第3章

数据加密技术

CHAPTER 3

数据加密技术是为了提高信息系统及数据的安全性和保密性，防止秘密数据被外部破译所采用的主要技术手段之一，为计算机网络带来了安全系数极高的发展环境。通过对数据的加密，能够有针对性地对网络数据传输过程中存在的安全威胁制定科学合理的防护举措。同时，它也是信息安全领域的一项关键技术，能够有效解决用户在网络上的信息被窃取的问题。

3.1 数据加密基础

人们希望把重要信息通过某种变换,转换为秘密形式的信息。转换方法可以分为两大类:一类是隐写术,即隐蔽信息载体(信号)的存在,在古代常用;另一类是编码术,将载荷信息的信号进行各种变换,使它们不为非授权者所理解。在利用现代通信工具的条件下,隐写术受到很大限制,但编码术(保密编码)却因计算机被广泛使用,取得了飞速的发展。


微课视频

3.1.1 数据加密技术的基本概念

1. 密码学

密码学(cryptology)是密码编码学与密码分析学的总称。密码编码学(cryptography)是密码体制的设计学,即如何将一个信息(或称明文,plain text)经过加密钥匙(encryption key)及加密函数转换,变成无意义的密文(cipher text);而密码分析学(cryptanalysis)则是在未知密钥的情况下从密文推演出明文或密钥的技术。

2. 密钥

数据加密技术包括两个要素:密钥和加密算法。密钥是一种参数,它是在将明文转换为密文或将密文转换为明文的算法中输入的参数(字符串或比特串)。密钥分为加密密钥和解密密钥,根据加密密钥和解密密钥是否相同或可相互推导,分别称为对称密钥(单密钥)与非对称密钥(双密钥)。

3. 数据加密方法(算法)

任何一个加密系统都是由明文、密文、密钥和算法组成。加密算法指的是将明文与密钥结合,变成密文的过程(步骤)。发送方通过加密设备或加密算法,用加密密钥将数据加密后发送出去。接收方在收到密文后,用解密密钥将密文解密,恢复为明文。在传输过程中,即使密文被非法分子偷窃获取,得到的也只是无法识别的密文,从而起到数据保密的作用。传统的加密算法采用手工操作和机械操作的方式进行,而现代的加密算法则使用计算机进行运算。尽管它们具体使用的算法思想和采用的处理方法各不一样,但所依据的信息变换的基本原理是一致的。将明文加密成密文基本方法有易位法和置换法。实际采用的算法通常是这两种方法的组合运用。

(1) 易位法。按照一定的规则,重新安排明文中的比特或字符的顺序来形成密文,而字符本身保持不变。这种内容的变位可以是一维或多维的。例如,可以定义一个 $m \times n$ 的矩阵,将明文逐字符地按行填入这个矩阵,然后按列输出便得到密文,这时变量 m 和 n 就是密钥。该方法还可以通过改变输出规律来进行变形。

(2) 置换法。按照一定的规则,用一个字符去置换(替代)另一个字符来形成密文,如凯撒密码。这样各字母保持其原来的位置不变,但其本身改变了。替换的方式可以多种多样。

① 单表置换密码算法。它将明文中的字母用密文中的对应字母代替,其过程为明文字母表到密文字母表的一一映射。

② 同义置换密码算法。明文中每个字符都可有多种(固定的)置换可能性,例如将字母 A 替换成数字 5、13、25 或 56 中的任意一个,这样加密时的替换是随机不确定的,而解密时的替换是唯一确定的。

③ 多字母置换密码算法。它是单字母置换密码算法的扩展,也叫密本式或辞典式密码,可以将一组字符(包括文字、单词、语句或短文)分别用各自不同的其他文字群来代替。例如,将 ABA 替换成 RTQ,将 ABB 替换成 SLL 等。使用隐语传递消息也可看成多字母替换方法的一种应用。

④ 插入式密码算法。该算法将明文嵌入其他信息中加以隐蔽。例如,把明文以字母为单位按照一定的规则插入一段平淡无奇的文章中,接收者只需根据事先约定的规则将明文字母逐一找出即可。再如,将密文嵌入图像中显示,如嵌入 JPG 格式的图像等。在每个字节的最低位放密文的一个比特,由此引起的颜色变化是人眼所不可察觉的,这样一幅 1024×1024 灰度的图像中可藏下 64KB 的密文。

⑤ 连锁置换式密码算法。该算法又称自身密钥式密码算法。加密时先利用数据中前面的字母作为相邻后面字母的密钥进行加密,再把前面的密文字母作为后面相邻明文字母的密钥逐次进行加密。这种使数据内容彼此相关的算法已广泛地被作为现代基于计算机处理的加密算法,用来进行数据完整性的保护。

现代加密技术则采用十分复杂的算法,将易位法和置换法交替使用多次而形成乘积密码。

4. 密码体制

密码体制也叫密码系统,是指能完整地解决信息安全中的机密性、数据完整性、认证、身份识别、可控性及不可抵赖性等问题中的一个或几个的系统。对一个密码体制的正确描述,需要用数学方法清楚地描述其中的各种对象、参数、解决问题所使用的算法等。密码体制必须易于使用,特别是应当可以在计算机上使用。密码体制的安全性依赖于密钥的安全性,现代密码学不追求加密算法的保密性,而是追求加密算法的完备,即使攻击者在不知道密钥的情况下,没有办法从算法中找到突破口。

传统的密码体制采用移位法、代替法和代数方法来进行加密和解密的变换,可以采用一种或几种方法结合的方式作为数据变换的基本模式。

5. 数据加密模型

一般数据加密模型如图 3-1 所示。用户 A 向用户 B 发送明文 X 之前,首先利用加密算法和加密密钥进行加密过程(运算),生成密文 Y,然后通过网络传输,当用户 B 收到密文 Y 后,利用解密算法和解密密钥,进行解密过程(运算),最后解析出明文 X。

