视频讲解

# 第3章 密码学基础理论

密码学研究进行保密通信和如何实现信息保密的问题，具体指通信保密传输和信息存储加密等。它以认识密码变换的本质、研究密码保密与破译的基本规律为对象，以可靠的数学方法和理论为基础，对解决信息安全中的机密性、数据完整性、认证和身份识别，以及对信息的可控性及不可抵赖性等问题提供系统的理论、方法和技术。密码学包括密码编码学和密码分析学两个分支。密码编码学研究对信息进行编码，实现对信息的隐藏。密码分析学研究加密消息的破译或消息的伪造。密码学的发展历史比较悠久，整个密码学的发展是由简单到复杂的逐步完善过程，也促进了数学、计算机科学、信息通信等学科的发展。

## 3.1 密码学概述

### 3.1.1 基本概念

明文(Plaintext/Message)：待加密的信息，用 $P$ 或 $M$ 表示。明文可以是文本文件、图形、数字化存储的语音流或数字化的视频图像的比特流等。

密文(Ciphertext)：明文经过加密处理后的形式，用 $C$ 表示。

加密(Encryption)：用某种方法伪装消息以隐藏它的内容的过程。

解密(Decryption)：把密文转换成明文的过程，加密过程对应的逆过程。

密钥(Key)：变换函数所用的一个控制参数。加密和解密算法的操作通常是在一组密钥控制下进行的，分别称为加密密钥和解密密钥，通常用 $K$ 表示。

加密算法(Encryption Algorithm)：将明文变换为密文的变换函数，通常用 $E$ 表示。

解密算法(Decryption Algorithm)：将密文变换为明文的变换函数，通常用 $D$ 表示。

密码分析(Cryptanalysis)：截获密文者试图通过分析截获的密文从而推断出原来的明文或密钥的过程。

被动攻击(Passive Attack)：对一个保密系统采取截获密文并对其进行分析和攻击。这种攻击对密文没有破坏作用。

主动攻击(Active Attack)：攻击者非法侵入一个密码系统，采用伪造、修改、删除等手段向系统注入假消息进行欺骗。这种攻击对密文具有破坏作用。

密码系统(Cryptosystem)：用于加密和解密的系统。加密时，系统输入明文和加密密钥，加密变换后，输出密文；解密时，系统输入密文和解密密钥，解密变换后，输出明文。

单向函数(One-way Function)：单向函数的计算是不可逆的。给定任意两个集合 $X$ 和 $Y$。函数 $f:X\to Y$ 称为单向的，对每个 $x$ 属于 $X$，很容易计算出函数 $f(x)$ 的值，而对大多数 $y$ 属于 $Y$，要确定满足 $y=f(x)$ 的 $x$ 计算比较困难(假设至少有这样一个 $x$ 存在)。

单向陷门函数(One-way Trap Door Function)：一类特殊的单向函数，它包含一个秘密陷门。在不知道该秘密陷门的情况下，计算函数的逆是非常困难的。若知道该秘密陷门，计算函数的逆就非常简单。单向陷门函数满足下面三个条件。

(1) 对 $f(x)$ 的定义域中的每一个，均存在函数 $f^{-1}(x)$，使得 $f(f^{-1}(x))=$

$f^{-1}(f(x))=x$。

（2）$f(x)$与 $f^{-1}(x)$都很容易计算。

（3）仅根据已知的 $f(x)$计算 $f^{-1}(x)$非常困难。

### 3.1.2 基本原理

密码系统通常由明文、密文、密钥（包括加密密钥和解密密钥）与密码算法（包括加密算法和解密算法）四个基本要素组成。一个密码体制可以用五元组$(M,C,K,E,D)$来定义，该五元组应满足以下条件。

（1）明文空间：$M$ 是可能明文的有限集。由明文 $m$ 的二进制数位数确定，若明文 $m$ 的二进制数位数为 $nm$，则明文集合 $M$ 包含 $2^{nm}$ 个不同的明文。

（2）密文空间：$C$ 是可能密文的有限集。由密文 $c$ 的二进制数位数确定，若密文 $c$ 的二进制数位数为 $nc$，则密文集合 $C$ 包含 $2^{nc}$ 个不同的密文。

（3）密钥空间：$K$ 是可能密钥构成的有限集。加密密钥空间由加密密钥的二进制数位数确定，若加密密钥的二进制数位数为 $nk$，则加密密钥集合 $K$ 包含 $2^{nk}$ 个不同的密钥；解密密钥空间由解密密钥的二进制数位数确定，如果解密密钥的二进制数位数为 $nd$，则解密密钥集合 $K$ 包含 $2^{nd}$ 个不同的密钥。

（4）加密算法空间：$E$ 是可能加密算法的有限集。

（5）解密算法空间：$D$ 是可能解密算法的有限集。

明文 $M$ 转换成密文 $c$ 的过程如下：

$$c=E(m,ke)$$

加密过程是以明文 $m$ 和加密密钥 $ke$ 为输入的加密函数运算过程。$c=E(m,ke)$也可以用 $c=E_{ke}(m)$表示。

密文 $c$ 转换成明文 $m$ 的过程如下：

$$m=D(c,kd)$$

加密和解密如图 3-1 所示。

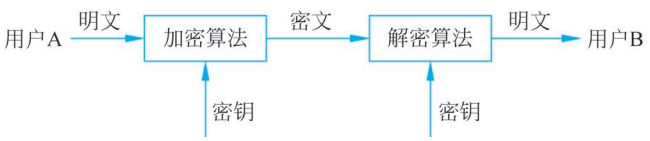


图 3-1 加密和解密

对于任意 $k\in K$，有一个加密算法 $E_k\in E$ 和相应的解密算法 $D_k\in D$，使得 $E_k:M\to C$ 和 $D_k:C\to M$ 分别为加密函数和解密函数，满足 $D_k(E_k(m))=m,x\in M$。

根据柯克霍夫（Kerckhoffs）原则，所有加密解密算法都是公开的，保密的只是密钥。发送端将明文 $m$ 和加密密钥 $ke$ 作为加密函数 $E$ 的输入，加密函数 $E$ 的运算结果是密文 $c$。密文 $c$ 沿着发送端至接收端的传输路径到达接收端。接收端将密文 $c$ 和解密密钥 $kd$ 作为解密函数 $D$ 的输入，解密函数 $D$ 的运算结果是明文 $m$。

### 3.1.3 密码体制分类

根据密钥的特点，密码体制分为对称和非对称密码体制两种，而介于对称和非对称之间的密码体制称为混合密码体制。

1. 对称密码体制

如果加密密钥等于解密密钥，那么这种密钥密码称为对称密码。对称密码又称单钥密码或私钥密码，是指在加解密过程中使用相同或可以推出本质上相同的密钥，即加密与解密密钥相同，且密钥需要保密。信息的发送方和接收方在进行信息的传输与处理时必须共同持有该密钥，因此密钥的安全性成为保证系统机密性的关键。信息的发送方将持有的密钥对要发送的信息进行加密，加密后的密文通过网络传送给接收方，接收方用与发送方相同的私有密钥对接收的密文进行解密，得到信息明文。

对称加密算法的特点是算法公开、计算量小、加密速度快、加密效率高。其不足之处是交付双方都使用同样钥匙，因此安全性得不到保证。此外，每对用户每次使用对称加密算法时都需要使用其他人不知道的唯一密钥，使得发收信双方拥有的密钥数量呈几何级数增长，密钥管理成为用户的负担。对称加密算法在分布式网络系统上使用较为困难，主要是因为密钥管理困难，使用成本较高。对称加密算法与公开密钥加密算法相比，能够提供加密和认证却缺乏了签名功能，使其使用范围缩小。对称密码体制主要算法包括数据加密标准 DES 法，2DES(DDES)算法、三重数据加密算法、高级加密标准 AES 法、Blowfish 算法、RC5 算法、国际数据加密算法 IDEA、SM4 算法等。

2. 非对称密码体制

2015 年，“图灵奖”的得主是前 Sun Microsystems 公司首席安全官菲尔德·迪菲(Whitfield Diffie)和斯坦福大学电气工程系名誉教授马丁·赫尔曼(Martin Hellman)。两位获奖者在发表的论文 *New Directions in Cryptography* 中提出了划时代的公开密钥密码系统的概念，这个概念为密码学的研究开辟了一个新的方向，有效地解决了秘密密钥密码系统中通信双方密钥共享困难的缺点，并引进了创新的数字签名的概念。

非对称密码体制需要：公开密钥(Public Key)和私有密钥(Private Key)。公开密钥与私有密钥是一对，如果用公开密钥对数据进行加密，只有用对应的私有密钥才能解密；如果用私有密钥对数据进行加密，那么只有用对应的公开密钥才能解密，因为加密和解密使用的是两个不同的密钥。

非对称密码体制实现机密信息交换的基本过程：用户 A 生成一对密钥并将其中的一个作为公用密钥向其他方公开；得到该公用密钥的用户 B 使用该密钥对机密信息进行加密后再发送给用户 A；用户 A 再用自己保存的另一个专用密钥对加密后的信息进行解密。另外，用户 A 可以使用用户 B 的公钥对机密信息进行签名后再发送给用户 B；用户 B 再用自己的私匙对数据进行验签。

非对称密码体制的特点：算法强度复杂、安全性依赖算法与密钥；但是由于其算法复杂，加密解密速度没有对称加密解密速度快。对称密码体制中只有一种密钥，并且是非公开的，如果解密就得让对方知道密钥，所以保证其安全性就是保证密钥的安全。而

非对称密钥体制有两种密钥，其中一个是公开的，这样就可以不需要像对称密码那样传输对方的密钥，安全性相对好。

非对称密码体制不要求通信双方事先传递密钥或有任何约定就能完成保密通信，并且密钥管理方便，可防止假冒和抵赖，因此更适合网络通信中的保密通信要求。

非对称密码体制主要算法包括 RSA 算法、Elgamal 算法、背包算法、Rabin 算法、Diffie Hellman 算法、椭圆曲线密码(ECC)算法、概率公钥算法、NTRU 算法、SM2 算法、SM9(标识密码)算法。

3. 混合密码体制

混合密码体制利用非对称密码体制分配私钥密码体制的密钥，消息的收发双方共用这个密钥，然后按照私钥密码体制的方式进行加密和解密运算。混合密码体制的工作流程如下。

(1) 用户 A 用对称密钥把需要发送的消息加密。

(2) 用户 A 用用户 B 的公开密钥将对称密钥加密，形成数字信封，然后一起把加密消息和数字信封传送给用户 B。

(3) 用户 B 收到用户 A 的加密消息和数字信封后，用自己的私钥将数字信封解密，获取用户 A 加密消息时的对称密钥。

(4) 用户 B 使用用户 A 加密的对称密钥把收到的加密消息解开。

### 3.1.4 密码学发展阶段

密码学的发展经历了古典密码学阶段、近代密码学阶段到现代密码学阶段的演变。

1. 古典密码学阶段

古代文明在实践中逐渐发明了密码。从某种意义上讲，战争是密码系统诞生的催化剂，战争提出了安全通信的需求，从而促进了密码的诞生。

早在公元前 440 年，古希腊战争出现了隐写术，奴隶主将信息刺青在奴隶的头皮上，用头发掩盖来达到安全通信的目的。公元前 400 年，斯巴达人也使用了一种塞塔(Scytale)式密码的加密工具，该工具将信息写在缠绕在锥形指挥棒的羊皮上，羊皮解开后信息无法识别，必须绕在同一种指挥棒上才能恢复原始信息。我国古代的藏头诗、藏尾诗、漏格诗以及各种书画，将要表达的真正意思或“密语”隐藏在诗文或画卷中特定位置的记载，一般人只注意诗或画的表面意境，而不会注意或很难发现隐藏其中的“话外之音”。周朝兵书《六韬·龙韬》也记载了密码学的运用，其中的《阴符》和《阴书》记载了姜子牙通过令牌长短给出不同的密令。

这一时期的密码学更像是一门艺术，其核心手段是代换和置换。代换是指明文中的每一个字符被替换成密文中的另一个字符，接收方对密文做反向替换便可恢复出明文；置换是密文和明文字母保持相同，但顺序被打乱。代换密码的著名例子有凯撒(Caesar)密码(公元前 1 世纪)、圆盘密码(15 世纪)、维吉尼亚密码(16 世纪)、Enigma 转轮组加密(1919 年)等。

2. 近代密码学阶段

这一阶段真正开始源于香农在20世纪40年代末发表的一系列论文，特别是1949年的《保密系统通信理论》，使密码学成为一门科学。近代密码发展中一个重要突破是“数据加密标准”(DES)的出现，使密码学得以从政府走向民间。其次，DES密码设计中的很多思想(Feistel结构、S盒等)被后来大多数分组密码采用。

3. 现代密码学阶段

1976年，Diffie和Hellman的“密码学的新方向”，提出了一种在不安全信道上进行密钥协商的协议的“公钥密码”概念。1977年，麻省理工学院(MIT)提出第一个公钥加密算法——RSA算法，之后ElGamal、ECC、双线性对等公钥密码相继被提出，密码学进入了新的发展时期。

21世纪初，我国研究并推出了系列商用密码算法，包括祖冲之序列密码算法、SM2公钥密码算法、SM3密码杂凑算法、SM4分组密码算法、SM9标识密码算法，其逐渐成为国际密码标准。

密码学的应用已经深入人们生活的各个方面，如数字证书、网上银行、身份证、社保卡和税务管理等，密码技术在其中都发挥了关键作用。近年来，其他相关学科的快速发展，促使密码学中出现了新的密码技术，如量子密码、混沌密码和DNA密码等。

量子密码学是现代密码学领域的一个很有前途的新方向，量子密码的安全性是基于量子力学的测不准性和不可复制性，其特点是对外界任何振动的可检测性和易于实现的无条件安全性。量子密码通信不仅是绝对安全、不可破译的，而且任何窃取量子的动作都会改变量子的状态，所以一旦存在窃听者，量子密码的使用者就会立刻获知。因此，量子密码可成为光通信网络中数据保护的强有力工具，而且能对付未来具有量子计算能力的攻击者。

混沌系统的两大特征是对初始条件敏感以及系统变化的不可预测性，这两个特性恰好满足密码学随机序列的要求。混沌在密码学中的研究可以分为两种形式：一种是序列密码，即利用混沌系统产生伪随机序列作为密钥序列，对明文进行加密；另一种是分组密码，即使用明文或密钥作为混沌系统的初始条件或结构参数，通过混沌映射的抚今追昔来产生密文。

DNA密码体制的特点是以DNA为信息载体，以现代生物技术为实现工具，挖掘DNA固有的高存储密度和高并行性等优点，实现加密、认证及签名等功能。

### 3.1.5 网络加密的实现方法

基于密码算法的数据加密技术是全网络上的通信安全所依赖的基本技术。目前，对网络数据加密主要有链路加密、节点对节点加密和端对端加密3种实现方式。

1. 链路加密

链路加密又称在线加密，它是对在两个网络节点间的某一条通信链路实施加密，是目前网络安全系统中主要采用的方式。链路加密能为网络传输的数据提供安全保证，所

有消息在被传输之前进行逐位加密，对接收到的消息进行解密，然后使用下一个链路的密钥对消息进行加密后再进行传输。在链路加密方式中，不仅对数据报文的正文加密，而且把路由信息、校验和等控制信息全部加密。所以，当数据报文传输到某一个中间节点时，必须先被解密以获得路由信息和检验和，进行路由选择、差错检测，再被加密，发送给下一个节点，直到数据报文到达目的节点为止。

如图 3-2 所示，在链路加密方式下，只对通信链路中的数据加密，而不对网络节点内的数据加密。因此，在中间节点上的数据报文是明文出现的，而且要求网络中的每一个中间节点都要配置安全单元（信息加密设备）。相邻两个节点的安全单元使用相同的密钥。这种使用不是很方便，因为需要网络设施的提供者配合修改每一个交换节点，这种方式在广域网上是不太现实的。在传统的加密算法中，用于解密消息的密钥与用于加密的密钥是相同的，必须秘密保存该密钥，并按一定规则进行变化。这样，密钥分配在链路加密系统中就成为一个问题，因为每一个节点必须存储与其相连接的所有链路的加密密钥，需要对密钥进行物理传送或者建立专用网络设施。而网络节点地理分布的广阔性使得这一过程变得复杂，同时增加了密钥连续分配时的费用。链路加密方式的优点是应用系统不受加密和解密的影响，容易被采用。

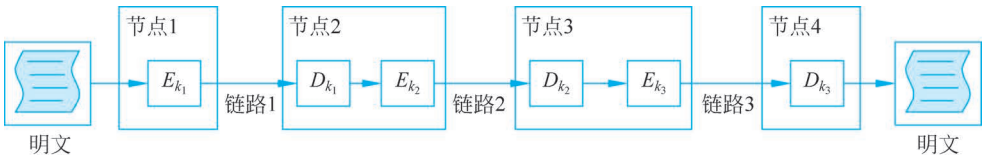


图 3-2　链路加密过程

2. 节点对节点加密

节点对节点加密是为了解决节点中的数据是明文的这一问题，在中间节点内装有用于加密和解密的保护装置，由这个装置来完成一个密钥向另一个密钥的交换。因而，除了在保护装置里，即使在节点内也不会出现明文。

尽管节点对节点加密能给网络数据提供较高的安全性，但它在操作方式上与链路加密类似：两者均在通信链路上为传输的消息提供安全性，都在中间节点先对消息进行解密再进行加密。因为要对所有传输的数据进行加密，所以加密过程对用户是透明的。然而，与链路加密不同，节点对节点加密不允许消息在网络节点以明文形式存在。它先把收到的消息进行解密，然后采用另一个不同的密钥进行加密，这一过程在节点上的一个安全模块中进行。节点对节点加密要求报头和路由信息以明文形式传输，以便中间节点能快速得到路由信息和校验和，加快消息的处理速度。但是，节点对节点加密与链路加密方式一样存在共同的弱点：需要公共网络提供者的配合来修改公共网络的交换节点以增加安全单元或保护装置。

3. 端对端加密

为了解决链路加密和相邻节点之间加密中存在的不足，人们提出了端对端加密方式。端对端加密又称为脱线加密或包加密，它允许数据在从源节点被加密后，到终点的

传输过程中始终以密文形式存在，只有消息到达目的节点后才被解密。因为消息在整个传输过程中均受到保护，所以即使有节点被损坏也不会泄露消息。因此，端对端加密方式可以实现按各通信对象的要求改变加密密钥以及按应用程序进行密钥管理等，而且采用这种方式可以解决文件加密问题。

链路加密方式是对整个链路通信采取保护措施，而端对端加密方式则是对整个网络系统采取保护措施。端对端加密系统更容易设计、实现和维护，且成本相对较低。端对端加密还避免了其他加密系统所固有的同步问题，因为每个报文段均是独立被加密的，所以一个报文段发生的传输错误不会影响后续的报文段。此外，端对端加密方便，不依赖底层网络基础设施，既可以在局域网内部实施，也可以在广域网上实施。端对端加密系统通常不允许对消息的目的地址进行加密，这是因为每一个消息所经过的节点都要用此地址来确定如何传输消息。因此，端对端加密方式是目前互联网应用的主流，应用层加密的实现多采用端对端加密方式。由于端对端加密方法不能掩盖被传输消息的源节点与目的节点，因此它对于防止攻击者分析通信业务是脆弱的。

## 3.2 替代密码

代换密码分为单字母代换密码和多字母代换密码，单字母代换密码又分为单表代换密码和多表代换密码。单表代换密码只使用一个密文字母表，并且用密文字母表中的一个字母来代替明文字母表中的一个字母。多表代换密码通过构造多个密文字母表，在密钥的控制下用相应密文字母表中的一个字母来代替明文字母表中的一个字母，一个明文字母有多种代替。多表代换密码是以两个或两个以上代换表依次对明文消息的字母进行代换的加密方法。

