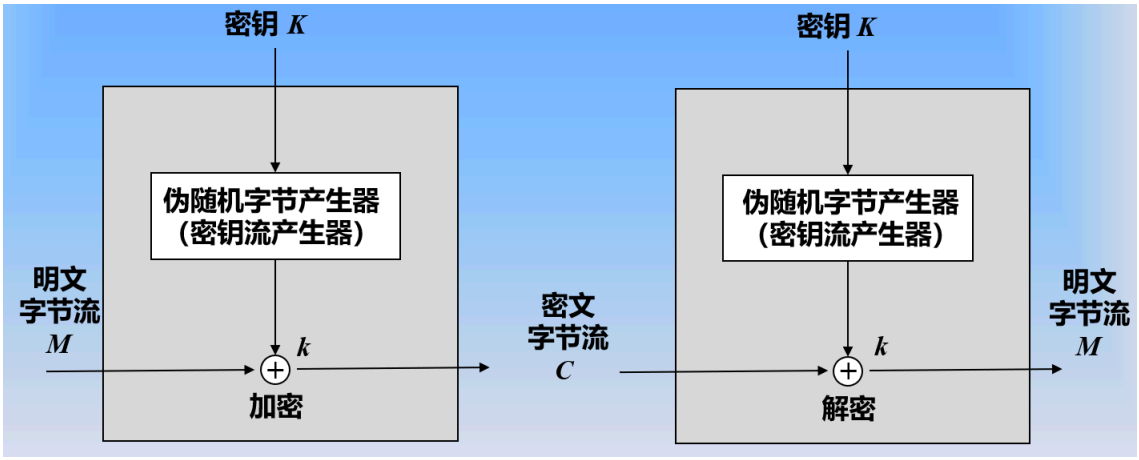
随机数（非重点）

- 真随机数无法重现，由一个有效的随机源（熵源）作为输入。
- 伪随机数可以通过随机性测试，但是可以通过确定性的算法重现，例如随机种子产生的随机数。

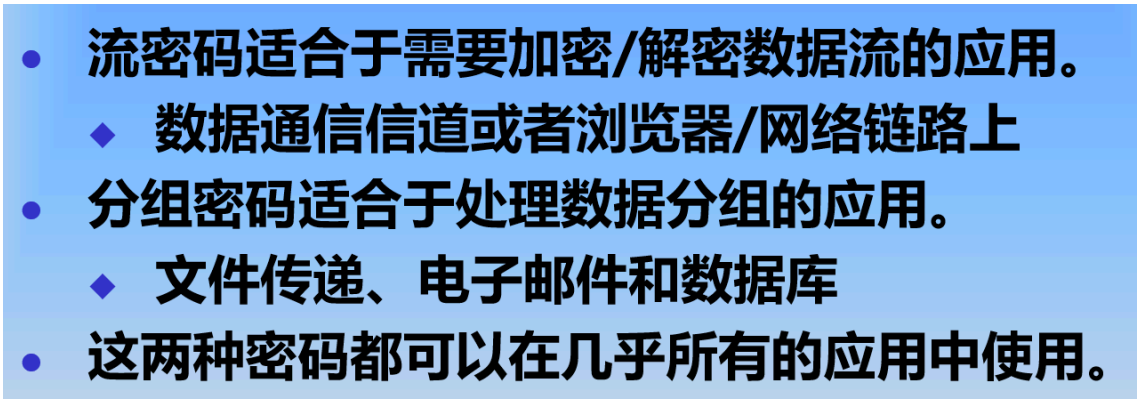


流密码（了解）

- 定义：流密码连续处理输入元素，在运行过程中一次产生一个输出元素，效率更快。
- 应用：SSL/TLS(安全套接字层/传输层安全)、WEP（有线等效保密）协议和Wi-Fi保护访问（WPA）协议
- 结构：其中的密钥K应该有一个长周期，接近随机数的性质，同时不可重复使用密钥K。



- 应用场景：



RC4（重点了解）

RC4算法非常简单，用一个可变长度为1~256字节(8~2048比特)的密钥来初始化256字节的状态向量S，其元素为$S[0], S[1], \ldots, S[255]$。

- **置换后的S从始至终包含从0到255的所有8比特数值**
- **加密/解密时从S的256个元素中选择一个作为密钥**
- **每次产生密钥后，都需要重新排列S的元素**

- 性质：具有可变密钥长度，使用面向字节的操作
- 算法流程：

1. 初始化S

S的256个元素按升序分别设置为0~255，然后创建一个临时向量T，根据keylen字节长度的密钥K循环赋值给T，即：

for $i = 0$ to 255 do
    $S[i] = i$
    $T[i] = K[i \bmod \text{keylen}]$

2. 初始置换S

用T产生S的初始置换，从$S[0]$~$S[255]$，对每个$S[i]$，根据$T[i]$确定的标签值，并将$S[i]$置换为S的另一字节，即

$j = 0$
for $i = 0$ to 255 do
    $j = (j + S[i] + T[i]) \pmod{256}$
    swap $(S[i], S[j])$ //数值置换

3. 密钥流产生以及加密

对每个$S[i]$，根据$S[i]$的标签值，将$S[i]$与S的另一字节置换，再依此选出的密钥值$S[t]$与明文异或，即

$i = j = 0$
for each message byte m_i
    $i = (i + 1) \pmod{256}$
    $j = (j + S[i]) \pmod{256}$
    swap$(S[i], S[j])$ // 数值置换
    $t = (S[i] + S[j]) \pmod{256}$
    $c_i = m_i \oplus S[t]$