密文 Y 在使用网络传输的过程中,可能会被截获或篡改,但由于截取者不知道解密密钥,无法解析出明文 X,起到了数据加密的作用。

3.1.2　密码学发展历史

数据加密技术是一项非常古老的技术,早在公元前 2000 多年前,埃及人就开始使用特别的象形文字作为信息编码来保护他们的秘密文件;而始于公元前 17 世纪由克里特岛人

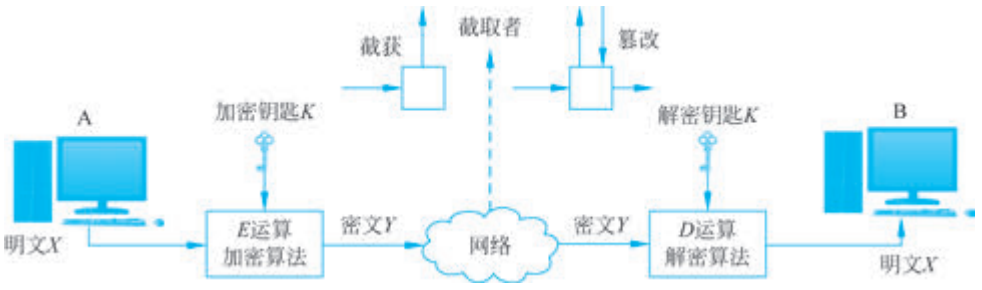

图 3-1 一般数据加密模型

发明的 Phaistos 圆盘更被誉为最难破解的密码之一。互联网因它的开放性,信息安全也难以实现面面俱到。如何保护这些数据的安全,已经成为每家企业不得不深思、考虑的问题。

数据加密技术发展至今可大致分为三个阶段。

(1) 1949 年以前。早期的数据加密技术简单,复杂程度不高,安全性较低,大部分是一些具有艺术特征的字谜。因此,这个时期的密码被称为古典密码。随着工业革命的到来和第二次世界大战的爆发,数据加密技术有了突破性的发展。通过密码算法或者机械的加密设备,明文可以转换为密文。

(2) 1949—1975 年。随着世界上第一台计算机的诞生,计算机技术发展突飞猛进,也使得加密技术从机械时代提升到了电子时代。这使得复杂计算的加密技术成为可能,也使得加密算法在复杂程度和安全性上得到了很大的提高。

(3) 1976 年至今。美国密码学专家迪菲和赫尔曼在 1976 年发表了《密码学新方向》一文,引发了密码学发展史上的一场革命。这篇论文首次证明了在发端和收端不需要传输密钥的保密通信的可能性,从而开创了公钥密码学的新纪元。从此以后,密码才开始充分发挥其商用价值和社会价值,普通大众能够接触到密码学并从中受益。

量子密码代表着密码学发展的一个新方向,但在最初的一段时间内,对其研究主要局限于理论探讨。目前,量子密码已经进入了实际应用阶段,引起了更多人的注意。

3.1.3 数据加密的安全性

衡量一个数据加密系统的安全性有两种基本的方法:一种是实际安全性;另一种是无条件安全性,又称理论安全性。

1. 实际安全性

实际安全性是根据破译加密系统所需的计算量来评价其安全性的。实际安全性又分为计算安全性和可证明安全性两种。

(1) 计算安全性。这种方法是指使用目前最好的方法攻破它所需要的计算远远超出攻击者的计算资源水平,则可以称这个密码体制在计算上是安全的。

(2) 可证明安全性。这种方法是将密码系统的安全性归结为某个经过深入研究的数学难题(如大整数素因子分解、计算离散对数等),数学难题被证明求解困难。这种评估方法存在的问题是它只说明了这个密码方法的安全性与某个困难问题相关,没有完全证明问题本身的安全性,并给出它们的等价性证明。

这两种安全性虽都是从计算量来考虑，但不尽相同，计算安全要算出或估计出破译它的计算量下限，而可证明安全则要从理论上证明破译它的计算量不低于解已知难题的计算量。人们说一个密码系统是“实际上安全的”，意指利用已有的最好的方法破译该系统所需要的努力超过了对手的破译能力（诸如时间、空间和资金等资源），或破译该系统的难度等价于解数学上的某个已知难题。当然，这只是提供了某种密码系统是计算上安全的一些证据，并没有真正证明系统是计算上安全的。

2. 理论安全性

理论安全性与对手的计算能力或时间无关，也就是说对手破译密码体制所做的任何努力都不会优于随机地选择来碰运气。说一个密码系统是“理论上安全的”，则具有无限计算资源（诸如时间、空间、设备和资金等）的密码分析者也无法破译该密码系统。

3.1.4　密码攻击

密码分析学的主要目的是在不知道密钥的情况下恢复出明文。成功的密码分析能恢复出密文所对应的明文或加密所使用的密钥，并且通过密码分析也可以发现密码系统的弱点，为达到最终掌握算法及密钥提供依据。

1. 攻击技术

对密码进行分析的尝试称为攻击。攻击技术主要有以下三种。

（1）穷举攻击。所谓穷举攻击就是密码分析者用遍所有密钥来破译密码。穷举攻击所花费的时间等于尝试次数乘以一次解密（加密）所需的时间。显然可以通过增大密钥量或加大解密（加密）算法的复杂度来对抗穷举攻击。现代密码体制往往经过了精心设计，这种攻击方法一般不会有什么效果。

（2）统计分析攻击。所谓统计分析攻击是指密码分析者通过分析明文与密文的统计规律来破译密码。统计分析攻击在历史上为破译密码做出过很大的贡献。对抗统计分析攻击的方法就是设法使明文的统计特性不带入密文，也就是明文的统计特性与密文的统计特性不一样。