在单表代换密码中，只使用一个密文字母表，并且用密文字母表中的一个字母来代换明文字母表中的一个字母。设 $A$ 和 $B$ 分别为含 $n$ 个字母的明文字母表和密文字母表：

$$A=\{a_0,a_1,\cdots,a_{n-1}\}$$

$$B=\{b_0,b_1,\cdots,b_{n-1}\}$$

单表代换密码定义了一个由 $A$ 到 $B$ 的一一映射 $f$：$A\to B$：$f(a_i)=b_i$。设明文 $m=(m_0,m_1,\cdots,m_{n-1})$，则密文 $c=(f(m_0),f(m_1),\cdots,f(m_{n-1}))$。下面介绍 3 种具体的单表代替密码体制。

1. *加法密码*

加法密码的映射函数为

$$f(a_i)=b_i=a_j$$

$$j\equiv(i+k)\bmod n$$

式中：$a_i\in A$；$k$ 是满足 $0<k<n$ 的正整数。

消息空间$\mathcal{M}$、密文空间$\mathcal{C}$和密钥空间$\mathcal{K}$都为$\mathbb{Z}_q$。对任意消息 $m\in\mathcal{M}$和密钥 $k\in\mathcal{K}$，加法密码的加密算法可以表示为

$$c = E_k(m) \equiv (m + k) \bmod q$$

解密算法可以表示为

$$m = D_k(c) \equiv (c - k) \bmod q$$

2. 乘法密码

乘法密码的映射函数为

$$f(a_i) = b_i = a_j$$

$$j \equiv ik \mod n$$

式中：$k$ 与 $n$ 互素。

因为仅当$(k,n)=1$ 时，$k$ 才存在乘法逆元，才能正确解密。

消息空间$\mathcal{M}$和密文空间都为$\mathbb{Z}_q^*$，密钥空间$\mathcal{K}$为$\mathbb{Z}_q^*$。对任意消息 $m \in \mathcal{M}$和密钥 $k \in \mathcal{K}$，乘法密码的加密算法可以表示为

$$c = E_k(m) \equiv mk \mod q$$

解密算法可以表示为

$$m = D_k(c) \equiv ck^{-1} \bmod q$$

乘法密码(模 $q$)也是不安全的，密钥空间也很小，只有 $\phi(q)$种可能的情况。

3. 仿射密码

乘法密码和加法密码相结合便构成仿射密码，其映射函数为

$$f(a_i) = b_i = a_j$$

$$j \equiv (k_1 + ik_2) \bmod n$$

式中：$0 < k_1 < n$ 且$(k_2, n) = 1$。

消息空间$\mathcal{M}$和密文空间都为$\mathbb{Z}_q$，密钥空间$\mathcal{K}$为$\mathbb{Z}_q \times \mathbb{Z}_q^*$。对任意消息 $m \in \mathcal{M}$和密钥$(k_1, k_2) \in \mathcal{K}$，仿射密码的加密算法可以表示为

$$c = E_k(m) \equiv (k_1 + mk_2) \bmod q$$

解密算法可以表示为

$$m = D_k(c) \equiv (c - k_1)k_2^{-1} \bmod q$$

显然，加法密码和乘法密码都是仿射密码的特例。仿射密码的密钥空间也不大，只有 $q\phi(q)$种可能的情况。

多表代换密码首先将明文 $m$ 分为 $n$ 个字母构成的分组 $\boldsymbol{m}_1, \boldsymbol{m}_2, \cdots, \boldsymbol{m}_j$，加密算法可以表示为

$$\boldsymbol{c}_i = (\boldsymbol{A}\boldsymbol{m}_i + \boldsymbol{B}) \bmod q, \quad i = 1, 2, \cdots, j$$

式中：$(\boldsymbol{A}, \boldsymbol{B})$是密钥，$\boldsymbol{A}$ 是$\mathbb{Z}_q$ 上的 $n \times n$ 可逆矩阵，满足 $\gcd(|\boldsymbol{A}|, N) = 1$($|\boldsymbol{A}|$是行列式)，$\boldsymbol{B} = (b_1, b_2, \cdots, b_n) \in \mathbb{Z}_q^n$，$\boldsymbol{c}_i = (y_1, y_2, \cdots, y_n) \in \mathbb{Z}_q^n$；$\boldsymbol{m}_i = (x_1, x_2, \cdots, x_n) \in \mathbb{Z}_q^n$。

解密算法可以表示为

$$\boldsymbol{m}_i = \boldsymbol{A}^{-1}(\boldsymbol{c}_i - \boldsymbol{B}) \bmod q, \quad i = 1, 2, \cdots, j$$

### 3.2.1 Caesar 密码

Caesar 密码是古罗马恺撒大帝发明的一种单表代换密码，它是最早有记载的密码之

一。历史学家苏埃托尼乌斯在他的著作《罗马十二帝王传》中记载了恺撒曾用此方法对重要的军事信息进行加密。Caesar 密码是一种通过用其他字符替代明文中的每一个字符，完成将明文转换成密文过程的加密算法。在使用 Caesar 密码之前，先将字母按 0～25 编号，如表 3-1 所示。

表 3-1　Caesar 密码中的字母编码

| A | B | C | D | E | F | G | H | I | G | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
| **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** |
| 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 |

设明文字母表示为 $\alpha$，密文字母表示为 $\beta$，则加解密方法如下：

加密：$\beta=\alpha+n(\mathrm{mod}\ 26)$

解密：$\alpha=\beta-n(\mathrm{mod}\ 26)$

式中：$n$ 表示密钥。显然，$n$ 的有效取值为 0～25。如果 $n=0$，那么相当于没有对明文进行加密操作。假设明文 $m=$It is a secret，数学语言可以表示为

$$P=C=\{x \mid x\in[0,25],x\in\mathbb{Z}\}$$

$$n=k_e=k_d=3$$

$$E(k_e,p)=(p+3)\mathrm{mod}\ 26$$

$$D(k_d,c)=(c-3)\mathrm{mod}\ 26$$

那么密文 $c=$LWLVDVHFUHW。

Caesar 密码具有加解密公式简单、加解密容易理解的优点，但是因为所用语言已知，容易识别、需要测试的密钥只有 25 个(表 3-2)，容易被穷举法破译。

表 3-2　Caesar 密码明密文对照

| ***m*** | **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 |
| | +3 mod 26 | | | | | | | | | | | | | | | | | | | | | | | | | |
| ***c*** | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 | 25 | 0 | 1 | 2 |
| | **D** | **E** | **F** | **G** | **H** | **I** | **J** | **K** | **L** | **M** | **N** | **O** | **P** | **Q** | **R** | **S** | **T** | **U** | **V** | **W** | **X** | **Y** | **Z** | **A** | **B** | **C** |

### 3.2.2　Vigenère 密码

为了增加密码破译的难度，在单表代换密码的基础上扩展出多表代换密码，并将其称为 Vigenère 密码。Vigenère 密码引入了“密钥”的概念，即根据密钥来决定用哪一行的密表来进行替换，以此对抗字频统计。

词组中每一个字母都作为索引来确定采用某个代换表，加密时需要循环使用代换表完成明文字母到密文字母的代换，最后所得到的密文字母序列即为密文。Vigenère 密码的特点是将 26 个 Caesar 表合成一个 Vigenère 密码坐标图，形成了 26×26 的矩阵。矩

阵的第一行是按正常顺序排列的字母表，第二行是第一行左移循环 1 位得到的，以此类推，得到其余各行。然后在基本方阵的最上方附加一行，最左侧行加一列，分别依序写上 A 到 Z，共计 26 个字母。表的第一行与附加列上的字母 A 相对应，表的第二行与附加列上的字母 B 相对应，以此类推，最后一行与附加列上的字母 Z 相对应。如果把上面的附加行看成明文序列，则下面的 26 行就分别构成了左移 0 位，1 位，2 位，…，25 位的 26 个单表代换加同余密码的密文序列。同理，也可以把附加列看作明文序列，加密时按照密钥信息来决定采用相关的单表。Vigenère 密码坐标图如图 3-3 所示。

| | 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 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| A | 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 |
| B | 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 | A |
| C | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B |
| D | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C |
| E | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D |
| F | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E |
| G | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F |
| H | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G |
| I | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H |
| J | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I |
| K | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J |
| L | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K |
| M | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L |
| N | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M |
| O | O | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N |
| P | P | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O |
| Q | Q | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P |
| R | R | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q |
| S | S | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R |
| T | T | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S |
| U | U | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T |
| V | V | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U |
| W | W | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V |
| X | X | Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W |
| Y | Y | 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 |
| Z | 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 |

图 3-3　Vigenère 密码坐标图

设 $m$ 为某个固定的正整数，$P$、$C$ 和 $K$ 分别表示为明文空间、密文空间和密钥空间，且 $P=C=K=(Z_{26})^m$，对于一个密钥 $k=(k_1,k_2,\cdots,k_m)$，可以定义如下：

加密：$E_k(k_1,k_2,\cdots,k_m)=(x_1+k_1,x_2+k_2,\cdots,k_m+k_m)$

解密：$D_k(y_1,y_2,\cdots,y_m)=(y_1-k_1,y_2-k_2,\cdots,y_m-k_m)$

其中：$(x_1,x_2,\cdots,x_m)$为一个明文分组中的 $m$ 个字母，密钥空间大小为 $26^m$。

先从一位的密钥开始。此时 Vigenère 密码就变成 Caesar 密码，加密的方法是将原文字母顺序移位密钥字母在字母表中的个数。例如，使用密钥 b，加密单词 and，每一组

的两个字母就成为坐标。在 Vigenère 密码坐标图中分别查找横向和纵向。横向和纵向相交点就是加密后的字母。明文第一个字母是 a，密钥是 b，在表中左边查找 a 那一行和顶端 b 那一列，两者相交的交点字母就是密文，即 b，如图 3-3 所示。根据字母表的顺序，and 加密后就为 boe。对于多位密钥，如以明文 data security，密钥 best 为例，获得密文为 eelttiunsmlr。

Vigenère 密码算法具有相对复杂的密钥，相同的字母将被加密为不同的密文字母的优点。但是，如果密文足够长，其间会有大量重复的密文序列出现；或者通过计算重复密文序列间的公因子，分析者可能猜出密钥长度。

## 3.3 置换密码

### 3.3.1 置换密码概述

置换密码(Permutation Cipher)又称换位密码，在置换密码中，明文的字母相同，但出现的顺序被打乱，经过多步置换会进一步打乱字母顺序，如图 3-4 所示。由于密文字符与明文字符相同，密文中字母的出现频率与明文中字母的出现频率相同，密码分析者可以很容易地辨别。如果将置换密码与其他密码技术结合，则可以得出十分有效的密码编码方案。置换密码是一种通过改变明文中每一个字符的位置，将明文转换成密文的加密算法。置换密码加解密的过程方便、计算量小，速度较快；但是完全保留字符的统计信息，具有加密结果简单、容易被破译等特点。


(a) 加密过程　　(b) 解密过程

图 3-4　置换密码加密和解密

设 $n$ 为正整数，$M$、$C$ 和 $K$ 分别为明文空间、密文空间和密钥空间。明文、密文都是长度为 $n$ 的字符序列，分别记为 $X=(x_1,x_2,\cdots,x_n)\in M$，$Y=(y_1,y_2,\cdots,y_n)\in C$，$K$ 是定义在$\{1,2,\cdots,n\}$的所有置换组成的集合。对于任何一个密钥 $\sigma\in K$，即任何一个置换，定义置换密码为

$$\begin{cases} e_\sigma(x_1,x_2,\cdots x_n)=(x_{\sigma(1)},x_{\sigma(2)},\cdots x_{\sigma(n)}) \\ d_{\sigma^{-1}}(y_1,y_2,\cdots,y_n)=(y_{\sigma^{-1}(1)'},y_{\sigma^{-1}(2)'},\cdots,y_{\sigma^{-1}(n)}) \end{cases}$$

式中：$\sigma^{-1}$ 是 $\sigma$ 的逆置换；密钥空间 $K$ 的大小为 $n!$。

置换密码可以分为列置换密码和周期置换密码。列置换密码是指明文按照密钥的规程按列换位并且按列读出序列得到密文。周期置换密码是指将明文串 $P$ 按固定长度 $m$ 分组，然后对每组中的子串按 $1,2,\cdots,m$ 的某个置换重排位置从而得到密文 $C$。其中密钥 $k$ 既包含分组长度的信息，也包含明文变化信息。解密时按照密钥 $k$ 的长度 $m$ 分组后进行求逆重新排列，即可得到明文。

### 3.3.2 Rail Fence 密码

与 Caesar 密码和 Vigenère 密码相比，Rail Fence（栅栏加密）密码属于置换密码。Rail Fence 密码非常简单，把明文（去掉空格）分成 $n$ 组，每组 $m$ 个，然后按一定的排序方法来将这些字符重新组合得到了密文。例如，待加密的信息为 THE LONGEST DAY MUST HAVE AN END，将传递的信息中的字母交替排成上下两行：

T E O G S D Y U T A E N N

H L N E T A M S H V A E D

再将下面一行字母排在上面一行的后边，从而形成一段密文 TEOGSDYUTAENN HLNETAMSHVAED。

## 3.4 对称密码

### 3.4.1 对称密码概述

对称密码的基本特征是用于加密和解密的密钥相同，或者相对容易推导，因此也称为单密钥密码。典型对称密码有分组密码和流密码。分组密码和流密码的区别：其输出的每一位数字不是只与对应（时刻）的输入明文数字有关，还与长度为 $N$ 的一组明文数字有关。分组密码中二进制明文分组的长度称为该分组密码的分组规模。

分组密码原理如图 3-5 所示。

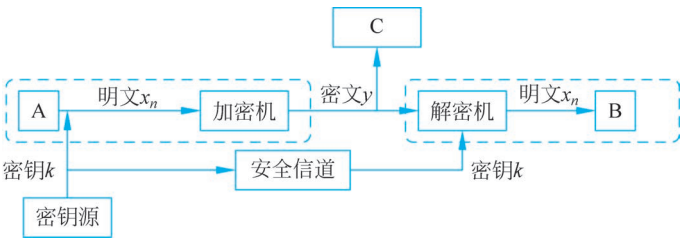


图 3-5 分组密码原理

分组长度 $n$ 通常为 64 位或 128 位；密钥 $k$ 长度为 64 位、128 位或 256 位；密文不随时间的发化而发化。扩散和混淆是香农提出的设计密码体制的两种基本方法，其目的是抵抗攻击者对密码体制的统计分析。扩散就是让明文以及密钥中的每一位能够影响密文中的许多位，或者说密文中的每一位受明文和密钥中的许多位的影响。这样可以隐蔽明文的统计特性，从而增加密码的安全性。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和密钥中所有位的影响。混淆就是将密文与明文、密钥之间的统计关系变得尽可能复杂，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。

对称密码加密、解密处理速度快，具有很高的数据吞吐率，硬件加密实现可达到几百兆字节每秒，软件也可以达到兆字节每秒的吞吐率。密钥相对较短。但是，密钥是保密通信安全的关键，发信方必须安全、妥善地把密钥护送到收信方，不能泄露其内

容。对称密钥的分发过程十分复杂，代价高。多人通信时密钥组合数量会出现爆炸性膨胀，使密钥分发更加复杂，$N$ 个人进行两两通信，共需要的密钥数为 $N(N-1)/2$。通信双方必须统一密钥才能发送保密的信息。对称密码算法还存在数字签名困难问题。

### 3.4.2 密码标准

密码标准体系框架从技术维、管理维和应用维对密码标准进行组织和刻画。技术维主要从标准所处技术层次的角度进行刻画，共有七大类，各类之间的依赖关系如图 3-6 所示。

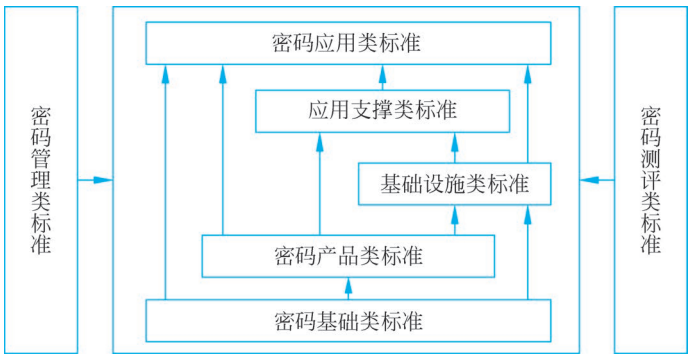


图 3-6 密码标准体系框架

1. 密码基础类标准

密码基础类标准对通用密码技术进行规范，它是体系框架内的基础性规范，主要包括密码术语与标识标准、密码算法标准、算法使用标准、密钥管理标准和密码协议标准等。

2. 基础设施类标准

基础设施类标准主要针对密码基础设施进行规范，包括证书认证系统密码协议、数字证书格式、证书认证系统密码及相关安全技术等。目前已颁布的密码标准涉及公钥基础设施及标识基础设施，未来可能还会出现其他密码基础设施类标准。

3. 密码产品类标准

密码产品类标准主要规范各类密码产品的接口、规格以及安全要求。对于各类密码产品给出设备接口、技术规范和产品规范；对于密码产品的安全性，则不区分产品功能的差异，而以统一的准则给出要求和设计指南；对于密码产品的配置管理，设备统一管理以 GM/T 0050—2016《密码设备管理 设备管理技术规范》为基础制定，针对具体设备也可能单独制定管理规范。

4. 应用支撑类标准

应用支撑类标准针对交互报文、交互流程、调用接口等方面进行规范，包括通用支撑类和典型支撑类两个层次。通用支撑类规范 GM/T 0019—2012《通用密码服务接口规

范》通过统一的接口向典型支撑标准和密码应用标准提供加解密、签名验签等通用密码功能，典型支撑类标准是基于密码技术实现的与应用无关的安全机制、安全协议和服务接口，如可信计算可信密码支撑平台接口、证书应用综合服务接口等。

5. 密码应用类标准

密码应用类标准针对使用密码技术实现某种安全功能的应用系统提出的要求以及规范，包括应用要求、应用指南、应用规范和密码服务等子类。应用要求旨在规范社会各行业信息系统对密码技术的合规使用。应用指南用于指导社会各行业建设符合密码应用要求标准的信息系统。应用规范定义了具体的密码应用规范，应用规范类标准也包括其他行业标准机构制定的与行业密切相关的标准，如 JR/T 0025—2018《中国金融集成电路(IC)卡规范》中，对金融 IC 卡业务过程中的密码技术应用做了详细规范。密码服务类则用以规范面向公众或特定领域提供的各类密码服务，目前该类标准暂时空缺。

6. 密码测评类标准

密码测评类标准针对标准体系所确定的基础、产品和应用等类型的标准出台对应检测标准，如针对随机数、安全协议、密码产品功能和安全性等方面的检测规范。其中，对于密码产品的功能检测，分别针对不同的密码产品定义检测规范；对于密码产品的安全性检测，基于统一的准则执行。

7. 密码管理类标准

密码管理类标准包括国家密码管理部门在密码标准、密码算法、密码产业、密码服务、密码应用、密码监察、密码测评等方面的管理规程和实施指南。

### 3.4.3 对称密码标准

1. GB/T 33133—2016《信息安全技术 祖冲之序列密码算法》

该标准描述了祖冲之密码(ZUC)算法，以及使用祖冲之算法实现机密性和完整性保护的方法。该标准适用于使用祖冲之序列密码算法产品的研制、生产和检测。

GM/T 0001—2012《祖冲之序列密码算法》分为 3 个部分：GM/T 0001.1《祖冲之序列密码算法 第 1 部分：算法描述》，描述了祖冲之密码算法的基本原理，该部分已发布为国家标准 GB/T 33133.1—2016；GM/T 0001.2《祖冲之序列密码算法 第 2 部分：基于祖冲之算法的机密性算法》，描述了使用祖冲之密码算法加密明文数据流的方法；GM/T 0001.3《祖冲之序列密码算法 第 3 部分：基于祖冲之算法的完整性算法》，描述了使用祖冲之密码算法针对明文生成 32 位 MAC 值的方法。