（3）数学求解攻击。所谓数学求解攻击密码分析者针对加密算法的数学基础，通过数学求解的方法来破译密码。为对抗这种攻击，应选用具有坚实的数学基础和足够复杂的加密算法。

2. 攻击方法分类

根据密码分析者可利用的数据来分类，可将攻击方法分为以下几类。当然每一类都假设密码分析者知道所用的加密算法的全部知识。

1）唯密文攻击

唯密文攻击（ciphertext only attack，COA）是指仅仅知道密文的情况下进行分析，求解明文或密钥的密码分析方法。假定密码分析者拥有密码算法及明文统计特性，并截获了一个或者多个用同一密钥加密的密文，通过对这些密文进行分析求出明文或密钥。COA 已知条件最少，经不起唯密文攻击的密码是被认为不安全的。可以简单理解为只知道密文，推出

明文或密钥,一般用穷举攻击。

2) 已知明文攻击

已知明文攻击(known plaintext attack,KPA)是指攻击者掌握了部分的明文 M 和对应的密文 C,从而求解或破解出对应的密钥和加密算法。可以简单理解为知道部分的明文和密文对,推出密钥和加密算法。

3) 选择明文攻击

选择明文攻击(chosen plaintext attack,CPA)是指攻击者除了知道加密算法,还可以选定明文消息,从而得到加密后的密文,即知道选择的明文和加密的密文,但是不能直接攻破密钥。可以简单理解为知道明文就知道密文,目标为推出密钥。

4) 选择密文攻击

选择密文攻击(chosen ciphertext attack,CCA)是指攻击者可以选择密文进行解密,除了知道已知明文攻击,攻击者可以任意制造或选择一些密文,并得到解密的明文,是一种比已知明文攻击更强的攻击方式。若一个密码系统能抵抗 CCA,那必然能够抵抗 COA 和 KPA。密码分析者的目标是推出密钥,CCA 主要应用于分析公钥密钥体制。可以简单理解为知道密文就知道明文,目标为推出密钥。

当密码系统只有承受住 CPA 和 CCA,才能算是安全的。四种攻击方式对应的攻击强度为攻击难度:CCA>CPA>KPA>COA。


微课视频

3.1.5 数据加密基本方式

数据加密技术是网络安全最有效的技术之一。一个加密的数据传输网络不但可以防止非授权用户的搭线窃听和入网,而且也是对付恶意软件的有效方法之一。一般的数据加密可以在通信的三个层次实现:链路加密、端到端加密和节点加密。

1. 链路加密

对于在两个网络节点间的某一次通信链路,链路加密(又称在线加密)能为网上传输的数据提供安全保证。对于链路加密,所有消息在被传输之前进行加密,在每一个节点对接收到的消息进行解密,然后先使用下一个链路的密钥对消息进行加密,再进行传输。在到达目的地之前,一条消息可能要经过许多通信链路的传输。

由于在每一个中间传输节点消息均被解密后重新进行加密,因此,包括路由信息在内的链路上的所有数据均以密文形式出现。这样,链路加密就掩盖了被传输消息的源点与终点。填充技术的使用以及填充字符在不需要传输数据的情况下就可以进行加密,使得消息的频率和长度特性得以掩盖,从而可以防止对通信业务进行分析。

尽管链路加密在计算机网络环境中使用得相当普遍,但它并非没有问题。链路加密通常用在点对点的同步或异步线路上,它要求先对在链路两端的加密设备进行同步,然后使用一种链模式对链路上传输的数据进行加密。这就给网络的性能和可管理性带来了副作用。

在线路/信号经常不通的海外或卫星网络中,链路上的加密设备需要频繁地进行同步,带来的后果是数据丢失或重传。另外,即使仅一小部分数据需要进行加密,也会使得所有传输数据被加密。

在一个网络节点,链路加密仅在通信链路上提供安全性,消息以明文形式存在,因此所

有节点在物理上必须是安全的，否则就会泄露明文内容。然而，保证每一个节点的安全性需要较高的费用。为每一个节点提供加密硬件设备和一个安全的物理环境所需要的费用由保护节点物理安全的雇员开销、为确保安全策略和程序的正确执行而进行审计时的费用，以及为防止安全性被破坏时带来损失而参加保险的费用组成。

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，该密钥必须被秘密保存，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了一个问题，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，这就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很广，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。

2. 端到端加密

端到端加密允许数据在从源点到终点的传输过程中始终以密文形式存在。采用端到端加密（又称脱线加密或包加密），消息在被传输时到达终点之前不进行解密，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使消息泄露。

端到端加密系统的价格便宜些，并且与链路加密和节点加密相比更可靠，更容易设计、实现和维护。端到端加密还避免了其他加密系统所固有的同步问题，因为每个报文包均是独立被加密的，所以一个报文包所发生的传输错误不会影响后续的报文包。此外，从用户对安全需求的直觉上讲，端到端加密更自然些。单个用户可能会选用这种加密方法，以便不影响网络上的其他用户，此方法只需要源节点和目的节点是保密的即可。

端到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。由于这种加密方法不能掩盖被传输消息的源点与终点，因此，它对于防止攻击者分析通信业务是脆弱的。

3. 节点加密

在网络通信的节点处，采用一个与节点机相连的密码装置，密文在该装置中被解密并重新加密。尽管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是类似的。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，然后进行加密。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，这一过程是在节点上的一个安全模块中进行的。