2. GB/T 32907—2016《信息安全技术 SM4 分组密码算法》

该标准描述了 SM4 分组密码算法，是一种密钥长度为 128 位，分组长度也是 128 位的密码算法。该标准适用于使用分组密码算法进行数据保护的场合，实现对明文数据的加密保护，以及以 CBC-MAC 等方式实现的完整性保护。

该标准主要内容包括：SM4 算法的结构，SM4 的 128 位密钥和 32 个 32 位轮密钥，

以及算法中用到的 FK 和 CK 两个算法参量，每轮运算的轮函数 $F$ 和算法（包括加密算法、解密算法以及密钥扩展算法）的实现。

3. GB/T 17964《信息安全技术 分组密码算法的工作模式》

该标准描述了分组密码算法的 7 种工作模式，以便规范分组密码的使用。该标准描述的工作模式仅适用于保护数据的机密性，不适用于保护数据的完整性，可与具有鉴别功能的 GB/T 15852.1—2020《信息技术 安全技术 消息鉴别码 第 1 部分：采用分组密码的机制》、GB/T 36624—2018《信息技术 安全技术 可鉴别的加密机制》等标准搭配使用。

GB/T 17964—2021 标准分别描述了电子编码本（ECB）模式、密码分组链接（CBC）模式、密码反馈（CFB）模式、输出反馈（OFB）模式、计数器（CTR）模式、分组链接（BC）模式和带非线性函数的输出反馈（OFBNLF）模式，包括变量定义、加密方式、解密方式以及必要的图示等。在 3.6 节着重讲解了前 5 种分组密码算法的工作模式。

## 3.5 分组密码

分组密码是将明文消息编码后的序列划分成固定大小的组，每组明文分别在密钥的控制下变成等长的密文序列。首先对任意长度明文进行填充，使得填充后的明文长度是加密算法要求的长度的整数倍；然后将填充后的明文分割成长度等于加密算法规定长度的数据段，对每一段数据段独立进行加密运算，产生和数据段长度相同的密文，密文序列和明文分段后产生的数据段序列一一对应。

输入是 $n$ 位明文 $m$ 和 $b$ 位密钥 $k$，输出是 $n$ 位密文 $c$，表示成 $Ek(m)=c$。假如明文和密文的分组长度都为 $n$ 位，明文和密文取值都在 GF(2)中，那么明文和密文的每个分组都有 $2^n$ 个可能的取值。为使加密运算可逆（解密可行），明文的每个分组都应产生唯一一个密文分组，这样的变换是可逆的，明文分组到密文分组的可逆变换称为代换。不同可逆变换的个数为 $2^n!$，但考虑密钥管理问题和实现效率，现实中的分组密码的密钥长度 $k$ 往往与分组长度 $n$ 差不多，共有 $2^k$ 个代换，而不是理想分组的 $2^k!$ 个代换。在使用时还需考虑分组长度。如果分组长度太小，那么等价于古典的代换密码。如果分组长度足够大，则可逆代换结构是不实际的。

### 3.5.1 数据加密标准

数据加密标准（Data Encryption Standard，DES）是在美国国家安全局（NSA）资助下由 IBM 公司开发的一种对称密码算法，其初衷是为政府非机密的敏感信息提供较强的加密保护。它是美国政府担保的第一种加密算法，并在 1977 年被正式作为美国联邦信息处理标准。

DES 的密钥长度和数据段长度均为 64 位，加密运算前，将数据分为长 64 位的数据段，通过一个初始置换，将明文分成左半部分和右半部分，长度均为 32 位。然后进行 16 轮完全相同的运算。在运算过程中数据与密钥结合，经过 16 轮后，左、右半部分合在一起，经过一个末置换（初始置换的逆置换），产生长 64 的密文。密钥长 64 位，密钥只有 56 位参与 DES 运算，第 8、16、24、32、40、48、56、64 位是校验位，使得每个密钥都有奇数个 1。

DES 算法包括 IP 初始置换、密钥置换、E 扩展置换、S 盒代替、P 盒置换和 $IP^{-1}$ 末置换等步骤(图 3-7)。

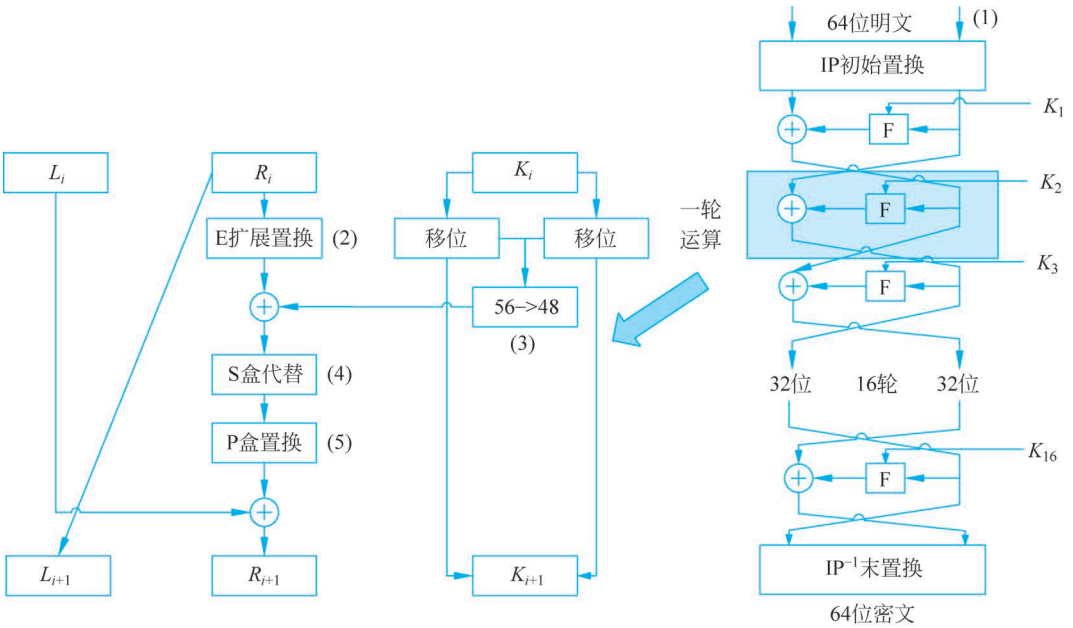


图 3-7　DES 算法

1. IP 初始置换

将输入的 64 位数据块按位重新组合,并把输出分为 $L_0$、$R_0$ 两部分,每部分各占 32 位。IP 初始置换规则如表 3-3 所示。

表 3-3　IP 初始置换规则

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

表 3-3 中的数值代表新数据中此位置的数据在原数据中的位置,即原数据块的第 58 位放到新数据的第 1 位,第 50 位放到第 2 位……第 7 位放到第 64 位。

2. E 扩展置换

E 扩展置换目标是 IP 置换后获得的右半部分 $R_0$,将 32 位输入扩展为 48 位(分为 4 位×8 组)输出。E 扩展置换之后,右半部分数据 $R_0$ 变为 48 位,与密钥置换得到的轮密

钥进行异或。

E扩展置换能够生成与密钥相同长度的数据，以进行异或运算或者提供更长的结果，便于后续的替代运算中可以进行压缩。E扩展置换规则如表3-4所示，其中两列阴影数据是扩展的数据。可以看出，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位的为1位，靠近1位的为32位。表中第二行的4取自上组中的末位，9取自下组中的首位。

表3-4 E扩展置换规则

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

3. 密钥置换

不考虑每字节的第8位，DES的密钥由64位减至56位，每字节的第8位作为奇偶校验位。产生的56位密钥由表3-5生成。注意第8、16、24、32、40、48、56、64位是校验位。

表3-5 产生的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 |

在DES的每轮中，从56位密钥产生出不同的48位子密钥，确定子密钥方法如下。

(1) 将56位的密钥分成两部分，每部分28位。

(2) 根据轮数，将这两部分分别循环左移1位或2位。每轮移动的位数如表3-6所示。

表3-6 每轮移动的位数

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

移动后，在56位中选出48位，这样既置换了每位顺序，又选择了子密钥，因此称为压缩置换。压缩置换规则如表3-7所示，注意表中没有第9、18、22、25、35、38、43和54位。置换方法类同。

表 3-7　压缩置换规则

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

4. S 盒代替

压缩后的密钥与扩展分组异或以后得到 48 位的数据，将数据送入 S 盒，进行代替运算，如图 3-8 所示。代替由 8 个不同的 S 盒完成，每个 S 盒有 6 位输入和 4 位输出。48 位输入分为 8 个 6 位的分组，一个分组对应一个 S 盒，对应的 S 盒对各组进行代替操作。

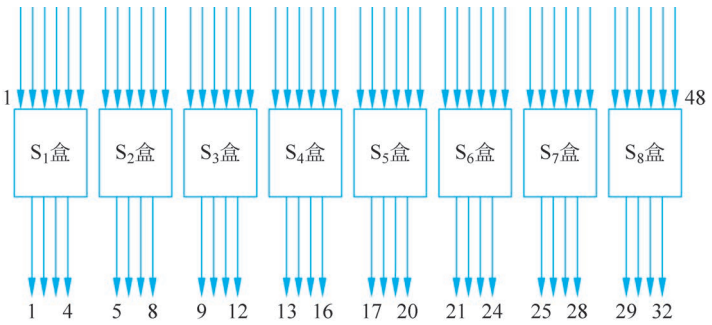


图 3-8　S 盒

$S_1$～$S_8$ 盒分别如表 3-8～表 3-15 所示。

表 3-8　$S_1$ 盒

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

表 3-9　$S_2$ 盒

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

表 3-10　$S_3$ 盒

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

表 3-11 $S_4$ 盒

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | 19 |
| 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 |

表 3-12 $S_5$ 盒

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

表 3-13 $S_6$ 盒

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

表 3-14 $S_7$ 盒

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

表 3-15 $S_8$ 盒

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

5. P 盒置换

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。P 盒置换规则如表 3-16 所示。表 3-16 中的数值代表原数据中此位置的数据在新数据中的位置，即原数据块的第 16 位放到新数据的第 1 位，第 7 位放到第 2 位……第 25 位放到第 32 位。P 盒置换的结果与最初的 64 位分组左半部分 $L_0$ 异或，然后左、右半部分交换，开始另一轮。

表 3-16 P 盒置换规则

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

6. $IP^{-1}$ 末置换

$IP^{-1}$ 末置换是初始置换的逆过程，DES 最后一轮后，左、右两半部分并未进行交换，而是两部分合并形成一个分组作为末置换的输入。$IP^{-1}$ 末置换规则如表 3-17。

表 3-17 $IP^{-1}$ 末置换规则

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

可以看出，DES 算法具有如下特点。

（1）分组加密算法。以 64 位为分组，64 位一组的明文从算法一端输入，64 位密文从另一端输出。

（2）对称算法。加密和解密用同一密钥。

（3）有效密钥长度为 56 位。密钥通常表示为 64 位数，但每个第 8 位用作奇偶校验，可以忽略。

（4）代替和置换。DES 算法是两种加密技术的组合，即先代替后置换。

（5）易于实现。DES 算法只是使用了标准的算术和逻辑运算，其作用的数量最多只有 64 位，因此用 20 世纪 70 年代末期的硬件技术很容易实现。

由于 DES 算法密钥长度偏短等缺陷，不断受到差分密码分析和线性密码分析等各种攻击威胁，使其安全性受到严重的挑战。DES 算法具有如下具体安全隐患。

（1）密钥太短。DES 算法的初始密钥实际长度只有 56 位，密钥长度不足以抵抗穷举搜索攻击，穷举搜索攻击破解密钥，不太可能提供足够的安全性。

（2）DES 算法的半公开性。DES 算法中的 8 个 S 盒替换表的设计标准自 DES 算法公布以来仍未公开，替换表中的数据是否存在某种依存关系用户无法确认。

（3）DES 算法迭代次数偏少。DES 算法的 16 轮迭代次数被认为偏少，在以后的 DES 改进算法中，都不同程度地进行了提高。

## 3.5.2 3DES 算法

DES 算法的最大缺陷是使用了短密钥。为了克服这个缺陷，Tuchman 提出了 3DES 算法，使用了 168 位的长密钥。3DES 使用 3 倍 DES 的密钥长度的密钥，执行 3 次 DES 算法。由于 DES 密钥的长度实质上是 56 位，因此 3DES 的密钥长度就是 $56\times3=168$ 位。3DES 并不是进行 3 次 DES 加密（加密→加密→加密），而是加密→解密→加密的过程，如图 3-9 所示。

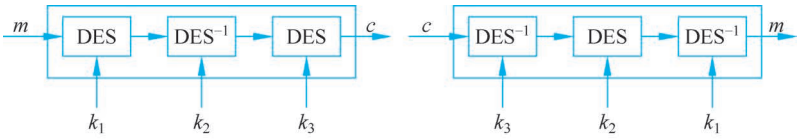


图 3-9　3DES 密钥

如图 3-10 所示，如果密钥 1 和密钥 3 使用相同的密钥，而密钥 2 使用不同的密钥（也就是只使用两个 DES 密钥），这种 3DES 也称为 DES-EDE2。EDE2 表示加密→解密→加密过程。双密钥 3DES 加密，密钥长度为 112 位。

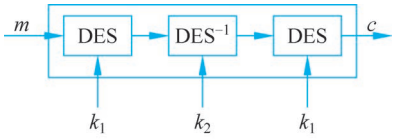


图 3-10　DES-EDE2 密钥

3DES 的密钥长度为 112 位或者 168 位，足够长。3DES 底层算法与 DES 相同，因此承受密码分析时间远远长于其他加密算法，对密码分析攻击有很强免疫力。3DES 的设计主要针对硬件实现，但相对在许多领域用软件方法来实现，则效率相对较低，实现速度更慢。虽然密钥增加了，但分组长度仍为 64 位，似乎应该更长。因此，3DES 不能成为长期使用的加密算法标准。

### 3.5.3　高级加密标准

高级加密标准（Advanced Encryption Standard，AES）在密码学中又称 Rijndael 加密法，是美国联邦政府采用的一种区块加密标准。这个标准用于替代原先的 DES，已经被多方分析且在全世界广泛使用。美国国家标准与技术研究院（NIST）于 2001 年 11 月 26 日发布高级加密标准，并于 2002 年 5 月 26 日成为有效的标准。AES 密钥长度可以是 128 位、192 位或者 256 位，数据段长度固定为 128 位。加密运算前，将数据分为 128 位长度的数据段，然后对每一段数据段进行加密运算，产生 128 位长度的密文。

AES 中的许多运算是按字节定义，或按 4 字节的字定义的。将字节看作有限域的一个元素，一个 4 字节的字看作 GF($2^8$)中并且次数小于 4 的多项式。有限域的元素在本算法中采用传统的多项式表达式，GF($2^8$)中的所有元素的系数为 GF(2)中且次数小于 8 的多项式。将 $b_7b_6b_5b_4b_3b_2b_1b_0$ 构成的一个字节看成多项式：

$$b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0(b_i \in \mathrm{GF}(2),0<i<7)$$

如十六进制数 57 对应的二进制数为 01010111，看作一字节，对应的多项式为 $xx^6+x^4+x^2+x+1$。采用的有加法运算、乘法运算和 $x$ 乘运算。

（1）加法运算：有限域 GF($2^8$)中的两个元素相加，结果是一个次数不超过 7 的多项式，其系数等于两个元素对应系数的模 2 加（比特异或）。有限域 GF($2^8$)中的两个元素加法与两字节的按位模 2 加是一致的。

例如，“57”和“83”的和为

$57 \oplus 83 = D_4$

$57 \rightarrow 01010111 \rightarrow x^6 + x^4 + x^2 + x + 1$

$83 \rightarrow 10000011 \rightarrow x^7 + x + 1$

$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2 \rightarrow 11010100 \rightarrow D_4$

显然，该加法与简单的以字节为组织的比特异或是一致的。

(2) 乘法运算：要计算有限域 GF($2^8$)上的乘法，必须先确定 GF(2)上的八次不可约多项式。GF($2^8$)上两个元素的乘积就是这两个多项式模乘(八次不可约多项式为模)。若一个多项式除了 1 和自身没有其他因子，则就是不可约的。对于 AES，这个八次不可约多项式确定为 $m(x) = x^8 + x^4 + x^2 + x + 1$，十六进制表示为 011$b$，二进制表示为 0000000100011011。

例如，“57”和“83”的乘为

$57 \cdot 83 = C_1$

$(x^6 + x^4 + x^2 + x + 1) \cdot (x^7 + x + 1)$

$= x^{13} + x^{11} + x^9 + x^8 + x^7 + x^7 + x^5 + x^3 + x^2 + x + x^6 + x^4 + x^2 + x + 1$

$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

而

$$(x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1) \bmod m(x)$$

$$= (x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1)$$

$$\bmod (x^8 + x^4 + x^3 + x + 1)$$；% 计算时按降幂排

$$= x^7 + x^6 + 1$$

所以

$(x^6 + x^4 + x^2 + x + 1) \cdot (x^7 + x + 1) = x^7 + x^6 + 1$；%多项式表示

01010111 · 10000011 = 11000001；%二进制表示

7 · 83 = C1；%16 进制表示

(3) $x$ 乘运算：用 $x$ 乘以一个多项式，简称 $x$ 乘。

$$(b_7 x^7 + b_6 x^6 + b_5 x^5 + b_4 x^4 + b_3 x^3 + b_2 x^2 + b_1 x + b_0) \otimes x$$

$$= b_7 x^8 + b_6 x^7 + + b_4 x^5 + b_3 x^4 + b_2 x^3 + b_1 x^2 + b_0 x$$

将上面的结果模 $m(x)$求余得到 $xb(x)$。如果 $b_7 = 0$，则结果就是 $x \cdot b(x)$；如果 $b_7 = 1$，则乘积结果先减去 $m(x)$，结果也为 $xb(x)$。$x$(十六进制数表示为 02)乘可以用字节内左移一位和紧接着一个 1$b$ 的按位模 2 加来实现，该运算即为 xtime 运算。

对于系数在 GF($2^8$)上的多项式，其系数可以定义为 GF($2^8$)中的元素，通过 4 字节构成的字可以表示为系数在 GF($2^8$)上的次数小于 4 的多项式，多项式的加法就是对应系数相加。GF($2^8$)中的加法为按模 2 加，因此两字节的加法就是按模 2 加。乘法比较复杂，规定多项式的乘法运算必须要取模 $m(x) = x^4 + 1$，这样使次数小于 4 的多项式的乘

积仍然是一个次数小于 4 的多项式，将多项式的模乘运算记为“⊗”。固定多项式 $a(x)$ 与多项式 $b(x)$ 做“⊗”运算可以写成矩阵乘法：

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

其中，矩阵是一个循环矩阵。$M(x)$不是 GF($2^8$)中的不可约多项式，因此被一个固定多项式相乘不一定是可逆。AES 中选择了一个有逆元的固定多项式，即

$$a(x) = \{03\} - x^3 + \{01\} \cdot x^2 + \{01\}x + \{02\}$$

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2 + \{09\}x + \{0e\}$$

$$a(x) \otimes a^{-1}(x) = a^{-1}(x) \otimes a(x) = \{01\}$$

假设 $c(x) = x \otimes b(x)$定义为 $x$ 与 $b(x)$的模 $x^4+1$ 乘法，即

$$c(x) = x \otimes b(x) = b_2x^3 + b_1x^2 + b_0x + b_3$$

则用矩阵表示为

$$\begin{bmatrix} c_0 \\ c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0 & 00 & 00 & 01 \\ 01 & 00 & 00 & 00 \\ 00 & 01 & 00 & 00 \\ 00 & 00 & 01 & 00 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