节点加密要求报头和路由信息以明文形式传输，以便中间节点能得到如何处理消息的信息。因此，这种方法对于防止攻击者分析通信业务是脆弱的。

三种数据加密方式的原理如图 3-2 所示，图中 E_K 是加密运算，D_K 是解密运算。

3.1.6　数据加密的目的

数据加密的目的主要有两个：一是数据的保密性；二是数据的真实性。

1. 保密性

数据加密最初的目的是保护数据在存储状态下和在传输过程中不被窃取、解读和利用。

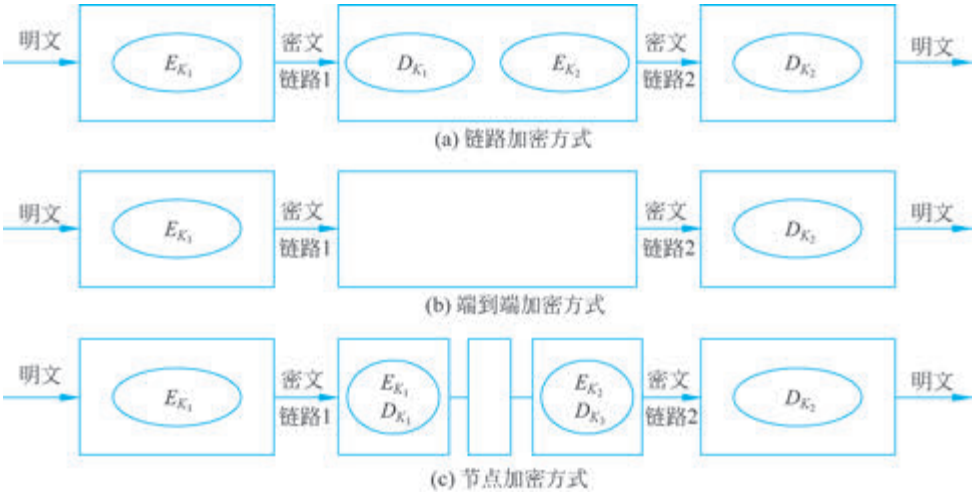

图 3-2 三种数据加密方式的原理

加密过的数据以密文的形式存在,即使被窃取,没有密钥也无法解读成原文;而合法用户收到密文后,可以用掌握的密钥解密后获得信息。

保密性能够做到密码分析员无法从截获的密文中求出明文,即使截获了一段密文 C,甚至知道了与它对应的明文 M,密码分析员要从中系统地求出解密变换仍然在计算上是不可能实现的,也就是说,密码分析员要从由截获的密文 C 系统地求出明文 M 在计算上是不可能的。要做到保密性必须满足两个要求:第一个要求保证不能系统地求出解密变换;第二个要求保证在不知道解密变换的情况下无法从密文中解出明文,无论被截获的密文消息的数量和长度是多少。为了实现保密性,这两个要求都必须成立。

保密性只要求对变换 D_K(解密密钥)加以保密,只要不影响 D_K 的保密,变换 E_K 可以公布于众。图 3-3(a)表示了这种情况。

2. 真实性

数据的真实性要求密码分析员无法用虚假的密文 C' 代替真实密文 C 而不被觉察。它包括两个要求:对于给定的 C,即使密码分析员知道对应于它的明文 M,要系统地求出加密变换 E_K 仍然在计算上是不可能实现的;密码分析员要系统地找到密文 C',使 $D_K(C')$ 是明文空间上有意义的明文,这在计算上是不可能实现的。

第一个要求保证不能系统地求出加密变换 E_K;第二个要求保证在不知道加密变换 E_K 的情况下不能找到虚假密文 C',使它在解密后变为有意义的明文。与保密性类似,为了实现真实性,无论被截获的密文数量多大,这两个要求都必须成立。真实性只要求变换 E_K(加密密钥)保密,变换 D_K 可公布于众。图 3-3(b)表示了这种情况。

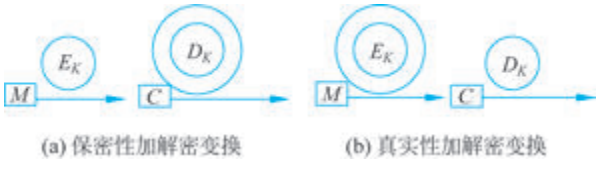

图 3-3 加解密变换

3.1.7　密码体制的分类

密码体制的分类方法有多种，常用的分类方法有以下三种。

1. 根据加密和解密算法所用的密钥是否相同分类

根据加密算法和解密算法所使用的密钥是否相同，可以将密码体制分为以下几种。

(1) 对称密码体制(symmetric cipher)，也称为单钥密码体制、秘密密钥密码体制、对称密钥密码体制或常规密码体制。常用的对称密码体制有 DES、3DES、AES、IDEA 等系列密码。

如果一个提供保密服务的密码系统，它的加密密钥和解密密钥相同，或者虽然不相同，但由其中的任意一个可以很容易地导出另外一个，那么该系统所采用的就是对称密码体制。对称密码体制的主要优势是加密、解密运算的处理速度快、效率高、算法安全性高。对称密码体制存在的局限性或不足：①对称密码算法的密钥分发过程复杂，所花代价高；②密钥管理量的困难；③保密通信系统的开放性差；④存在数字签名的困难性。

(2) 非对称密码体制(asymmetric cipher)，也称为双钥密码体制、公开密钥密码体制、非对称密钥密码体制。常用的非对称密码体制有 RSA、ElGamal、椭圆曲线密码等。

如果一个提供保密服务的密码系统，其加密算法和解密算法分别用两个不同的密钥实现，并且由加密密钥不能推导出解密密钥，则该系统所采用的就是非对称密码体制。采用非对称密钥密码体制的每个用户都有一对选定的密钥。其中一个是可以公开的，称为公开密钥(public key，简称公钥)；另一个由用户自己秘密保存，称为私有密钥(private key，简称私钥)。非对称密码体制的主要优势是：①密钥分配简单；②系统密钥量少，便于管理；③系统开放性好；④可以实现数字签名。非对称密码体制也存在局限性：①加密、解密运算效率较低，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较多；②由于加密密钥是公开发布的，客观上存在“可能报文攻击”的威胁。