因此，系数在 GF($2^8$)上的多项式 $a_3x^3+a_2x^2+a_1x+a_0$ 是模 $x^4+1$ 可逆的，当且仅当矩阵

$$\begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix}$$

在 GF($2^8$)上可逆。

### 3.5.4 国际数据加密算法

国际数据加密算法(International Data Encryption Algorithm，IDEA)是最强大的加密算法之一。尽管 IDEA 很强大，但不像 DES 那么普及，原因有两个：一是 IDEA 受专利的保护，而 DES 不受专利的保护，因此 IDEA 要先获得许可证之后才能在商业应用程序中使用；二是 DES 比 IDEA 具有更长的历史和跟踪记录。

IDEA 是块加密，与 DES 一样，IDEA 也处理 64 位明文块，但是其密钥更长，共 128 位。IDEA 和 DES 一样是可逆的，即可以用相同的算法加密和解密。IDEA 也用扩展与混淆进行加密。图 3-11 显示了 IDEA 流程。

64 位输入明文块分为 4 个部分(各 16 块)$P_1 \sim P_4$。$P_1 \sim P_4$ 是算法的第 1 轮输入，

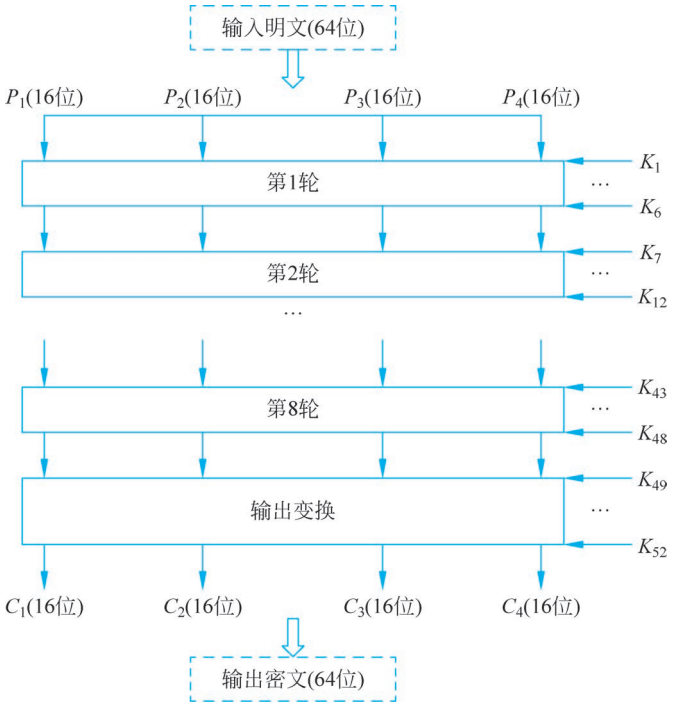


图 3-11　IDEA 流程

共 8 轮。密钥为 128 位，每轮从原先的密钥产生 6 个子密钥，各为 16 位。这 6 个子密钥作用于 4 个输入块 $P_1 \sim P_4$。第 1 轮有 6 个密钥 $K_1 \sim K_6$，第 2 轮有 6 个密钥 $K_7 \sim K_{12}$……第 8 轮有 6 个密钥 $K_{43} \sim K_{48}$。最后一步是输出变换，只用 4 个子密钥 $K_{49} \sim K_{52}$。产生的最后输出是输出变换的输出，为 4 个密文块 $C_1 \sim C_4$（各为 16 位），从而构成 64 位密文块。

DEA 每轮需要 6 个子密钥（因此 8 轮共需要 48 个子密钥），最后输出变换使用 4 个子密钥（共需要 52 个子密钥）。从 128 位的输入密钥得到 52 位子密钥，其前两轮的做法如下（根据前两轮的做法，可以得到后面各轮的子密钥表）。

第 1 轮：原始密钥为 128 位，可以产生第 1 轮的 6 个子密钥 $K_1 \sim K_6$。由于 $K_1 \sim K_6$ 各为 16 位，因此用到 128 位中的前 96 位（6 个子密钥，各为 16 位）。这样，第 1 轮结束时，第 97～128 位密钥还没有使用，如图 3-12 所示。

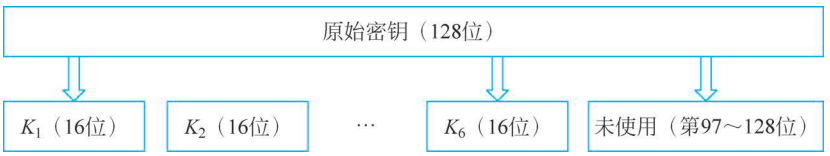


图 3-12　IDEA 密钥第 1 轮

第 2 轮：首先使用第 1 轮没有使用过的 32 位（第 97～128 位）密钥，共需要 96 位密钥，因此 IDEA 采用了密钥移位技术，在这个阶段原始密钥循环左移 25 位，即原始密钥

的第 26 位移到第 1 位(称为移位后的第 1 位),原始密钥的第 25 位移到最后一位(称为移位后的第 128 位)。其整个过程如图 3-13 所示。

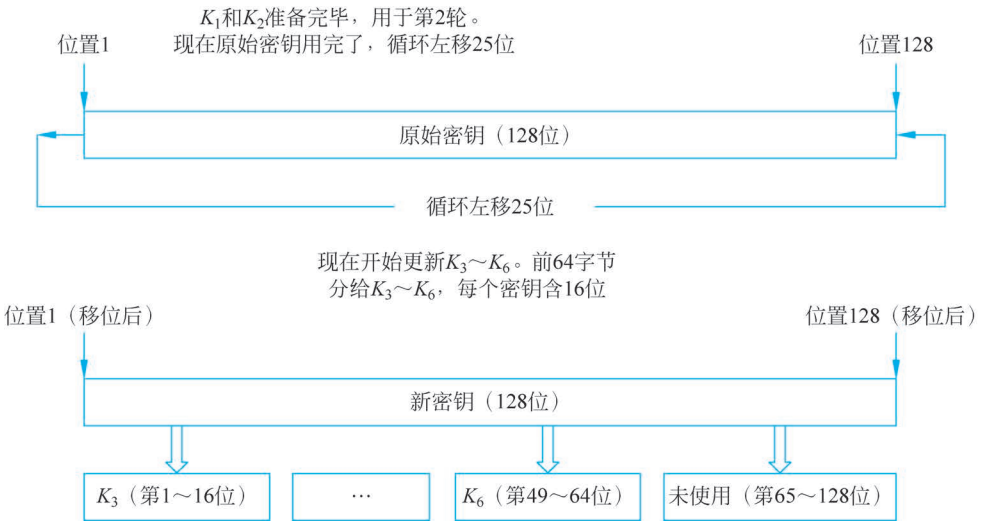


图 3-13　IDEA 密钥第 2 轮

可以看到,第 2 轮使用了第 1 轮的第 97～128 位,以及经过 25 位移位后的第 1～64 位。然后,第 3 轮其余的部分,即第 65～128 位(总共 64 位)。再次进行 25 位的移位,移位后,在第 3 轮使用第 1～32 位,以此类推。

解密过程与加密过程完全相同,只是子密钥的生成与模式不同。解密子密钥实际上是加密子密钥的逆。

### 3.5.5　SM4 算法

SM4 算法是一个迭代分组密码算法,由加解密算法和密钥扩展算法组成。SM4 算法采用非平衡 Feistel 结构,分组长度为 128 位,密钥长度为 128 位。加密算法与密钥扩展算法均采用 32 轮非线性迭代结构。加密运算和解密运算的算法结构相同,解密运算的轮密钥的使用顺序与加密运算相反。

SM4 算法的加密密钥长度为 128 位,表示为 $MK=(MK_0,MK_1,MK_2,MK_3)$,其中 $MK_i(i=0,1,2,3)$为 32 位。轮密钥表示为$(rk_0,rk_1,\cdots,rk_{31})$,其中 $rk_i(i=0,1,\cdots,31)$为 32 位。轮密钥由加密密钥生成。$FK=(FK_0,FK_1,FK_2,FK_3)$为系统参数,$CK=(CK_0,CK_1,\cdots,CK_{31})$为固定参数,用于密钥扩展算法,其中 $FK_i(i=0,1,\cdots,3)$,$CK_i(i=0,1,\cdots,31)$均为 32 位。

SM4 算法由 32 次迭代运算和 1 次反序变换 $R$ 组成。设明文输入为$(X_0,X_1,X_2,X_3)\in(Z_2^{32})^4$,密文输出为$(Y_0,Y_1,Y_2,Y_3)\in(Z_2^{32})^4$,轮密钥为 $rk_i\in Z_2^{32}(i=0,1,\cdots,31)$。加密运算过程如下。

首先执行 32 次迭代运算:

$$X_{i+4}=F(X_i,X_{i+1},X_{i+2},X_{i+3},rk_i)=$$

$$X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus \mathrm{rk}_i), \quad i=0,1,\cdots,31$$

对最后一轮数据进行反序变换并得到密文输出：

$$(Y_0,Y_1,Y_2,Y_3)=R(X_{32},X_{33},X_{34},X_{35})=(X_{35},X_{34},X_{33},X_{32})$$

其中，$T: Z_2^{32} \to Z_2^{32}$ 一个可逆变换，由非线性变换 $\tau$ 和线性变换 $L$ 复合而成，即 $T(\cdot)=L(\tau(\cdot))$。

非线性变换 $\tau$ 由 4 个并行的 $S$ 盒构成。设输入为 $A=(a_0,a_1,a_2,a_3)\in(Z_2^8)^4$，非线性变换 $\tau$ 的输出为 $B=(b_0,b_1,b_2,b_3)\in(Z_2^8)^4$，即

$$(b_0,b_1,b_2,b_3)=\tau(A)=(\mathrm{Sbox}(a_0),\mathrm{Sbox}(a_1),\mathrm{Sbox}(a_2),\mathrm{Sbox}(a_3))$$

$\mathrm{SM}_4$ 的 S 盒数据如表 3-18 所示。

表 3-18　SM4 算法的 S 盒数据

| | 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 | 99 |
| 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 |

设 S 盒的输入为 EF，则经 S 盒运算的输出结果为表中第 E 行、第 F 列的值，即 $\mathrm{Sbox}(\mathrm{EF})=0\times84$。$L$ 是线性变换，非线性变换 $\tau$ 的输出是线性变换 $L$ 的输入。设输入为 $B\in Z_2^{32}$，输出为 $C\in Z_2^{32}$，则有

$$C=L(B)=B\oplus(B<<<2)\oplus(B<<<10)\oplus(B<<<18)\oplus(B<<<24)$$

SM4 加密算法的运算如图 3-14 所示。

SM4 的解密变换与加密变换结构相同，不同的仅是轮密钥的使用顺序。解密时，使用轮密钥序 $(\mathrm{rk}_{31},\mathrm{rk}_{30},\cdots,\mathrm{rk}_0)$。轮密钥由加密密钥通过密钥扩展算法生成。设加密密钥为

$$\mathrm{MK}=(\mathrm{MK}_0,\mathrm{MK}_1,\mathrm{MK}_2,\mathrm{MK}_3)\in(Z_2^{32})^4$$

轮密钥生成方法为

$$\mathrm{rk}_i=K_{i+4}=K_i\oplus T'(K_{i+1}\oplus K_{i+2}\oplus K_{i+3}\oplus \mathrm{CK}_i) \quad (i=0,1,\cdots,31)$$

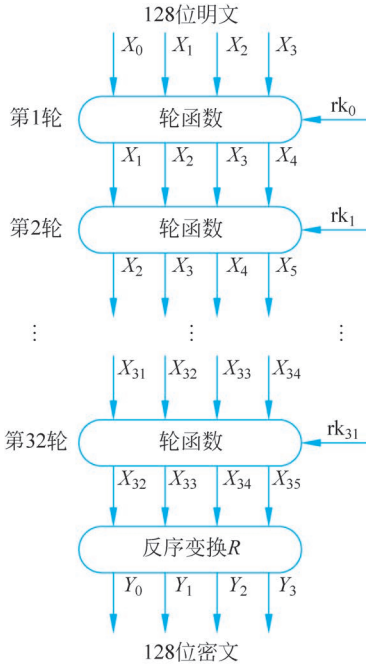


图 3-14　SM4 加密算法流程

其中

$$K_0 = \mathrm{MK}_0 \oplus \mathrm{FK}_0$$
$$K_1 = \mathrm{MK}_1 \oplus \mathrm{FK}_1$$
$$K_2 = \mathrm{MK}_2 \oplus \mathrm{FK}_2$$
$$K_3 = \mathrm{MK}_3 \oplus \mathrm{FK}_3$$

$T'$是将合成置换 $T$ 的线性变换 $L$ 替换为 $L'$：

$$\mathrm{FK}_1 = (56\mathrm{AA}3350)$$
$$\mathrm{FK}_2 = (677\mathrm{D}9197)$$
$$\mathrm{FK}_3 = (\mathrm{B}27022\mathrm{DC})$$

## 3.6 分组密码工作模式

分组密码是将消息作为数据分组来加密或解密的，而实际应用中大多数消息的长度是不定的，数据格式也不同。当消息长度大于分组长度时，需要分成几个分组分别进行处理。为了能灵活地运用基本的分组密码算法，人们设计了分组密码的工作模式，也称为分组密码算法的运行模式。

工作模式能为密文组提供一些其他的性质，如隐藏明文的统计特性、数据格式、控制错误传播等，以提高整体安全性，减少删除、重放、插入和伪造等攻击的机会。常用的工作模式有电子编码本模式、密码分组链接模式、密码反馈模式、输出反馈模式和计数器模式。

### 3.6.1 电子编码本模式

电子编码本模式是最简单的一种工作模式，一次对一个长 64 位的明文分组加密，而且每次的加密密钥都相同。当密钥取定时，对明文的每个分组都有唯一的密文与之对应。对任意一个可能的明文分组，电子编码本中都有一项对应于它的密文。ECB 模式如图 3-15 所示。

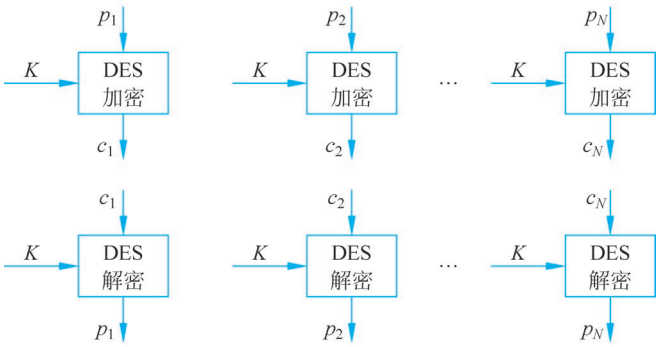


图 3-15 ECB 模式

ECB 模式用于短数据（如加密密钥）时非常理想，因此如果需要安全地传递 DES 密钥，ECB 模式是最合适的。若消息长于 64 位，则将其分为长为 64 位的分组；若最后一个分组不足 64 位，则需要填充。ECB 模式的最大缺陷是，若同一明文分组在消息中重复出现，则产生的密文分组也相同。因此，ECB 模式用于长消息时不够安全，若消息有固定结构，则密码分析者有可能找出这种关系。

### 3.6.2 密码分组链接模式

为了解决 ECB 模式的安全缺陷，可以让重复的明文分组产生不同的密文分组，密码分组链接模式就可以满足这一要求。CBC 模式一次对一个明文分组加密，每次加密使用同密钥，加密算法的输入是当前明文分组和前一次密文分组的异或，因此加密算法的输入不会显示与这次的明文分组之间的固定关系，且重复的明文分组不会在密文分组中暴露这种重复关系。每个密文分组被解密后再与前一个密文分组异或。CBC 模式如图 3-16 所示。

在产生第 1 个密文分组时，需要有一个初始化向量（Initialization Vector，IV）与第 1 个明文分组异或。解密时，IV 和解密算法对第 1 个密文分组的输出进行异或以恢复第 1 个明文分组。IV 对于收发双方都应是已知的，为使安全性提高，IV 应像密钥一样被保护，可使用 ECB 模式发送初始变量 IV。由于 CBC 模式的链接机制，CBC 模式非常适合加密长于 64 位的消息。CBC 模式除能够获得保密性，还能用于认证。

### 3.6.3 密码反馈模式

DES 是分组长度为 64 位的分组密码，但利用密码反馈模式或输出反馈模式可将

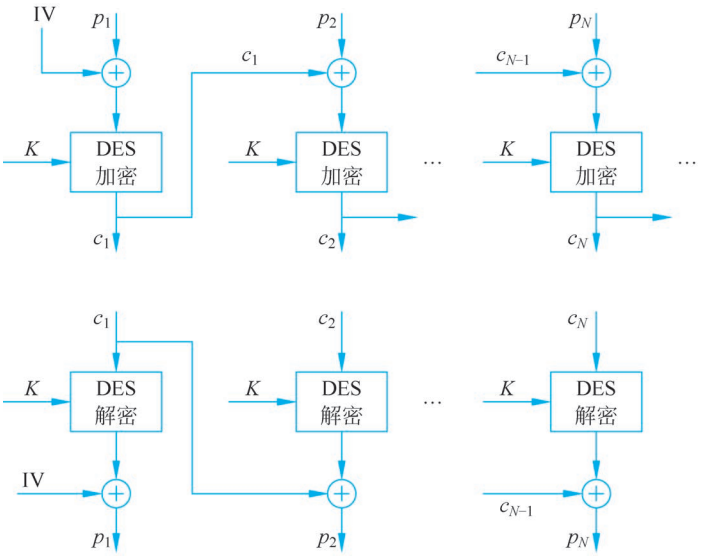


图 3-16　CBC 模式

DES 转换为流密码。流密码不需要对消息进行填充，而且运行是实时的，因此如果传送字母流，可使用流密码对每个字母直接加密并传送。流密码具有密文和明文一样的性质，因此如果需要发送的每个字符长度为 8 位，就应使用 8 位密钥来加密每个字符。若密钥长度超过 8 位，则将造成浪费。设传送的每个单元（如一个字符）长是 $d$ 位，通常取 $d=8$，与 CBC 模式一样，明文单元被链接在一起，使得密文是前面所有明文的函数。CFB 模式如图 3-17 所示。

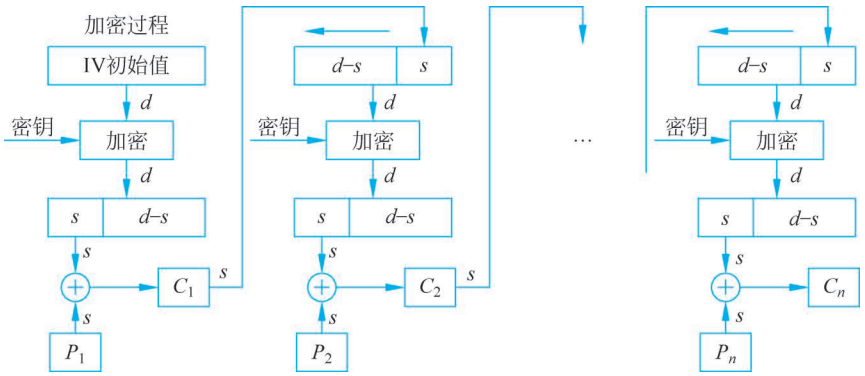


图 3-17　CFB 模式

加密时，加密算法的输入是 64 位移位寄存器，其初始值为某个 IV。加密算法输出的最左边的 $k$ 位（最高有效位）与明文的第一个单元 $x$ 进行异或，产生密文的第一个单元 $c_1$，并传送该单元，然后将移位寄存器的内容左移 $k$ 位并将 $y_1$ 送入移位寄存器最右边的 $k$ 位，此过程一直进行到明文的所有单元都被加密为止。CBC 模式除能够获得保密性，还能用于认证。

### 3.6.4 输出反馈模式

输出反馈模式的结构类似于 CFB 模式。不同之处在于 OFB 模式是将加密算法的输出反馈到移位寄存器，而 CFB 模式是将密文单元反馈到移位寄存器。OFB 模式传输过程中的比特错误不会被传播，但是它比 CFB 模式更易受到消息流的篡改攻击。OFB 模式的特点是消息作为比特流、分组加密的输出与被加密的消息相加、比特错误不易传播等。OFB 模式如图 3-18 所示。

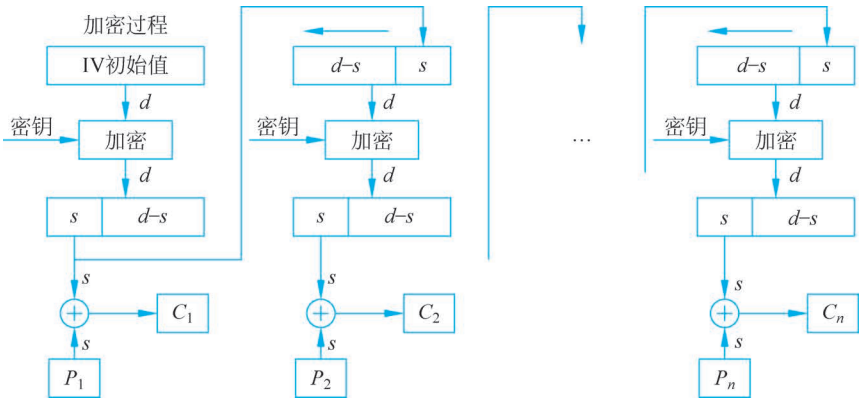


图 3-18 OFB 模式

### 3.6.5 计数器模式

计数器模式是对一系列输入数据块（称为计数）进行加密，产生一系列的输出块，输出块与明文异或得到密文。对于最后的数据块，可能是长 $k$ 位的局部数据块，这 $k$ 位就将用于异或操作，而剩下的 $d-k$ 位将被丢弃（$d$ 为块的长度）。CTR 模式加密时产生一个 16 字节的伪随机码块流，伪随机码块与输入的明文进行异或运算后产生密文输出。密文与同样的伪随机码进行异或运算后可以重新产生明文，CTR 模式如图 3-19 所示，其中 $T_i$ 为计时器，$T_i = T_i - 1 + 1 (1 \leqslant i \leqslant d)$。CTR 模式被广泛用于 ATM 网络安全和 IPsec 中。

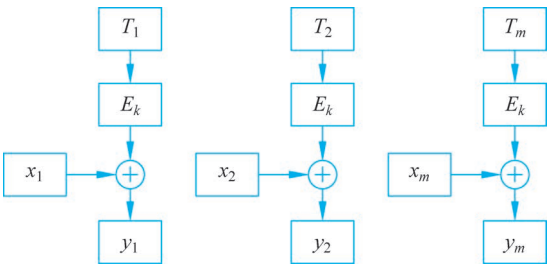


图 3-19 CTR 模式

分组密码 5 种工作模式比较如表 3-19 所示。

表 3-19 分组密码 5 种工作模式比较

| 工作模式 | 优点 | 缺点 | 应用场景 |
| --- | --- | --- | --- |
| 电子编码本模式 | 易于理解且简单易行；便于实现并行操作；没有误差传递的问题 | 不能隐藏明文的模式，若明文重复，则对应的密文也会重复，密文内容很容易被替换、重排、删除、重放；对明文进行主动攻击的可能性较高 | 适合加密密钥、随机数等短数据。例如，安全地传递 DES 密钥，ECB 是最合适的模式 |
| 密码分组链接模式 | 密文链接模式加密后的密文上下文关联，即使在明文中出现重复的信息也不会产生相同的密文；密文内容被替换、重排、删除、重放或网络传输过程中发生错误，后续密文即被破坏，无法完成解密还原；对明文的主动攻击的可能性较低 | 不利于并行计算，目前没有已知的并行运算算法；误差传递，如果在加密过程中发生错误，则错误将被无限放大，导致加密失败；需要初始化向量 | 可加密任意长度的数据；适用于计算产生检测数据完整性的消息认证码 MAC |
| 密码反馈模式 | 隐藏了明文的模式，每一个分组的加密结果必受其前面所有分组内容的影响，即使出现多次相同的明文，也均产生不同的密文；分组密码转化为流模式，可产生密钥流；可以及时加密传送小于分组的数据 | 与 CBC 模式类似。不利于并行计算，目前没有已知的并行运算算法；存在误差传送，一个单元损坏影响多个单元；需要初始化向量 | 因错误传播无界，可用于检查发现明文密文的篡改 |
| 输出反馈模式 | 隐藏了明文的模式；分组密码转化为流模式；无误差传送问题；可以及时加密传送小于分组的数据 | 不利于并行计算；对明文的主动攻击是可能的，安全性较 CFB 差 | 适用于加密冗余性较大的数据，如语音和图像数据 |
| 计数器模式 | 可并行计算；安全性至少与 CBC 模式一样好；加密与解密仅涉及密码算法的加密 | 没有错误传播，因此不易确保数据完整性 | 适用于各种加密应用 |

## 3.7 序列密码

序列密码是将明文划分成字符(如单个字母)，或其编码的基本单元(如 0、1 数字)，字符分别与密钥序列作用进行加密，解密时以同步产生的同样的密钥序列实现。序列密码体制框图如图 3-20 所示。

流密码也称序列密码，它是在“一次一密码”的追求中发展起来的一种密码。流密码强度完全依赖密钥序列的随机性和不可预测性。所有序列密码都有密钥，且密钥发生流的输出是密钥的函数。

对于无穷大的密钥集，密钥不可能重复，密钥之间没有任何相关性。由于密文 $c_i =$

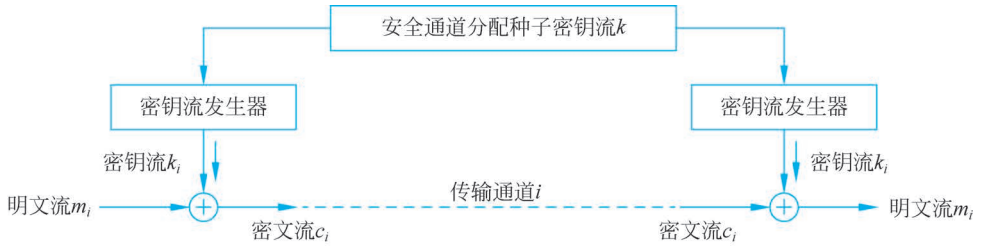


图 3-20　序列密码体制框图

$m_i \oplus k_i$，因此，很容易根据明文和密文得出密钥，即 $k_i = m_i \oplus c_i$。密钥的安全性在于不重复、不可预测。

在流密码技术中，序列密码分为同步序列密码和自同步序列密码两种。如果密钥流完全独立于明文流或密文流，则称这种流密码为同步流密码；如果密钥流的产生与明文流或密文流有关，则称这种流密码为自同步流密码。

同步序列密码要求发送方和接收方必须是同步的，在同样的位置用同样的密钥才能保证正确地解密。若在传输过程中密文序列有被篡改、删除、插入等错误导致同步失效，则不可能成功解密，只能通过重新同步来实现解密、恢复密文。在传输期间，一个密文位的改变只影响该位的恢复，不会对后继位产生影响。

自同步序列密码密钥的产生与已产生的固定数量的密文位有关，因此，密文中产生的一个错误会影响后面有限位的正确解密。所以，自同步密码的密码分析比同步密码的密码分析更加困难。

流密码的重要部件是密钥流生成器，希望密钥流生成器产生的密钥流是完全随机的；但是，实际使用的密钥流序列都是根据一定的算法生成的，因此不可能做到完全随机。人们提出使用密钥序列产生器来实现密钥流生成器。

密钥序列产生器可分成驱动部分和非线性组合部分。其中：驱动部分产生控制生成器的状态序列，并控制生成器的周期和统计特性；非线性组合部分对驱动部分的各个输出序列进行非线性组合，控制和提高产生器输出序列的统计特性、线性复杂度和不可预测性等，从而保证输出密钥序列的安全强度。密钥序列产生器组成如图 3-21 所示。

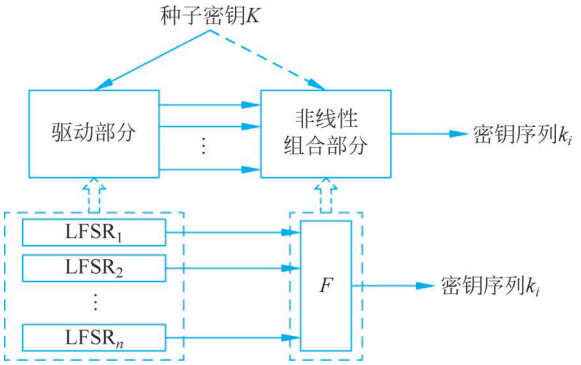


图 3-21　密钥序列产生器

驱动部分常用一个或多个线性反馈移位寄存器(Linear Feedback Shift Register, LFSR)实现,非线性组合部分用非线性组合函数 $F$ 实现。GF(2)上一个 $n$ 级反馈移位寄存器由 $n$ 个二元存储器和一个反馈函数 $f(a_1,a_2,\cdots,a_n)$组成,如图 3-22 所示。

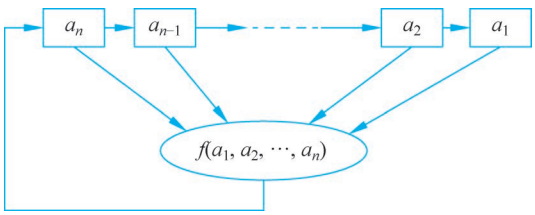


图 3-22 GF(2)上的 $n$ 级反馈移位寄存器

序列密码属于对称密码体制,与分组密码相比较:分组密码把明文分成相对比较大的块,对于每块使用相同的加密函数进行处理。分组密码是无记忆的。序列密码处理的明文长度为 1 位,而且序列密码是有记忆的。序列密码又称为状态密码,因为它的加密不仅与密钥和明文有关,而且与当前状态有关。两者区别不是绝对的,分组密码增加少量的记忆模块就形成了序列密码。

序列密码具有实现简单、便于硬件计算、加密与解密处理速度快、低错误(没有或只有有限位的错误)传播等优点,但也暴露出对错误的产生不敏感的缺点。序列密码涉及大量的理论知识,许多研究成果并没有完全公开,因为序列密码目前主要用于军事和外交等机要部门。目前,公开的序列密码主要有 RC4、SEAL 等。

分组密码是对一个大的明文数据块(分组)进行固定变换的操作,可以很容易采用软件实现。序列密码是对单个明文位的随时间变换的操作,更适合用硬件实现。尽管分组和序列密码非常不同,但分组密码也可作为序列密码使用,反之亦然。

### 3.7.1 A5 算法

A5 是用于 GSM 系统的序列密码算法,实现对从电话到基站连接的加密。A5 算法的特点是效率高,适合硬件上高效实现,能通过已知的统计检验。起初该算法的设计没有公开,但被泄露。

A5 算法由线性反馈移位寄存器 $R_1$、$R_2$、$R_3$ 组成,长度分别是 $n_1=19$,$n_2=22$ 和 $n_3=23$,A5 算法移位寄存器如图 3-23 所示。它们的特征多项式分别为

$$f_1(x)=x^{19}+x^5+x^2+x+1$$

$$f_2(x)=x^{22}+x+1$$

$$f_3(x)=x^{23}+x^{15}+x^2+x+1$$

所有的反馈多项系数都较少。3 个 LFSR 的异或值作为输出。A5 算法通过“停/走”式钟控方式相连。A5 算法原理图如图 3-24 所示。

图 3-24 中,$S_{i,j}$ 表示 $t$ 时刻、$R_i$ 的状态向量的第 $j$ 个比特;$\tau_1=10$,$\tau_2=11$,$\tau_3=12$。钟控函数为

$$c(t)=g(S_{1,\tau_1}(t-1),S_{2,\tau_2}(t-1),S_{3,\tau_1}(t-1))$$

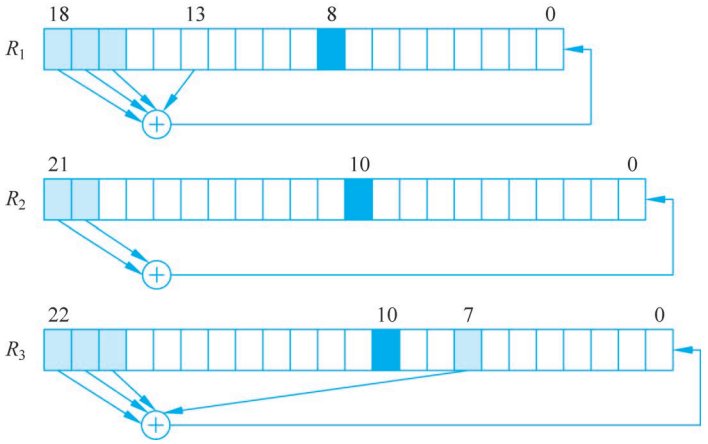


图 3-23　A5 算法移位寄存器

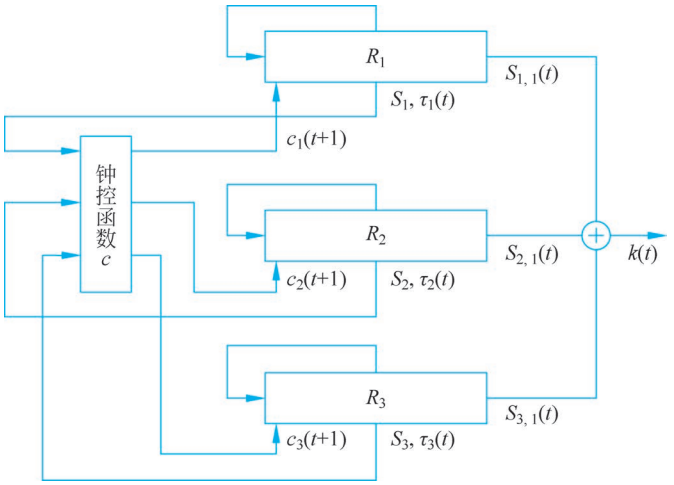


图 3-24　A5 算法原理图

是一个四值函数：

$$g(S_1,S_2,S_3)=\begin{cases}\{1,2\}, & S_1=S_2\neq S_3\\ \{1,3\}, & S_1=S_3\neq S_2\\ \{2,3\}, & S_2=S_3\neq S_1\\ \{1,2,3\}, & S_1=S_2=S_3\end{cases}$$

$R_i$ 的“停/走”规则：当 $i\in c(t)$ 时，$R_i$ 走；否则，$R_i$ 停。

A5 算法的密钥 $K$ 是 64 位。顺次填入为 $R_1$、$R_2$、$R_3$ 的初始状态，然后经过 100 次的初始化运算，不输出。加密过程：首先为通信的一个方向生成 114 位的密钥序列，然后空转 100 次，接着为通信的另一个方向生成 114 位的密钥序列，以此类推。用密钥序列与明文序列按位模 2 相加得到相应的密文，对方用密钥序列与密文序列按位模 2 相加得到相应的明文。

A5 算法的统计性很好，但是移位寄存器太短，容易遭受穷举攻击。A5 算法把主密钥作为算法中 3 个寄存器的初始值，长度为 64 位。如果利用已知明文攻击，知道其中两个寄存器的初始值就可以计算出另一个寄存器的初始值，$2^{40}$ 步就可以得出寄存器 LFSR-1 和 LFSR-2 的结构。此外，A5 算法还有一个冲突问题，即不同寄存器初始值可能产生相同的密钥流，实验显示这种可能性高达 30%，且合成后的密钥流的非线性度非常差。

### 3.7.2 祖冲之算法

祖冲之算法是一个基于字设计的同步序列密码算法，其种子密钥 SK 和 IV 的长度均为 128 位，在种子密钥 SK 和 IV 的控制下，每拍输出一个 32 位的密钥字。ZUC 算法采用过滤生成器结构设计，在线性驱动部分首次采用素域 GF($2^{31}-1$)上的 $m$ 序列作为源序列，具有周期大、随机统计特性好等特点，且在二元域上是非线性的，可以提高抵抗二元域上密码分析的能力；过滤部分采用有限状态机设计，内部包含记忆单元，使用分组密码中扩散和混淆特性好的线性变换和 S 盒，可提供高的非线性。ZUC 算法结构主要包含三层，上层为线性反馈移位寄存器，中间层为比特重组(BR)，下层为非线性函数 $F$，如图 3-25 所示。

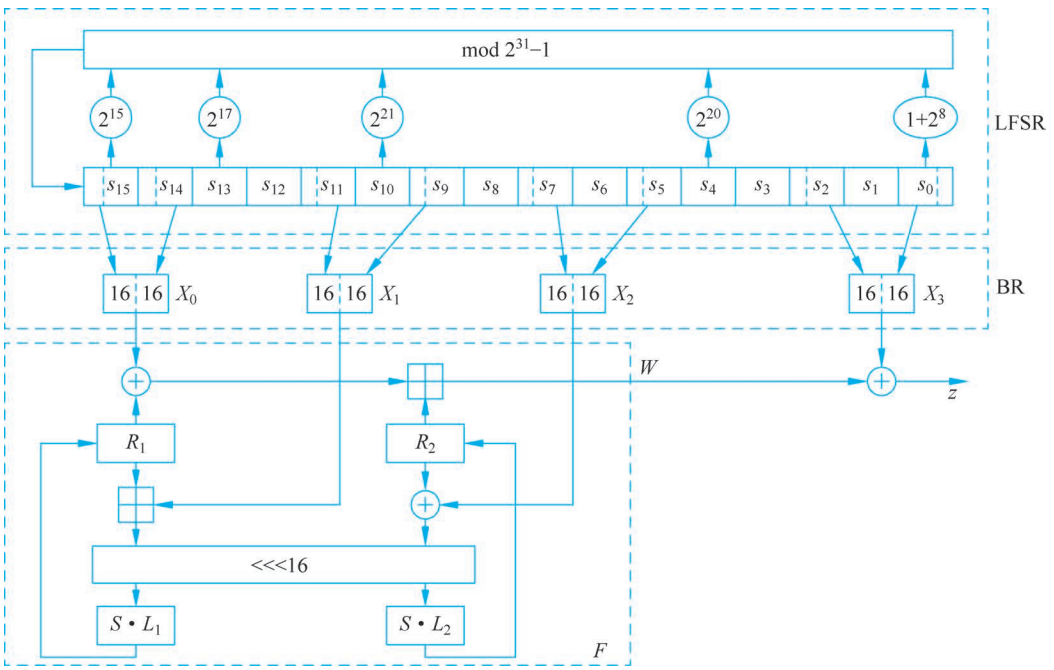


图 3-25 ZUC 算法结构

线性反馈移位寄存器采用素域 GF($2^{31}-1$)上的本原序列，主要提供周期大、统计特性好的源序列。由于素域 GF($2^{31}-1$)上的加法在二元域 GF(2)上是非线性的，素域 GF($2^{31}-1$)上本原序列可视作二元域 GF(2)上的非线性序列，其具有权位序列平移等价、大的线性复杂度和好的随机统计特性等特点，并在一定程度上提供好的抵抗现有的