2. 根据对明文信息的处理方式分类

根据对明文信息的处理方式分类，可以分为以下几类。

(1) 分组密码(block cipher)，将消息进行分组，一次处理一个数据块(分组)元素的输入，对每个输入块产生一个输出块。在用分组密码加密时，一个明文分组被当作一个整体来产生一个等长的密文分组输出。分组密码通常使用的分组大小是 64 比特或 128 比特。一般对称密码采用分组加密形式。常用的分组密码有 DES、3DES、AES、IDEA 等。

分组密码是在密钥的控制下一次变换一个明文分组的密码体制。分组密码体制的设计要求：①分组长度要足够大，以防止通过收集不同的密文明文对来进行穷举明文空间的攻击；如果分组长度只有 8 比特，则若收集到 256 个不同的密文明文对，就可破译每个分组。②密钥量应足够大，以防止穷举密钥空间的攻击。③密码算法应足够复杂，使破译者除了穷举法攻击之外，找不到其他简洁的数学破译方法。

(2) 序列密码(stream cipher)，也称为流密码，连续地处理输入元素，并随着处理过程的进行，一次产生一个元素的输出。在用序列密码加密时，一次加密一比特或一字节。常用的序列密码有 A5、SEAL 和 RC 系列密码等。

3. 根据是否能进行可逆的加密变换分类

根据是否能进行可逆的加密变换分类,可以将密码体制分为以下几类。

(1) 单向函数密码体制,是一类特殊的密码体制,其性质是可以很容易地把明文转换为密文,但再把密文转换为正确的明文却是不可行的,有时甚至是不可能的。单向函数只适用于某种特殊的、不需要解密的应用场合,如用户口令的存储和信息的完整性保护与鉴别等。常见的单向密码体制有 MD 系列密码和 SHA 系列密码等。

(2) 双向变换密码体制,指能够进行可逆的加密、解密变换。绝大多数加密算法都属于这一类,它要求所使用的密码算法能够进行可逆的双向加解密变换,否则接收者就无法把密文还原成明文。常见的双向密码体制包括对称密码、非对称密码、序列密码。如 DES、3DES、AES 等和 RSA、ElGamal、椭圆曲线密码以及 A5、SEAL 和 RC 系列密码等。

另外,关于密码体制的分类,还有一些其他方法,例如按照在加密过程中是否引入客观随机因素,可以分为确定型密码体制和概率密码体制等。

3.2 传统密码体制


微课视频

3.2.1 古典密码体制

存于石刻或史书中的记载表明,许多古代文明,包括埃及人、希伯来人、亚述人都在实践中逐步发明了密码系统。从某种意义上说,战争是科学技术进步的催化剂。人类自从有了战争,就有了通信安全的需求,密码技术源远流长。

古代加密方法大约起源于公元前 440 年出现在古希腊战争中的隐写术。当时为了安全传送军事情报,奴隶主剃光奴隶的头发,将情报写在奴隶的光头上,待头发长长后将奴隶送到另一个部落,再次剃光头发,原有的信息浮现出来,从而实现这两个部落之间的秘密通信。

公元前 400 年,斯巴达人就发明了“塞塔式密码”,即把长条纸螺旋形地斜绕在一个多棱棒上,将文字沿棒的水平方向从左到右书写,写一个字旋转一下,写完一行再另起一行从左到右写,直到写完。接下来,纸条上的文字消息杂乱无章、无法理解,这就是密文,但将它绕在另一个同等尺寸的棒子上后,就能看到原始的消息。这是最早的密码技术。较古典的密码体制有以下几种。

1. 凯撒密码

在密码学中,凯撒密码(Caesar cipher),或称凯撒加密、凯撒变换、变换加密,是一种最简单且最广为人知的加密技术。它是一种替换加密的技术,明文中的所有字母都在字母表上向后(或向前)按照一个固定数目进行偏移后被替换成密文。

凯撒密码源于古罗马时期凯撒大帝加解密军事情报产生的算法,它是通过移位的方式对明文信息中的每个英文字母进行加密。这里我们需要知道位移量,例如位移量是 3 时,所有的字母 A 将被替换成 D,B 变成 E,以此类推,那么原文为 hello 的字符串会被加密成密文 khoor,如图 3-4 所示。这个加密方法是以罗马共和时期凯撒的名字命名的,当年凯撒曾用此方法与其将军们进行联系。

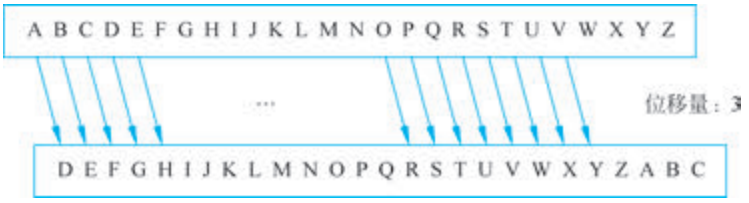

图 3-4　凯撒密码加密原理

2. 维吉尼亚密码

维吉尼亚密码是凯撒密码的改进。它对同一条信息中的不同字母用不同的密码进行加密。例如，如果使用关键词 BIG 加密，发件人将把信息按三个字母的顺序排列，则加密的信息每三个字母分别向后移动 1、8、6 位，具体算法如图 3-5 所示。

$$E_k(x_1, x_2, \cdots, x_n)=((x_1+k_1)\bmod 26, (x_2+k_2)\bmod 26, \cdots, (x_m+k_m)\bmod 26,$$
$$(x_{m+1}+k_1)\bmod 26, (x_{m+2}+k_2)\bmod 26, \cdots, (x_{2m}+k_m)\bmod 26,$$
$$\vdots$$
$$(x_{n-m+1}+k_1)\bmod 26, (x_{n-m+2}+k_2)\bmod 26, \cdots, (x_n+k_m)\bmod 26]$$

图 3-5　维吉尼亚密码加密算法

3. 波利比奥斯方阵密码

公元前 2 世纪，一个叫波利比奥斯(Polybius)的希腊人设计了一种将字母编码成符号对的方法，他使用了一个称为波利比奥斯的校验表，这个表中包含许多后来在加密系统中非常常见的成分。波利比奥斯校验表由一个 5 行 5 列的网格组成，网格中包含 26 个英文字母，其中 i 和 j 在同一格中。检验表(方阵)的行和列也可以用数字(可以不按序)来表示，在古代，这种棋盘(方阵)密码被广泛使用。波利比奥斯校验表如表 3-1 所示。

表 3-1　波利比奥斯校验表

	A(1)	B(2)	C(3)	D(4)	E(5)
A(1)	a	b	c	d	e
B(2)	f	g	h	i/j	k
C(3)	l	m	n	o	p
D(4)	q	r	s	t	u
E(5)	v	w	x	y	x

例如，单词 hello 加密后的字母或数字串为 BCAECACACD/2315313124。

4. 莫尔斯电码

电报最早是由美国的**莫尔斯**在 1844 年发明的，故也被叫作**莫尔斯电码**。它由两种基本信号和不同的间隔时间组成：短促的点信号“ . ”，读“的”(Di)；保持一定时间的长信号“—”，读“答”(Da)。间隔时间：滴，1t；答，3t；滴答间，1t；字母间，3t；字间，5t。其中，t 表示停顿的时间间隔。**莫尔斯**电码表如图 3-6 所示。

字符	电码符号	字符	电码符号	字符	电码符号
A	·—	N	—·	1	·————
B	—···	O	———	2	··———
C	—·—·	P	·——·	3	···——
D	—··	Q	——·—	4	····—
E	·	R	·—·	5	·····
F	··—·	S	···	6	—····
G	——·	T	—	7	——···
H	····	U	··—	8	———··
I	··	V	···—	9	————·
J	·———	W	·——	0	—————
K	—·—	X	—··—	?	··——··
L	·—··	Y	—·——	/	—··—·
M	——	Z	——··	()	—·——·—
				—	—····—
				·	·—·—·—

图 3-6　莫尔斯电码表

3.2.2　对称-分组密码体制基础

对称密码体制也称为传统密码体制、单钥或私钥密码体制，其中大部分是分组密码，本小节介绍对称-分组密码体制的基础知识。

分组密码是指将明文信息编码表示后的数字序列 $x_0,x_1,\cdots,x_i,\cdots$ 划分成长度为 n 的组 $\boldsymbol{x}=(x_0,x_1,\cdots,x_{n-1})$，各组(长度为 n 的向量)分别在密钥 $\boldsymbol{k}=(k_0,k_1,\cdots,k_{l-1})$ 的控制下，变换成输出序列 $\boldsymbol{y}(y_0,y_1,\cdots,y_{m-1})$(长度为 m 的向量)，其加密函数为 $E:V_n\times K\to V_m$，解密函数为 $D:V_m\times K\to V_n$，V_n 和 V_m 分别是 n 维和 m 维向量空间，K 为密钥空间，如图 3-7 所示。若 $m>n$，则为有数据扩展的分组密码；若 $m<n$，则为有数据压缩的分组密码；若 $m=n$，则分组密码对明文加密后既无数据扩展也无数据压缩。

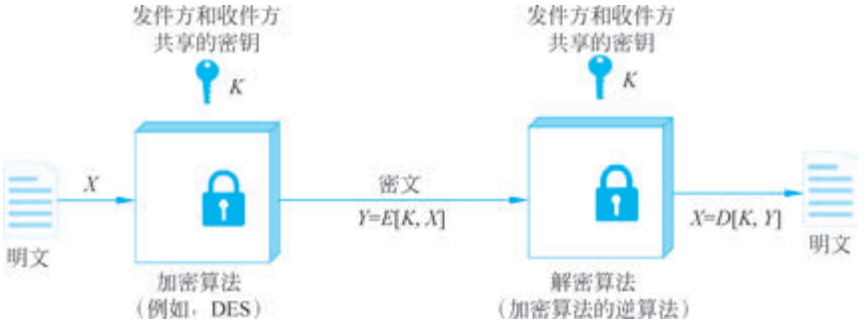

图 3-7　对称-分组密码框架

分组密码的特点是加密密钥与解密密钥相同。分组密码的安全性主要依赖于密钥的保密，而不是加密算法与解密算法的保密。较早的著名分组密码算法有 DES(数据加密标准)和 IDEA(国际数据加密算法)。2002 年 11 月，美国公布了旨在取代 DES 的 21 世纪的加密标准 AES(高级加密标准)算法。

1. 分组密码的设计原则

下面从安全性和实现两方面介绍分组密码的设计原则。

1）针对安全性的设计原则

影响分组密码安全性的因素很多，诸如分组长度和密钥长度等，但针对安全性的两个一般设计原则是香农(Shannon)提出的扩散和混淆原则，这两个原则的目的就是抵抗对密码的统计分析。如果攻击者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，攻击者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。在香农称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。

所谓扩散，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，也就是说，密文中每一位均受明文中多位影响。在分组密码中，可对数据重复执行某个置换，再对这一置换作用于一个函数，可获得扩散。

所谓混淆，就是使密文和密钥之间的统计关系变得尽可能复杂，使得攻击者即使获取了关于密文的一些统计特性，也无法推测密钥。使用复杂的代换算法可以得到预期的混淆效果。扩散和混淆成功地实现了分组密码的本质属性，因而成为设计现代分组密码的基础。