基于二元域的密码分析的能力，如二元域上的代数攻击、相关攻击和区分分析等。

LFSR 包括 16 个 31 位寄存器单元变量 $s_0, s_1, \cdots, s_{15}$。LFSR 运行以下两种模式。

(1) 初始化模式：LFSR 接收 1 个 31 位字 $u$ 的输入，对寄存器单元变量 $s_0, s_1, \cdots, s_{15}$ 进行更新，计算过程如下。

① $v = 2^{15}s_{15} + 2^{17}s_{13} + 2^{21}s_{10} + 2^{25}s_4 + (1+2^8)s_0 \bmod (2^{31}-1)$。

② $s_{16} = (v+u) \bmod (2^{31}-1)$。

③ 若 $s_{16}=0$，则置 $s_{16} = 2^{a1}-1$。

④ $(s_1, s_2, \cdots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \cdots, s_{11}, s_{15})$。

(2) 工作模式：LFSR 无输入，直接对寄存器单元变量 $s_{11}, s_1, \cdots, s_{15}$ 进行更新，计算过程如下。

① $s_{16} = 2^{15}s_{15} + 2^{17}s_{13} + 2^{21}s_{10} + 2^{30}s_4 + (1+2^8)s_{10} \bmod (2^{31}-1)$。

② 若 $s_{16}=0$，则置 $s_{16} = 2^{31}-1$。

③ $(s_1, s_2, \cdots, s_{15}, s_{16}) \rightarrow (s_0, s_1, \cdots, s_{14}, s_{15})$。

比特重组主要功能是衔接 LFSR 和 $F$，将上层 31 位数据转化为 32 位数据以供 $F$ 使用。比特重组采用软件实现友好的移位操作和字符串连接操作，其主要目的是打破 LFSR 的线性代数结构，并在一定程度上提供抵抗素域 $GF(2^{31}-1)$ 上的密码攻击的能力。

输入为 LFSR 单元变量 $s_0, s_2, s_5, s_7, s_9, s_{11}, s_{14}, s_{15}$，输出为 4 个 32 位字 $X_0$、$X_1$、$X_2$、$X_3$。计算过程如下。

(1) $X_n = s_{15H} \| s_{t-1}$。

(2) $X_1 = s_{11} \| s_{9H}$。

(3) $X_2 = s_{iL} \| s_{5H}$。

(4) $X_3 = s_{2L} \| s_{0H}$。

式中："$\|$"表示为字符串或字节串连接符。

非线性函数主要借鉴了分组密码的设计思想，采用具有最优差分/线性分支数的线性变换和密码学性质优良的 S 盒来提供好的扩散性和高的非线性性。此外，非线性函数基于 32 位的字设计，采用异或、循环移位、模 2 加、S 盒等不同代数结构上的运算，打破源序列在素域 $GF(2^{31}-1)$ 上的线性代数结构，进一步提高算法抵抗素域 $GF(2^{31}-1)$ 上的密码分析能力。

非线性函数包含 2 个 32 位记忆单元变量 $R_1$ 和 $R_2$。非线性函数的输入为 3 个 32 位字 $X_0$、$X_1$、$X_2$，输出为一个 32 位字 $W$。$F(X_0, X_1, X_2)$ 计算过程如下。

(1) $W = (X_n \oplus R_1) \boxplus R_z$。

(2) $W_1 = R_1 \boxplus X_{17}$。

(3) $W_2 = R_2 \oplus X_2$。

(4) $R_1 = S[L_1(W_{1L} \| W_{2H})]$。

(5) $R_2 = S[L_2(W_{2L} \| W_{1H})]$。

式中："$\boxplus$"为模 $2^{32}$ 加法运算；$S$ 为 32 位的 S 盒变换；$L_1$ 和 $L_2$ 为 32 位线性变换，定义为

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

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

ZUC 算法密钥装入时将初始密钥 $k$ 和初始向量 iv 分别扩展为 16 个 31 位字作为 LFSR 单元变量 $s_0,s_1,\cdots,s_{15}$ 的初始状态。

(1) 设 $k$ 和 iv 分别为 $k_0\|k_1\|\cdots\|k_{15}$ 和 $\mathrm{iv}_0\|\mathrm{iv}_1\|\cdots\|\mathrm{iv}_{15}$，其中 $k_i$ 和 $\mathrm{iv}_i$ 均为 8 位字节，$0\leqslant i\leqslant 15$。

(2) 对于 $0\leqslant i\leqslant 15$，有 $s_i=k_i\|d_i\|\mathrm{iv}_i$。这里 $d_i$ 为 16 位的常量串，定义如下：

$$d_0=100010011010111_2$$
$$d_1=010011010111100_2$$
$$d_2=110001001101011_2$$
$$d_3=001001101011110_2$$
$$d_4=101011110001001_2$$
$$d_5=011010111100010_2$$
$$d_6=111000100110101_2$$
$$d_7=000100110101111_2$$
$$d_8=100110101111000_2$$
$$d_9=010111100010011_2$$
$$d_{10}=110101111000100_2$$
$$d_{11}=001101011110001_2$$
$$d_{12}=101111000100110_2$$
$$d_{13}=011110001001101_2$$
$$d_{14}=111100010011010_2$$
$$d_{15}=100011110101100_2$$

ZUC 算法的输入参数为初始密钥 $k$，初始向量 iv 和正整数 $L$，输出参数为 $L$ 个密钥字 $Z$。算法运行过程包含初始化步骤和工作步骤。还要用到 S 盒和线性变换 $L$ 两种固定的运算工具，它们也是密码算法中经常用到的重要组成部分。

## 3.8 非对称密码

Diffie 与 Hellman 首先提出了非对称密码的概念，有效地解决了秘密密钥密码系统通信双方密钥共享困难的问题，并引进了创新的数字签名的概念。非对称密码可为加解密或数字签名。由于加密或签名验证密钥是公开的，因此称为公钥；由于解密或签名产生密钥是秘密的，因此称为私钥。因为公钥与私钥不同，且公钥与私钥必须存在成对与唯一对应的数学关系，使得由公钥去推导私钥在计算上不可行，因此非对称密码又称为公开密钥或双钥。公钥密码模型如图 3-26 所示。

用户 A 用加密算法 $E$ 和密钥 pk 对明文 $m$ 进行加密，用户 B 用解密算法 $D$ 和密钥

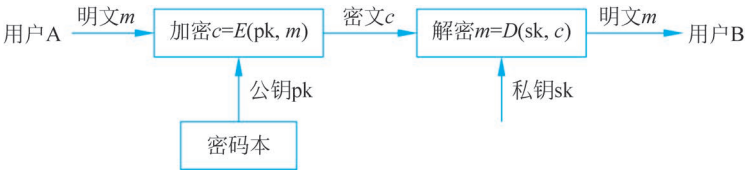


图 3-26　公钥密码模型

sk 对密文 $c$ 进行解密。加密密钥 pk 是公开的，而解密密钥 sk 是保密的，只有接收方知道。公开密钥加密算法的原则如下。

(1) 产生密钥对(公钥 pk 和私钥 sk)在计算上是容易的；对消息 $m$ 加密产生密文 $c$，即 $c=E_{pk}(m)$容易计算；接收方用自己的私钥对 $c$ 解密，即 $m=D_{sk}(c)$容易计算。

(2) 通过公钥 pk 求私钥 sk 在计算上是不可行的；由密文 $c$ 和公钥 pk 恢复明文 $m$ 在计算上是不可行的。

(3) 加密和解密次序可换，即 $E_{pk}(D_{sk}(m))=D_{sk}(E_{pk}(m))$。

以上三条要求的本质是需要一个单向陷门函数。设 $f$ 是一个函数，如果对任意给定的 $x$，计算 $y$ 使得 $y=f(x)$是容易解的，但对任意给定的 $y$，计算 $x$ 使得 $f(x)=y$ 是难解的，即求 $f$ 的逆函数是难解的，则 $f$ 称为单向函数。

单向函数是集合 $X$、$Y$ 之间的一个映射，使得 $Y$ 中每个元素 $y$ 都有唯一一个原像 $x\in X$，且由 $x$ 易于计算它的像 $y$，由 $y$ 计算它的原像 $x$ 是不可行的。易于计算是指函数值能在其输入长度的多项式时间内求出，即如果输入长 $n$ 位，那么求函数值的计算时间是 $n^a$ 的某个倍数，其中 $a$ 是固定的常数。这时称求函数值的算法属于多项式类 $P$，否则就是不可行的。例如，函数的输入是 $n$ 位，如果求函数值所用的时间是 $2^n$ 的倍数，那么认为求函数值是不可行的。可见，单向函数是一组可逆函数 $f$，满足以下条件。

(1) $y=f(x)$易于计算(当 $x$ 已知时，求 $y$)。

(2) $x=f^{-1}(y)$在计算上是不可行的(当 $y$ 已知时，求 $x$)。

设 $f$ 是一个函数，$t$ 是与 $f$ 有关的参数，对任意给定的 $s$，计算 $y$ 是容易的。若参数 $t$ 未知时，$f$ 的逆函数是难解的，但参数 $t$ 已知时，$f$ 的逆函数是容易解的，则 $f$ 称为单向陷门函数，参数 $t$ 称为陷门。

研究公钥密码算法就是要找出合适的单向陷门函数。用来构造密码算法的单向函数是单向陷门函数，即对密码攻击者来讲，当 $y$ 已知时，计算 $x$ 是困难的；但对合法的解密者来讲，可利用一定的陷门知识计算 $x$。

设 $f$ 是定义在有限域 GF($p$)上的指数函数，其中 $p$ 是大素数，即 $f(x)=g^x$，$x\in$ GF($p$)，$x$ 是满足 $0\leqslant x<p-1$ 的整数，其逆运算是 GF($p$)上的对数运算，即给定 $y$，寻找 $x(0\leqslant x<p-1)$，使得 $y=g^x$。当 $p$ 充分大时，也就是计算 $x=\log g^y$。可以看出：给定 $x$，计算 $y=f(x)=g^x$ 是容易的；当 $p$ 充分大时，计算 $x=\log g^y$ 是困难的。

下面给出基于单向陷门函数设计非对称密码体制的标准模式。假设给定一个陷门单向函数，可以构造如下公钥加密方案。

(1) 密钥生成：令 pk=$f$，sk=$f^{-1}$。

(2) 加密算法：已知消息 $m$ 和接收方公钥 pk，密文 $c=f(m)$正向计算容易。

(3) 解密算法：已知密文 $c$ 和私钥 sk，计算明文 $m=f^{-1}(c)$。在已知私钥的情况下，$m=f^{-1}(c)$计算容易。在不知道私钥的情况下，$m=f^{-1}(c)$计算困难。

非对称密码体制中的公钥用于单向陷门函数的正向加密运算，私钥用于反向解密运算。

非对称密码的每个用户只需要保护自己的私钥，同时，$N$ 个用户仅需要产生 $N$ 对密钥，密钥数量少，便于管理。密钥分配简单，不需要秘密的通道和复杂的协议来传送密钥，就可以实现数字签名。但是，非对称密码体制与对称密码体制相比，加密、解密处理速度较慢。同等安全强度下，非对称密码体制的密钥位数要求多一些。常用的非对称加密的算法有 RSA、Elgamal、背包算法、Rabin、D-H、ECC 等。

### 3.8.1 RSA 算法

RSA 算法是由 R. Rivest、A. Shamir 和 I. Adleman 提出的一种基于数论方法，也是理论上最为成熟的公开密钥体制，并已经得到广泛应用。RSA 算法私钥的安全性取决于密钥长度 $n$，当 $n>1024$ 时，根据目前的计算能力，RSA 算法私钥的安全性是可以保证的。但 $n$ 越大，加密和解密运算越复杂。

在介绍 RSA 算法之前，首先需要掌握 RSA 算法的因子分解基础问题。根据数论，任意大于 1 的整数能够表达成素数的乘积，即对于任意整数 $a>1$，有 $a=p_1p_2\cdots p_n$，$p_1\leqslant p_2\leqslant\cdots\leqslant p_n$，其中，$p_1,p_2,\cdots,p_n$ 是素数。但是，当 $a$ 很大时，对 $a$ 的分解是相当困难的。RSA 算法的安全性是建立在大数分解为素因子困难性基础上的。RSA 算法如下。

(1) 通信实体选择两个大的素数 $p$、$q$。

(2) 计算 $n=pq$，$\phi(n)=(p-1)(q-1)$。

(3) 选择 $e$，使得 $e$ 远小于 $\phi(n)$，并且 $\gcd(e,\phi(n))=1$，即 $e$ 和 $\phi(n)$的最大公约数为 1。

(4) 求 $d$，使得 $ed=1 \bmod \phi(n)$。

(5) 发布$(n,e)$，即公钥为$(n,e)$；自己秘密保存私钥 $d$ 并销毁 $p$ 和 $q$。

假设 A 要使用 RSA 算法加密消息并通过网络发送给 B，那么 A 应当按照以下步骤进行。

(1) A 从权威机构获得 B 的公钥$(n,e)$。

(2) A 首先将消息表示为一整数 $m$，使得 $m<n$。

(3) 计算 $c=m^e \bmod n$，计算结果 $c$ 即为密文。

(4) 通过网络将密文 $c$ 发送给 B。

当 B 收到密文 $c$ 以后只需一步计算就可以进行解密，$m=c^d \bmod n$。解密过程的正确性可以利用欧拉定理得到证明：

$$c^d \bmod n=(m^e \bmod n)^d=(m^e)^d \bmod n=m^{ed} \bmod n$$
$$=m^{k\varphi(n)+1} \bmod n=(m^{k\varphi(n)} \bmod n)(m \bmod n)$$

$$=1\times m=m$$

在 RSA 算法中，最重要的是 A 对 $p$ 和 $q$ 的选择，选择不恰当将会极大地降低 RSA 算法的安全性。一般来说，$p$ 和 $q$ 数值不能太接近，并且 $p-1$ 和 $q-1$ 都有大的素因子，$\gcd(p-1,q-1)$应该很小。通常选择 $p$ 使得 $p$ 和$(p-1)/2$ 都是素数。此外，使用 RSA 算法的用户应该遵守用户之间不能使用同一个 $n$。如果多个用户使用同一个 $n$，就可能对 $n$ 进行因数分解，从而可能计算出用户的私钥。

目前，通常认为 512 位的密钥已经不够安全，推荐采用 1024 位。另外，RSA 算法采用的幂模运算比 AES 的操作要慢得多。由此可见，对称密码算法的加密速度是非对称密码算法的速度的 100 倍以上也就不足为奇。

### 3.8.2 Elgamal 算法

Elgamal 算法的安全性是建立在有限域上的离散对数很难计算这一数学难题的基础上。

数论中的离散对数指的是：设 $p$ 为奇素数，$g$ 为 $z_p$ 的原根，$p$ 不能整除整数 $y$，则存在整数 $k$($k$ 为 $y$ 对模数 $p$ 的离散对数，$0\leqslant k<p-1$)使得 $y\equiv g^k \bmod p$。

假设 A 希望用 Elgamal 算法加密消息，那么 A 首先选取一个很大的素数 $p$ 和 $p$ 的一个原根 $g$；然后，A 选择一个秘密密钥 $a$($0<a<p-1$)，计算 $b\equiv g^a \bmod p$，公钥 $k=(g,b,p)$。如果 B 希望向 A 发送消息 $m$，那么加密方法如下。

(1) B 从权威机构或 A 处获得其公钥 $k$。

(2) B 任选一个秘密的整数 $t$($1<t<p-1$)。

(3) B 计算 $y_1\equiv g^t \bmod p$，$y_2\equiv mb^t \bmod p$，然后向 A 发送密文$(y_1,y_2)$。

A 收到密文之后，计算 $y_2(y_1)^{-a} \bmod p$，就可以得到正确的解密结果。其中，$y_1^{-1}$ 的定义是 $y_1y_1^{-1}\equiv 1 \bmod p$。解密过程的正确性证明如下：

$$y_2(y_1)^{-a} \bmod p \equiv (mb^t \bmod p)(g^t \bmod p)^{-a} \bmod p \equiv mb^t(g^t)^{-a} \bmod p$$

$$\equiv mb^t(g^a)^{-t} \bmod p \equiv mb^tb^{-t} \bmod p \equiv m$$

需要注意的是，为了确保 Elgamal 算法的安全性，$p$ 通常具有 150 位以上的十进制数字，约为 512 位的二进制数，而且 $p-1$ 至少有一个大的素因子。在满足上述条件的情况下，根据 $p$、$g$、$b$ 计算离散对数是相当困难的。

Elgamal 算法非对称密码体制可以在计算离散对数困难的任何群中实现。但是，通常使用有限域，但不局限于有限域，比如，其还可以在圆锥曲线群或椭圆曲线群上实现。

### 3.8.3 椭圆曲线密码算法

RSA 算法解决分解整数问题需要亚指数时间复杂度，且目前已知计算椭圆曲线离散对数问题(ECDLP)的最好方法都需要使用全指数时间复杂度。这意味着在椭圆曲线系统中只需要使用相对于 RSA 算法短得多的密钥就可以达到与其相同的安全强度。例如，一般认为 160 位的椭圆曲线密钥提供的安全强度与 1024 位 RSA 密钥相当。使用短

的密钥的好处是加解密速度快、节省能源、节省带宽、存储空间。比特币以及中国的二代身份证都使用了256位的椭圆曲线密码算法。

椭圆曲线密码算法的安全性是基于椭圆曲线离散对数问题的困难性。目前人们普遍认为，椭圆曲线离散对数问题要比大整数因子分解问题和有限域上的离散对数问题难解得多。目前还没有找到求解椭圆曲线离散对数的亚指数算法，因此，椭圆曲线密码体制可以使用更短的密钥就可以获得相同的安全性。ECC算法的优点如下。

(1) 安全性高。安全性基于椭圆曲线上的离散对数问题的困难性，目前还没找到解决椭圆曲线上离散对数问题的亚指数时间算法。而大数因子分解和离散对数的求解都存在亚指数时间算法。

(2) 短密钥。随着密钥长度的增加，求解椭圆曲线上离散对数问题的难度比同等长度大数因子分解和求解离散对数问题的难度要大得多。如表3-20所示，椭圆曲线密码算法仅需要更小的密钥长度就可以提供RSA算法相当的安全性，因此可以减少处理负荷。

表3-20 密钥长度比较

| ECC密钥长度/位 | 106 | 132 | 160 | 220 | 600 |
|---|---|---|---|---|---|
| RSA密钥长度/位 | 512 | 768 | 1024 | 2048 | 21000 |
| 破解时间/MIPS年 | 104 | 108 | 1011 | 1020 | 1078 |
| ECC/RSA密钥长度比例 | 1∶5 | 1∶6 | 1∶7 | 1∶10 | 1∶35 |

(3)灵活性好。改变曲线的参数可以得到不同的曲线，形成不同的循环群，构造密码算法具有多选择性。

椭圆曲线的椭圆一词来源于椭圆周长积分公式。椭圆曲线并非椭圆，之所以称为椭圆曲线，是因为它的曲线方程与计算椭圆周长的方程相似。一条椭圆曲线是在射影平面上满足威尔斯特拉斯(Weierstrass)方程所有点的集合：

$$Y^2Z+a_1XYZ+a_3YZ^2=X^3+a_2X^2Z+a_4XZ^2+a_6Z^3$$

对普通平面上的点$(x,y)$，令$x=X/Z$，$y=Y/Z$，$Z\neq0$，得到如下方程：

$$y^2Z^3+a_1xyZ^3+a_3yZ^3=x^3Z^3+a_2x^2Z^3+a_4xZ^3+a_6Z^3$$

上式化简，可得

$$y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a_6$$

简化版的威尔斯特拉斯方程为

$$E:y^2=x^3+ax+b$$

其中要求曲线是非奇异的(处处可导)，有$4a^3+27b^2\neq0$，用来保证曲线是光滑的，即曲线的所有点都没有两个或者两个以上的不同的切线。$a,b\in K$，$K$为$E$的基础域。点$O\infty$是曲线的唯一的无穷远点。

如果椭圆曲线上的3个点位于同一直线上，那么它们的和为$O$。

(1) $O$为加法的组织元，对于椭圆曲线上的任何一点$P$，有$P+O=P$。

(2) 对于椭圆曲线上的一点$P=(x,y)$，它的逆元为$-P=(x,-y)$。注意到这里有

$P+(-P)=P-P=O$。

（3）设 $P$ 和 $Q$ 是椭圆曲线上 $x$ 坐标不同的两点，$P+Q$ 的定义如下：作一条通过 $P$ 和 $Q$ 的直线 $l$ 与椭圆曲线相交于 $R$，然后过 $R$ 点作 $y$ 轴的平行线 $l'$，$l'$ 与椭圆曲线相交的另一点 $S$ 就是 $P+Q$，如图 3-27 所示。

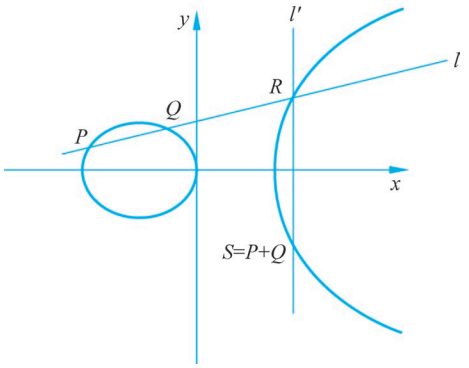


图 3-27　椭圆曲线示例

对于椭圆曲线上不互为逆元的两点 $P=(x_1,y_1)$ 和 $Q=(x_2,y_2)$，$S=P+Q=(x_3,y_3)$ 由以下规则确定：

$$x_3=\lambda^2-x_1-x_2$$

$$y_3=\lambda(x_1-x_3)-y_1$$

式中

$$\lambda=\begin{cases}\dfrac{y_2-y_1}{x_2-x_1}, & P\neq Q\\[2ex] \dfrac{3x_1^2+a}{2y_1}, & P=Q\end{cases}$$

椭圆曲线是连续的，并不适合用于加密，必须将椭圆曲线变成离散的点，把椭圆曲线定义在有限域上。

设 $G$ 是椭圆曲线 $Ep(a,b)$ 上的一个循环子群，$P$ 是 $G$ 的一个生成元，$Q\in G$。已知 $P$ 和 $Q$，求满足 $mP=Q$ 的整数 $m$，$0\leqslant m\leqslant \mathrm{ord}(P)-1$，称为椭圆曲线上的离散对数问题。其中椭圆曲线 $Ep(a,b)$ 上点 $P$ 的阶是指满足

$$nP=\underbrace{P+P+\cdots+P}_{n}=O$$

的最小正整数，记为 $\mathrm{ord}(P)$，其中 $O$ 是无穷远点。

在使用一个椭圆曲线密码时，首先需要将发送的明文 $m$ 编码为椭圆曲线上的点 $P_m=(x_m,y_m)$，然后对点 $P_m$ 做加密变换，在解密后还得将 $P_m$ 逆向译码才能获得明文。在椭圆曲线 $Ep(a,b)$ 上选取一个阶为 $n$（$n$ 为一个大素数）的生成元 $P$。随机选取整数 $x(1<x<n)$，计算 $Q=xP$。公钥为 $Q$，私钥为 $x$。为了加密 $P_m$，随机选取一个整数 $k$，$1<k<n$，计算 $C_1=kP$，$C_2=P_m+kQ$，则密文 $c=(C_1,C_2)$。为了解密一个密文 $c=(C_1,C_2)$，计算

$$C_2 - xC_1 = P_m + kQ - xkP = P_m + kxP - xkP = P_m$$

攻击者要想从 $c=(C_1,C_2)$ 计算出 $P_m$，就必须知道 $k$。而要从 $P$ 和 $kP$ 中计算出 $k$ 将面临求解椭圆曲线上的离散对数问题。

在有限域 GF($p$)确定的情况下，就确定了其上的循环群。而 GF($p$)上的椭圆曲线则可以通过改变曲线参数得到不同的曲线，形成不同的循环群。因此，椭圆曲线具有丰富的群结构和多选择性。也正因如此，椭圆曲线密码体制能够在保持与 RSA 算法、DSA 算法体制相同安全性的情况下大大缩短密钥长度。

由于在相同安全性下 ECC 算法比 RSA 算法的私钥位长及系统参数小得多，这意味着应用 ECC 算法所需的存储空间要小得多，传输所用的带宽要求更低，硬件实现 ECC 算法所需逻辑电路的逻辑门数要较 RSA 算法少得多，功耗更低，这使得 ECC 算法比 RSA 算法更适合实现到低功耗要求的移动通信设备、无线通信设备和智能卡等资源严重受限制的设备中。

### 3.8.4 SM2 算法

要保证 SM2 算法的安全性，就要使所选取的曲线能够抵抗各种已知的攻击，这就涉及选取安全椭圆曲线的问题。用于建立密码体制的椭圆曲线的主要参数有 $p$、$a$、$b$、$G$、$n$ 和 $h$。其中：$p$ 为有限域 $F(p)$中元素的数目；$a$、$b$ 为方程中的系数，取值于 $F(p)$；$G$ 为基点(生成元)；$n$ 为点 $G$ 的阶；$h$ 为椭圆曲线上点数 $N$ 除以 $n$ 的结果，也称余因子。为了使所建立的密码体制有较好的安全性，这些参数的选取应满足如下条件。

(1) $p$ 越大越安全，但计算速度会变慢，160 位可以满足目前的安全需求。

(2) 为了防止 Pohlig-Hellman 算法的攻击，$n$ 为大素数($n>2^{160}$)，对于固定的有限域 $F(p)$，$n$ 应当尽可能大。

(3) 因为 $x^3+ax+b$ 无重复因子才可基于椭圆曲线 $E_p(a,b)$定义群，所以要求 $4a^3+27b^2\neq 0(\mathrm{mod}\ p)$。

(4) 为了防止小步大步攻击，要保证 $P$ 的阶 $n$ 足够大，要求 $h\leqslant 4$。

(5) 为了防止 MOV 规约法，不能选取超奇异椭圆曲线和异常椭圆曲线等两类特殊曲线。

SM2 中规定发送方用接收方的公钥将消息加密成密文，接收方用自己的私钥对收到的密文进行解密还原成原始消息。用户 B 的密钥对包括其私钥 $d_B$ 和公钥 $P_B=[d_B]G$。

SM2 也需要使用密钥派生函数 KDF($Z$,klen)，具体如下。

输入：比特串 $Z$，整数 klen(表示要获得的密钥数据的位长，要求该值小于 $(2^{32}-1)v$)。

输出：长为 klen 的密钥数据比特串 $K$。

(1) 初始化一个 32 位构成的计数器 ct=0×00000001。

(2) 对 $i$ 从 1 到⌈klen/$v$⌉执行，计算 $Ha_i=H_v(Z\|ct)$；ct 加 1。

(3) 若 klen/$v$ 是整数，则令 $Ha!⌈\mathrm{klen}/v⌉=Ha_{⌈\mathrm{klen}/v⌉}$；否则，令 $Ha!_{⌈\mathrm{klen}/v⌉}$为

$Ha_{\lceil \mathrm{klen}/v \rceil}$ 最左边的(klen－($v$×⌊klen/$v$ ⌋))位。

(4) 令 $K=Ha_1 \| Ha_2 \| \cdots \| Ha_{i\mathrm{ken}/v\rceil-1} \| Ha!\ [\mathrm{klen}/v]$。

设需要加密的消息为比特串 $M$,klen 为 $M$ 的位长。为了对明文 $M$ 进行加密,加密用户 A 应实现以下运算步骤。

(1) 用随机数发生器产生随机数 $k \in [1, n-1]$。

(2) 计算椭圆曲线点 $C_1=[k]G=(x_1, y_1)$,将 $C_1$ 的数据类型转换为比特串。

(3) 计算椭圆曲线点 $S=[h]P_B$,若 $S$ 是无穷远点,则报错并退出。

(4) 计算椭圆曲线点 $[k]P_B=(x_2, y_2)$,将坐标 $x_2$、$y_2$ 的数据类型转换为比特串。

(5) 计算 $t=\mathrm{KDF}(x_2 \| y_2, \mathrm{klen})$,若 $t$ 为全 0 比特串,则需要重新选择随机数 $k$。

(6) 计算 $C_2=M \oplus t$。

(7) 计算 $C_3=\mathrm{Hash}(x_2 \| M \| y_2)$。

(8) 输出密文 $C=C_1 \| C_3 \| C_2$。

SM2 加密流程如图 3-28 所示。

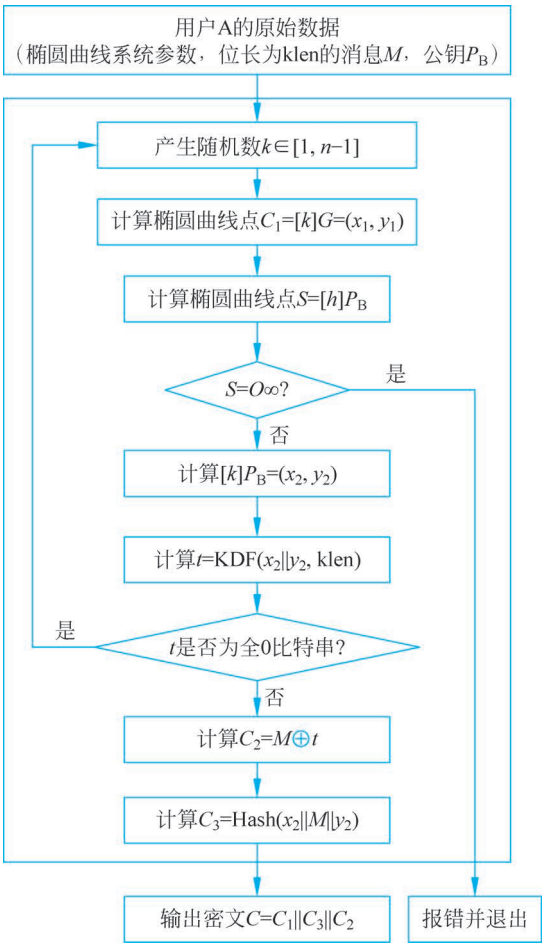


图 3-28　SM2 加密流程

对于 SM2 公钥解密算法，设 klen 为密文中 $C_2$ 的位长。为了对密文 $C=C_1 \| C_3 \| C_2$ 进行解密，解密用户 B 应实现以下运算步骤。

(1) 从 $C$ 中取出比特串 $C_1$，将 $C_1$ 的数据类型转换为椭圆曲线上的点，验证 $C_1$ 是否满足椭圆曲线方程。若不满足，则报错并退出。

(2) 计算椭圆曲线点 $S=[h]C_1$，若 $S$ 是无穷远点，则报错并退出。

(3) 计算 $[d_B]C_1=(x_2,y_2)$，将坐标 $x_2$、$y_2$ 的数据类型转换为比特串。

(4) 计算 $t=\mathrm{KDF}(x_2 \| y_2,\mathrm{klen})$，若 $t$ 为全 0 比特串，则报错并退出。

(5) 从 $C$ 中取出比特串 $C_2$，计算 $M'=C_2 \oplus t$。

(6) 计算 $u=\mathrm{Hash}(x_2 \| M' \| y_2)$，从 $C$ 中取出比特串 $C_3$；若 $u \neq C_3$，则报错并退出。

(7) 输出明文 $M'$。

## 3.9 公钥基础设施

公开密钥基础设施(Public Key Infrastructure，PKI)是以非对称密钥加密技术为基础，以数据机密性、完整性、身份认证和行为不可抵赖性为安全目的，实施和提供安全服务的具有普适性的安全基础设施。其内容包括数字证书、非对称密钥密码技术、认证中心、证书和密钥的管理、安全代理软件、不可否认性服务、时间邮戳服务、相关信息标准、操作规范等。

### 3.9.1 PKI 总体架构

一个网络的 PKI 包括以下基本构件。

(1) 数字证书：由认证机构经过数字签名后发给网上信息主体的一段电子文档，包括主体名称、证书序号、发证机构名称、证书有效期、密码算法标识、公钥和私钥信息和其他属性信息等。利用数字证书，配合相应的安全代理软件，可以在网络上信息交付过程中检验对方的身份真伪，实现信息交付双方的身份真伪，并保证交付信息的真实性、完整性、机密性和不可否认性。数字证书提供了 PKI 的基础。

(2) 认证中心(Certification Authority，CA)：PKI 的核心，是公正、权威、可信的第三方网上认证机构，负责数字证书的签发、撤销和生命周期的管理，还提供密钥管理和证书在线查询等服务。

(3) 数字证书注册机构(Registration Authority，RA)：RA 系统是 CA 的数字证书发放和管理的延伸。它负责数字证书申请者的信息录入、审核以及数字证书发放等工作；同时，对发放的数字证书实行相应的管理功能。发放的数字证书可以存放于 IC 卡、硬盘或软盘等介质中。RA 系统是整个 CA 得以正常运营不可缺少的一部分。

(4) 数字签名：利用发信者的私钥和可靠的密码算法对待发信息或其电子摘要进行加密处理，这个过程和结果就是数字签名。收信者可以用发信者的公钥对收到的信息进行解密，从而辨别真伪。经过数字签名后的信息具有真实性和不可否认(抵赖)性。

(5) 密钥和证书管理工具：管理和审计数字证书的工具，认证中心使用它来管理在一个 CA 上的证书。

(6) 双证书体系：PKI 采用双证书体系，非对称算法支持 RSA 算法和 ECC 算法，对称密码算法支持国家密码管理委员会指定的算法。

(7) PKI 的体系架构：宏观来看，PKI 概括为两大部分，即信任服务体系和密钥管理中心。

PKI 信任服务体系是为整个业务应用系统提供基于 PKI 数字证书认证机制的实体身份鉴别服务，它包括认证机构、注册机构、证书库、证书撤销和交叉认证等。PKI 密钥管理中心(Key Management Center，KMC)提供密钥管理服务，为授权管理部门提供应急情况下的特殊密钥恢复功能，包括密钥管理机构、密钥备份和恢复、密钥更新和密钥历史档案等。

### 3.9.2 双证书和双密钥机制

一对密钥(一张证书)应用中的问题如下。

(1) 若密钥不备份，当密钥损坏(或管理密钥的人员离职时带走密钥)时，则以前加密的信息不可解密。

(2) 若密钥不备份，则很难实现信息审计。

(3) 若密钥不备份，则数字签名的不可否认性很难保证。

两对密钥(两张证书)的客观需求：一对密钥用于签名(签名密钥对)，另一对密钥用于加密(加密密钥对)。加密密钥在密钥管理中心生成及备份，签名密钥由用户自行生成并保存。

双密钥证书的生成过程如下。

(1) 用户使用客户端产生签名密钥对。

(2) 用户的签名私钥保存在客户端。

(3) 用户将签名密钥对的公钥传送给 CA。

(4) CA 为用户的公钥签名，产生签名证书。

(5) CA 将签名证书传回客户端进行保存。

(6) KMC 为用户生成加密密钥对。

(7) 在 KMC 中备份加密密钥以备以后进行密钥恢复。

(8) CA 为加密密钥对生成加密证书。

(9) CA 将用户的加密私钥和加密证书打包成标准格式 PKCS#12。

(10) 将打包后的文件传回客户端。

(11) 用户的客户端装入加密公钥证书和加密私钥。

### 3.9.3 X.509 证书标准

在 PKI/CA 架构中，一个重要的标准就是 X.509 标准，数字证书就是按照 X.509 标准制作的。本质上，数字证书是把一个密钥对(明确的是公钥，而暗含的是私钥)绑定到一个身份上的被签署的数据结构。整个证书有可信赖的第三方签名。典型的第三方即大型用户群体(如政府机关或金融机构)所信赖的 CA。

此外，X.509 标准还提供了一种标准格式 CRL。

目前 X.509 有不同的版本，X.509V2 和 X.509V3 都是目前比较新的版本，都是在

X.509V1 版本的基础上进行功能的扩充。每一版本必须包含下列信息。

(1) 版本号：用来区分 X.509 的不同版本号。

(2) 序列号：由 CA 给每一个证书分配唯一的数字型编号，当证书被取消时，实际上是将此证书的序列号放入由 CA 签发的 CRL 中，这也是序列号唯一的原因。

(3) 签名算法标识符：用来指定用 CA 签发证书时所使用的签名算法。算法标识符用来指定 CA 签发证书时所使用的公开密钥算法和 Hash 算法，需向国际著名标准组织(如 ISO)注册。

(4) 认证机构：发出该证书的机构唯一的 CA 的 X.500 规范用名。

(5) 有效期限：证书有效的时间，包括证书生效期和证书失效期，在所指定的这两个时间之间有效。

(6) 主题信息：证书持有人的姓名、服务处所等信息。

(7) 认证机构的数字签名：确保证书在发放之后没有被更改。

(8) 公钥信息：包括被证明有效的公钥值和加上使用这个公钥的方法名称。

X.509V3 在 X.509V2 的基础上进行了扩展。X.509V3 引进一种机制，这种机制允许通过标准化和类的方式将证书进行扩展，以包含额外的信息，从而适应下面的一些要求。一个证书主体可以有多个证书。证书主体可以被多个组织或社团的其他用户识别；可按特定的应用名(不是 X.500 规范用名)识别用户，如将公钥同 E-mail 地址联系起来。在不同证书政策和实用下会发放不同的证书，这就要求公钥用户要信赖证书。

PKI/CA 对数字证书的管理是按照数字证书的生命周期实施的，包括证书的安全需求确定、证书申请、证书登记、分发、审计、撤回和更新。映射证书到用户的账户是使数字证书的拥有者安全使用制定的应用所必不可少的环节，也是 PKI/CA 对数字证书管理的重要内容。CA 是一个受信任的机构，为了当前和以后的事务处理，CA 给个人、计算机设备和组织机构颁发证书，以证实它们的身份，并为它们使用证书的一切行为提供信誉的担保。

数字证书是公开密钥体制的一种密钥管理媒介。它是一种权威性的电子文档，形同网络计算环境中的一种身份证，用于证明某一主体(如人、服务器等)的身份及其公开密钥的合法性。在使用公钥体制的网络环境中必须向公钥的使用者证明公钥的真实合法性。因此，在公钥体制环境中必须有一个可信的机构来对任何一个主体的公钥进行公证，证明主体的身份以及它与公钥的匹配关系。数字证书的主要内容如表 3-21 所示。

表 3-21　数字证书的主要内容

| 字　　段 | 定　　义 |
| --- | --- |
| 主题名称 | 唯一标识证书所有者的标识符 |
| 签证机关名称(CA) | 唯一标识证书签发者的标识符 |
| 主体的公开密钥 | 证书所有者的公开密钥 |
| CA 的数字签名 | CA 对证书的数字签名，保证证书的权威性 |
| 有效期 | 证书在该期间内有效 |
| 序列号 | CA 产生的唯一性数字，用于证书管理 |
| 用途 | 主体公钥的用途 |

## 3.10 权限管理基础设施

权限管理基础设施或授权管理基础设施(Privilege Management Infrastructure,PMI)的核心思想是以资源管理为核心,将对资源的访问控制权交由授权机构进行管理,即由资源的所有者来进行访问控制管理。只有 PKI 无法对信息系统的资源进行合理有效的管理。PMI 几乎完全按照 PKI 的体系架构建立,外形很相像,内容却完全不同。PMI 建立在 PKI 基础上,以向用户和应用程序提供权限管理和授权服务为目标,主要向业务应用信息系统提供授权服务管理;提供用户身份到应用授权的映射功能,实现与实际应用处理模式相对应的、与具体应用系统开发和管理无关的访问控制机制;能极大地简化应用中访问控制和权限管理系统的开发与维护,减少管理成本和复杂性。