2）针对实现的设计原则

分组密码可以用软件和硬件来实现，硬件实现的特点是可获得高速率，软件实现的特点则是灵活性强、代价低。因此，分组密码的设计可根据预定的实现方法来考虑。

软件实现的设计原则：使用子块和简单的运算，密码运算在子块上进行，因此子块的长度自然要适应软件编程，如 8 比特、16 比特、32 比特、64 比特等。子块上所进行的密码运算应该是易于实现的运算，最好使用一些标准处理器所具有的一些基本指令，如加法、乘法、移位等。

硬件实现的设计原则：加密与解密可用同样的器件来实现，且尽量使用规则结构，因为密码应有一个标准的组件结构，以便能适应大规模集成电路的实现。

2. 常见分组密码的设计结构

1）Feistel 密码结构

Feistel 网络是分组密码算法设计的重要结构之一，其思想实际上是 Shannon 提出的利用乘积密码实现混淆与扩散思想的具体应用。Feistel 提出利用乘积密码可获得简单的代换密码。所谓乘积密码就是指顺序地执行两个或多个基本密码系统，使得最后结果的密码强度高于每个基本密码系统产生的结果。

加密算法的输入是数据分组 M 和密钥 K。将每组明文分为左右两部分，即 L 和 R。把 L 和密钥 K 经过变换 F 得到 $F_K(L)$后与 R 异或，将这个过程说成是将 L 变换后异或到 R 中，并记这个运算过程为 $M'=XR^K(M)$。记

$$M(E,R),\quad M'=XR^K(M)=(L',R') \tag{3-1}$$

那么

$$L'=L,\quad R'=R\oplus F_K(L) \tag{3-2}$$

其中，$\oplus$表示异或运算，图 3-8 表示 $XR^K(M)$的产生过程。

要从 M'反过来计算 M 应该怎么做呢？只需要将 L 经过相同变换后重新异或到 R 中就可以得到 M。用公式描述就是 $XR^K(XR^K(M))=M$，即 XR 的逆运算就是 XR。换句话说，两次 XR 运算互相抵消。这个过程很简单，感兴趣的读者可以自行推导，这里就不再证明了。

图 3-8 $XR^K(M)$的产生过程

同样,也可以定义 $XL^K(M)$运算,也就是将数据的右半部分经过变换异或到左半部分。Feistel 密码结构就是经过多次重复的 XR 和 XL 运算实施数据的混合。在 Feistel 结构中,每轮的结构都相同,每轮中右半部分数据作用于轮函数 f 后,再与左半部分数据进行异或运算;每轮的轮函数的结构都相同,但以不同的子密钥 K_i 作为参数。每次异或运算结束后,再交换左右两半部分数据。

在进行完 n 轮迭代后,左右两部分再合并到一起以产生密文分组。其中第 i 轮迭代的输入为前一轮的输出的函数:

$$L_i = R_{i-1} \tag{3-3}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i) \tag{3-4}$$

其中,K_i 是第 i 轮的子密钥,由加密密钥 K 得到。一般地,各轮子密钥彼此不同而且与 K 也不同。

Feistel 网络的实现与以下参数和特性有关。

(1) 组大小:分组越大安全性越高,但加密速度就越慢。

(2) 密钥大小:密钥越长安全性越高,但加密速度就越慢。

(3) 轮数:单轮结构不能保证安全性,但多轮结构可提供足够的安全性。

(4) 子密钥产生算法:算法的复杂性越大,密码分析的困难性就越大。

(5) 轮函数:轮函数的复杂性越大,密码分析的困难性也越大。

Feistel 解密过程本质上与加密过程是一样的,算法使用密文作为输入,但使用子密钥 K_i 的次序与加密过程相反,即第一轮使用 K_n,第二轮使用 K_{n-1},……,最后一轮使用 K_1。这一特性保证了解密与加密采用统一算法。

将 Feistel 网络的级数由两级扩张到多级,即为广义 Feistel 网络,如 SM4 等。

2) SP 网络

SP 网络也是一类分组密码编码的实用技术,如 AES 等。S(substitute)是指代替,其主要作用是混淆输入信息;P(permutation)是指置换,其主要作用是扩散输入信息。SP 网络中的混淆层一般由若干小次数布尔置换 S 盒并置而成,它们是非线性部分;扩散层由分位可逆线性变换而成,该变换隶属于大次数布尔置换群的可逆线性变换子群。在层密钥的作用下,混淆层与扩散层合成 SP 网络中的一次输入迭代(简称一轮),在一定次数的迭代后,形成大次数的关于输入的密码学伪随机布尔置换,即 SP 网络是一类密码学布尔置换生成器。

若 $s \in S_2^n$,则称 s 为 n 次的布尔置换。

一类 SP 网络的定义设 $X_{r+1} = (S(X_{r1} \oplus K_{rl}), \cdots, S(X_{ri} \oplus K_{rl}))A$,其中 $X_r(X_{r1}, X_{r2}, \cdots, X_{ri})(r \geqslant 1)$表示第 r 轮输入;X_{r+1} 表示第 r 轮的输出或第 $r+1$ 轮的输入;$K_{rl} \in F^{w_2}$,$l = 1, 2, \cdots, i$,表示第 r 轮密钥;i 为每轮并置的小次数布尔置换的个数;S 为 w 次的

非线性布尔置换，简记 $X_{r+1}=SP_{Kr}(X_r)$。

3. 分组密码工作模式

1）ECB

ECB(electronic code book，电子编码本)模式是最简单的加密模式，明文消息被分成固定大小的块(分组)，并且每个块被单独加密。每个块的加密和解密都是独立的，且使用相同的方法进行加密，所以可以进行并行计算，但是这种方法一旦有一个块被破解，使用相同的方法可以解密所有的明文数据，安全性比较差。ECB 模式适用于数据较少的情形，加密前需要把明文数据填充到块大小的整倍数。分组密码的工作模式如图 3-9 所示。