### 3.10.1 PMI 与 PKI 的区别

PMI 主要进行授权管理,证明这个用户有什么权限,能干什么。PKI 主要进行身份鉴别,证明用户身份。它们之间的关系如同签证和护照的关系。签证具有属性类别,持有某一类别的签证才能在该国家进行某一类别的活动。护照是身份证明,唯一标识个人信息,只有持有护照才能证明这个人是合法的。PMI 与 PKI 的比较如表 3-22 所示。

表 3-22 PMI 与 PKI 的比较

| 概　念 | PMI 实体 | PKI 实体 |
|---|---|---|
| 证书 | 属性证书 | 公钥证书 |
| 证书签发者 | 属性证书管理中心 | 认证证书管理中心 |
| 证书用户 | 持有者 | 主体 |
| 证书绑定 | 持有者名和权限绑定 | 主体名和公钥绑定 |
| 撤销 | 属性证书撤销列表(ACRL) | 证书撤销列表(CRL) |
| 信任的根 | 权威源(SOA) | 根 CA/信任锚 |
| 从属权威 | 属性权威(AA) | 子 CA |

### 3.10.2 属性证书及其管理中心

属性证书(Attribute Certificate,AC)表示证书的持有者(主体)对于一个资源实体(客体)所具有的权限。它是由一个做了数字签名的数据结构来提供的,这种数据结构称为属性证书,由属性权威签发并管理。

公钥证书是对用户名称和他/她的公钥进行绑定,而属性证书是将用户名称与一个或更多的权限属性进行绑定。从这方面而言,公钥证书可看为特殊的属性证书。

数字签名公钥证书的机构称为认证中心,签名属性证书的机构称为属性权威。PKI 信任源有时被称为根 CA,而 PMI 信任源被称为 SOA。CA 可以有它们信任的次级 CA。次级 CA 可以代理鉴别和认证。同样,SOA 可以将它们的权利授给次级 AA。若用户需要废除他/她的签字密钥,则 CA 将签发一个证书撤销列表。与之类似,若用户需要废除授权,AA 将签发一个属性证书撤销列表。

属性证书的使用有两种模式：一是推模式，当用户在要求访问资源时，由用户自己直接提供其属性证书，即用户将自己的属性证书“推”给资源服务管理器。这意味着，在客户和服务器之间不需要建立新的连接，而且对于服务器来说，这种方式不会带来查找证书的负担，从而减少了开销。二是拉模式，是业务应用授权机构发布属性证书到目录服务系统，当用户需要用到属性证书的时候，由服务器从属性证书发放者（属性权威）或存储证书的目录服务系统“拉”回属性证书。这种“拉”模式的主要优点是实现这种模式不需要对客户端以及客户－服务器协议做任何改动。这两种模式可以根据应用服务的具体情况灵活应用。

## 3.11 密码安全性分析

密码学的基本目的就是保障不安全信道上的通信安全。密码学领域存在一个很重要的事实：“如果许多聪明人都不能解决的问题，那么它可能不会很快得到解决。”这暗示很多加密算法的安全性并没有在理论上得到严格证明，只是这种算法思想推出后，经过许多人许多年的攻击并没有发现其弱点，没有找到攻击它的有效方法，从而认为它是安全的。

### 3.11.1 设计原则

（1）计算安全性（Computational Security）：指一种密码系统最有效的攻击算法至少是指数时间的，又称实际保密性（Practical Secrecy）。密码学更关心在计算上不可破译的密码系统。破译密码的代价超出密文信息的价值或者破译密码的时间超出密文信息的有效生命期，那么认为这个密码体制在计算上是安全的。

（2）可证明安全性（Provable Security）：若密码体制的安全性可以归结为某个数学困难问题，则称其是可证明安全的。可证明安全性只是说明密码体制的安全与一个问题是相关的，并没有证明密码体制是安全的，可证明安全性也被称为归约安全性。

（3）无条件安全性（Unconditional Security）或者完善保密性（Perfect Secrecy）：假设存在一个具有无限计算能力的攻击者，若密码体制无法被这样的攻击者攻破，则称其为无条件安全。无论有多少可使用的密文，都不足以唯一地确定密文所对应的明文。

（4）密码算法安全强度高：就是说攻击者根据截获的密文或某些已知明文密文对，要确定密钥或者任意明文在计算上不可行。

（5）柯克霍夫原则：密码体制的安全性不应依赖加密算法的保密性，而应取决于可随时改变的密钥。

（6）密钥空间应足够大：使试图通过穷举密钥空间进行搜索的方式在计算上不可行。

（7）既易于实现又便于使用：主要是指加密函数和解密函数都可以高效地计算。

### 3.11.2 密码攻击与分析

1. 密码攻击

（1）穷举攻击：密码分析者通过试遍所有的密钥来进行破译。穷举攻击又称为蛮力

攻击，是指攻击者依次尝试所有可能的密钥对所截获的密文进行解密，直至得到正确的明文。

（2）统计分析攻击：密码分析者通过分析密文和明文的统计规律来破译密码。抵抗统计分析攻击的方式是在密文中消除明文的统计特性。

（3）数学分析攻击：密码分析者针对加密算法的数学特征和密码学特征，通过数学求解的方法来设法找到相应的解密变换。为对抗这种攻击，应该选用具有坚实的数学基础和足够复杂的加密算法。

2. 密码分析

密码攻击和解密的相似之处在于都是设法将密文还原成明文的过程。根据密码分析者可获取的信息量不同，密码分析（也称为破译）有下列几种基本方法。

（1）唯密文攻击（Ciphertext Only Attack）：已知加密方法、明文语言和可能内容，从密文求出密钥或明文。

（2）已知明文攻击（Know-plaintext Attack）：已知加密方法和部分明文密文对，从密文求出密钥或明文。

（3）选择明文攻击（Chose-plaintext Attack）：已知加密方法，而且破译者可以把任意（或相当数量）的明文加密为密文，求密钥。这对于保护机密性的密码算法来说是最强有力的分析方法。

（4）选择密文攻击（Chose-ciphertext Attack）：已知加密方法，而且破译者可以把任意（或相当数量）的密文脱密为明文，求密钥。这对于保护完整性的密码算法来说是最强有力的分析方法。

（5）自适应选择明文攻击（Adaptive Chosen Plaintext Attack）：选择明文攻击的一种特殊情况，指密码分析者不仅能够选择要加密的明文，而且能够根据加密的结果对以前的选择进行修正。

（6）选择密钥攻击（Chosen Key Attack）：这种攻击情况在实际中比较少见，它仅表示密码分析者知道不同密钥之间的关系，并不表示密码分析者能够选择密钥。

## 3.12 密码系统管理

密码系统的安全性依赖密码管理。密码管理主要分为密钥管理、密码管理政策和密码测评。

### 3.12.1 密钥管理

密钥管理主要围绕密钥的生命周期进行，具体包括以下内容。

（1）密钥生成。密钥应由密码相关产品或工具按照一定标准产生，通常包括密码算法选择、密钥长度等。密钥生成时要同步记录密钥的关联信息，如拥有者、密钥使用起始时间、密钥使用终止时间等。

（2）密钥存储。一般来说密钥不以明文方式存储保管，应采取严格的安全防护措施，防止密钥被非授权地访问或篡改。

(3) 密钥分发。密钥分发是指通过安全通道把密钥安全地传递给相关接收方,防止密钥遭受截取、篡改、假冒等攻击,保证密钥机密性、完整性以及分发方、接收方身份的真实性。目前,密钥分发主要有人工方式、自动化方式和半自动化方式。其中,自动化方式主要通过密钥交换协议进行。

(4) 密钥使用。密钥要根据不同的用途(加密、签名 VNAC 等)来选择。密钥使用和密码产品保持一致性,密码算法、密钥长度、密码产品都要符合相关管理政策,即安全合规。使用密钥前,要验证密钥的有效性,如公钥证书是否有效。密钥使用过程中要防止密钥的泄露和替换,按照密钥安全策略及时更换密钥。建立密钥应急响应处理机制,以应对突发事件,如密钥丢失事件、密钥泄密事件、密钥算法缺陷公布等。

(5) 密钥更新。当密钥超过使用期限、密钥信息泄露、密码算法存在安全缺陷等情况发生时,相关密钥应根据相应的安全策略进行更新操作,以保障密码系统的有效性。

(6) 密钥撤销。当密钥到期、密钥长度增强或密码安全应急事件出现的时候,需要进行撤销密钥,更换密码系统参数。撤销后的密钥一般不重复使用,以免密码系统的安全性受到损害。

(7) 密钥备份。密钥备份应按照密钥安全策略,采用安全可靠的密钥备份机制对密钥进行备份。备份的密钥与密钥存储要求一致,其安全措施要求保障备份的密钥的机密性、完整性、可用性。

(8) 密钥恢复。密钥恢复是在密钥丢失或损毁的情形下,通过密钥备份机制,能够恢复密码系统的正常运行。

(9) 密钥销毁。根据密钥管理策略可以对密钥进行销毁。一般来说,销毁过程应不可逆,无法从销毁结果中恢复原密钥。特殊情况下,密钥管理支持用户密钥恢复和司法密钥恢复。

(10) 密钥审计。密钥审计是对密钥生命周期的相关活动进行记录,以确保密钥安全合规,违规情况可查可追溯。

### 3.12.2 密码管理政策

密码管理政策是指国家对密码进行管理的有关法律政策文件、标准规范、安全质量测评等。目前,我国已经发布《商用密码管理条例》,主要内容有商用密码的科研生产管理、销售管理、使用管理、安全保密管理。《中华人民共和国密码法》也已颁布实施,相关工作正在推进,明确规定,密码分为核心密码、普通密码和商用密码,实行分类管理。核心密码、普通密码用于保护国家秘密信息,属于国家秘密,由密码管理部门依法实行严格统一管理。商用密码用于保护不属于国家秘密的信息,公民、法人和其他组织均可依法使用商用密码保护网络与信息安全。

为规范商用密码产品的设计、实现和应用,国家密码管理局发布了一系列密码行业标准,主要有《电子政务电子认证服务管理办法》《电子政务电子认证服务业务规则规范》《密码模块安全检测要求》《安全数据库产品密码检测准则》《安全隔离与信息交换产品密码检测指南》《安全操作系统产品密码检测准则》《防火墙产品密码检测准则》等。

### 3.12.3 密码测评

数字时代呼唤安全创新，密码是国之重器，是数字技术发展的安全基因，是保障网络与数据安全的核心技术，也是推动我国数字经济高质量发展、构建网络强国的基础支撑。从实战需求看，日趋严峻的网络与数据安全威胁使得数字经济迫切需要密码技术抵御外部黑客攻击、防止内部人员泄露。从合规需求看，以密码应用安全性评估为抓手落实《中华人民共和国密码法》，并结合《中华人民共和国网络安全法》《中华人民共和国数据安全法》《中华人民共和国个人信息保护法》等法律法规，也在持续拉动密码应用新需求。

“商用密码应用安全性评估”是指在采用商用密码技术、产品和服务集成建设的网络和信息系统中，对其密码应用的合规性、正确性和有效性进行评估。开展商用密码应用安全性评估工作是国家法律法规的强制要求，是网络安全运营者的法定责任和义务。同时，开展商用密码应用安全性评估是商用密码应用正确、合规、有效的重要保证，是检验网络和信息系统安全性的重要手段。

1. 商用密码应用安全性评估是商用密码应用的重要推动力

商用密码应用的正确、合规、有效，是网络和信息系统安全的关键所在；而商用密码应用安全性评估工作的开展可以促进商用密码应用做到合规、正确和有效，是商用密码应用正确、合规、有效的重要推动力。

2. 商用密码应用安全性评估是应对网络与数据安全形势的需要

通过商用密码应用安全性评估可以及时发现在密码应用过程中存在的问题，为网络和信息安全提供科学的评价方法，逐步规范密码的使用和管理，从根本上改变密码应用不广泛、不规范、不安全的现状，确保密码在网络和信息系统中得到有效应用，切实构建起坚实可靠的网络安全密码保障。

3. 商用密码应用安全性评估是系统安全维护的必然要求

密码应用是否合规、正确和有效涉及密码算法、协议、产品、技术体系、密钥管理、密码应用多个方面。因此，需委托专业机构和专业人员，采用专业工具和专业手段，对系统整体的密码应用安全进行专项测试和综合评估，形成科学准确的评估结果，以便及时掌握密码安全现状，采取必要的技术和管理措施。

依据 GB/T 39786—2021《信息安全技术 信息系统密码应用基本要求》编制的《信息系统密码应用测评要求》将信息系统密码应用测评要求分为通用测评要求和密码应用测评要求。其中：通用测评要求对“密码算法和密码技术合规性”和“密钥管理安全性”提出测评要求，适用于第一级到第五级的信息系统密码应用测评；密码应用测评要求对信息系统的物理和环境安全、网络和通信安全、设备和计算安全、应用和数据安全四个技术层面提出了第一级到第四级密码应用技术的测评要求，并对管理制度、人员管理、建设运行和应急处置四个方面提出了第一级到第四级密码应用管理的测评要求。通用测评要求的内容不单独实施测评，也不单独体现在密码应用安全性评估报告的单元测评结果和整体测评结果中，仅供密码应用测评要求的测评实施引用。

目前，我国设立了商用密码检测中心，其主要职责包括：商用密码产品密码检测；信息安全产品认证密码检测；含有密码技术的产品密码检测；信息安全等级保护商用密码测评；商用密码行政执法密码鉴定；国家电子认证根 CA 建设和运行维护；密码技术服务；商用密码检测标准规范制定；等等。表 3-23 给出了典型密码功能测评技术，供密码测评人员在对信息系统中具体使用的密码产品或应用的密码功能进行测评实施时参考。

表 3-23　典型密码功能测评技术

| 密码功能 | 测评实施 | 预期结果 |
| --- | --- | --- |
| 传输机密性 | ① 利用协议分析工具，分析传输的重要数据或鉴别信息是否为密文，数据格式（如分组长度等）是否符合预期。<br>② 若信息系统以外接密码产品的形式实现传输机密性，如 VPN、密码机等，则可参考对这些密码产品应用的测评方法 | ① 传输的重要数据和鉴别信息均为密文，数据格式（如分组长度等）符合预期。<br>② 实现传输机密性的外接密码产品符合相应密码产品应用的要求 |
| 存储机密性 | ① 通过读取存储的重要数据，判断存储的数据是否为密文，数据格式是否符合预期。<br>② 若信息系统以外接密码产品的形式实现存储机密性，如密码机、加密存储系统、安全数据库等，则可参考对这些密码产品应用的测评方法 | ① 存储的重要数据均为密文，数据格式符合预期。<br>② 实现存储机密性的外接密码产品符合相应密码产品应用的要求 |
| 传输完整性 | ① 利用协议分析工具，分析受完整性保护的数据在传输时的数据格式（如签名长度、MAC 长度）是否符合预期。<br>② 若使用数字签名技术进行完整性保护，则商用密码应用安全性评估人员可以使用公钥对抓取的签名结果进行验证。<br>③ 如果信息系统以外接密码产品的形式实现传输完整性，如 VPN、密码机等，则可参考对这些密码产品应用的测评方法 | ① 受完整性保护的数据在传输时的数据格式（如签名长度、MAC 长度）符合预期。<br>② 使用签名技术进行完整性保护的，使用公钥对抓取的签名结果验证通过。<br>③ 实现传输完整性的外接密码产品符合相应密码产品应用的要求 |
| 存储完整性 | ① 通过读取存储的重要数据，判断受完整性保护的数据在存储时的数据格式（如签名长度、MAC 长度）是否符合预期。<br>② 若使用数字签名技术进行完整性保护，则商用密码应用安全性评估人员可使用公钥对存储的签名结果进行验证。<br>③ 条件允许的情况下，商用密码应用安全性评估人员可尝试对存储数据进行篡改（如修改 MAC 或数字签名），验证完整性保护措施的有效性。<br>④ 若信息系统以外接密码产品的形式实现存储完整性保护，如密码机、智能密码钥匙，则可参考对这些密码产品应用的测评方法 | ① 受完整性保护的数据在存储时的数据格式（如签名长度、MAC 长度）符合预期。<br>② 使用签名技术进行完整性保护的，使用公钥对存储的签名结果验证通过。<br>③ 对存储数据进行篡改，完整性保护措施能够检测出存储数据的完整性受到破坏。<br>④ 实现存储完整性的外接密码产品符合相应密码产品应用的要求 |

续表

| 密码功能 | 测评实施 | 预期结果 |
| --- | --- | --- |
| 真实性 | ① 若信息系统以外接密码产品的形式实现对用户、设备的真实性鉴别，如 VPN、安全认证网关、智能密码钥匙、动态令牌等，则可参考对这些密码产品应用的测评方法。<br>② 对于不能复用密码产品检测结果的，还要查看实体鉴别协议是否符合 GB/T 15843—1999《信息技术 安全技术 实体鉴别》中的要求，特别是对于“挑战—响应”方式的鉴别协议，可以通过协议抓包分析，验证每次挑战值是否不同。<br>③ 对基于静态口令的鉴别过程，抓取鉴别过程的数据包，确认鉴别信息（如口令）未以明文形式传输；对采用数字签名的鉴别过程，抓取鉴别过程的挑战值和签名结果，使用对应公钥验证签名结果的有效性。<br>④ 若鉴别过程使用了数字证书，则可参考对证书认证系统应用的测评方法。若鉴别未使用证书，则商用密码应用安全性评估人员要验证公钥或（对称）密钥与实体的绑定方式是否可靠，实际部署过程是否安全 | ① 实现对用户、设备的真实性鉴别的外接密码产品符合相应密码产品应用的要求。<br>② 实体鉴别协议符合 GB/T 15843—1999 中的要求。<br>③ 静态口令的鉴别信息以非明文形式传输，对于使用数字签名进行鉴别，公钥验证签名结果通过，并且符合证书认证系统应用的相关要求。<br>④ 公钥和（对称）密钥与实体的绑定方式可靠，部署过程安全 |
| 不可否认性 | ① 若使用第三方电子认证服务，则应对密码服务进行核查；若信息系统中部署了证书认证系统，则可参考对证书认证系统应用的测评方法。<br>② 使用相应的公钥对作为不可否认性证据的签名结果进行验证。<br>③ 若使用电子签章系统，则参考对电子签章系统应用的测评方法 | ① 使用的第三方电子认证密码服务或系统中部署的证书认证系统符合相关要求。<br>② 使用相应公钥对不可否认性证据的签名结果的验证结果为通过。<br>③ 使用的电子签章系统符合电子签章系统应用的相关标准规范要求 |

## 3.13 本章小结

密码技术是现代信息安全的基础和核心技术，它不仅能够对信息加密，而且能完成信息的完整性验证、数字签名和身份认证等功能。本章着重介绍了常见的替代密码、置换密码、对称密码、非对称密码、PKI、密码管理和密码测评等技术和基础理论。

## 习　题

3-1　指出古典密码体制的不足。

3-2　指出一次一密的两个问题。

3-3　试述密码学的发展阶段及其主要特征。

3-4 什么是密码学？密码编码学和密码分析学区别是什么？

3-5 密码学的五元组是什么？它们分别有什么含义？

3-6 密码分析主要有哪些方式？各有何特点？

3-7 简述对称密码体制和非对称密码体制的优缺点。

3-8 试说明链路加密、节点对节点加密、端对端加密的应用场景。

3-9 试用 C 语言模拟实现 DES 算法的整个加密过程。