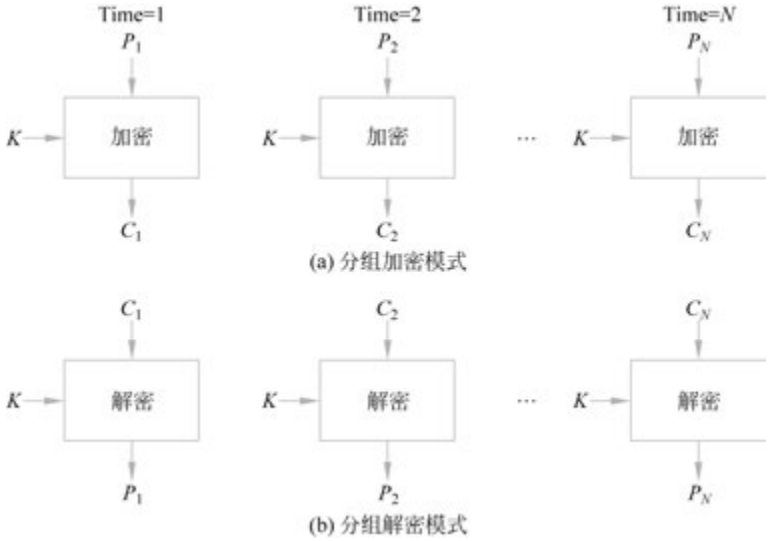

图 3-9　分组密码的工作模式

ECB 算法优点：简单、孤立，每个块单独运算。ECB 算法适合并行运算，传输错误一般只影响当前块。

ECB 算法缺点：同明文输出同密文，可能导致明文攻击。

2）CBC

CBC(cipher block chaining，密码分组链接)模式中每一个分组要先和前一个分组加密后的数据进行 XOR(异或)操作，然后再进行加密。这样每个密文块依赖该块之前的所有明文块，为了保持每条消息都具有唯一性，第一个数据块进行加密之前需要用初始化向量 **IV** 进行异或操作。CBC 模式是一种最常用的加密模式，它主要缺点是加密是连续的，不能并行处理，并且与 ECB 一样消息块必须填充到块大小的整倍数。密码分组链接的工作模式如图 3-10 所示。

CBC 算法优点：串行化运算，相同明文不同密文。

CBC 算法缺点：需要初始向量。

3）CFB

CFB(cipher feedback，密码反馈)模式和 CBC 模式比较相似，前一个分组的密文加密后和当前分组的明文异或操作生成当前分组的密文。CFB 模式的解密和 CBC 模式的加密在

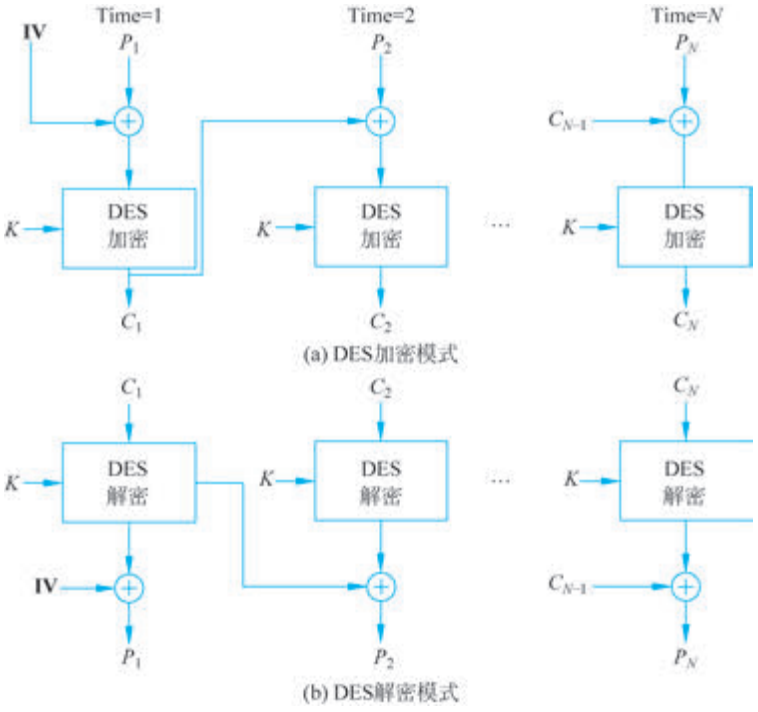

图 3-10 密码分组链接的工作模式

流程上其实是非常相似的。密码反馈的工作模式如图 3-11 所示。

CFB 算法优点：同明文不同密文，分组密钥转换为流密码。

CFB 算法缺点：串行运算不利并行，传输错误可能导致后续传输块错误。

4) OFB

OFB(output feedback，输出反馈)模式将分组密码转换为同步流密码，也就是说可以根据明文长度先独立生成相应长度的流密码。通过流程图可以看出，OFB 和 CFB 非常相似，CFB 是前一个分组的密文加密后 XOR 当前分组明文，OFB 是前一个分组与前一个明文块异或之前的流密码 XOR 当前分组明文。由于异或操作的对称性，因此 OFB 模式的解密和加密是完全一样的流程。

OFB 算法优点：同明文不同密文，分组密钥转换为流密码。

OFB 算法缺点：串行运算不利并行，传输错误可能导致后续传输块错误。

3.2.3 数据加密标准

1. 背景概述

20 世纪 70 年代以前，美国对所有的加密算法研究进行保密，因此没有满足商业用途的标准民用加密算法。1972 年，美国 NBS(现在的 NIST)发起了一个计算机及通信安全的研究项目。于是 IBM 公司的 W. Tuchman 和 C. Meyers 等于 1974 年提出了一个数据加密算法 Lucifer，它满足可测试和验证、易于实现和使用、可标准化和具有互操作性等特点。在经过 NSA 对其进行安全认证之后，1975 年 3 月 17 日，NBS 公布该算法，并重新命名为

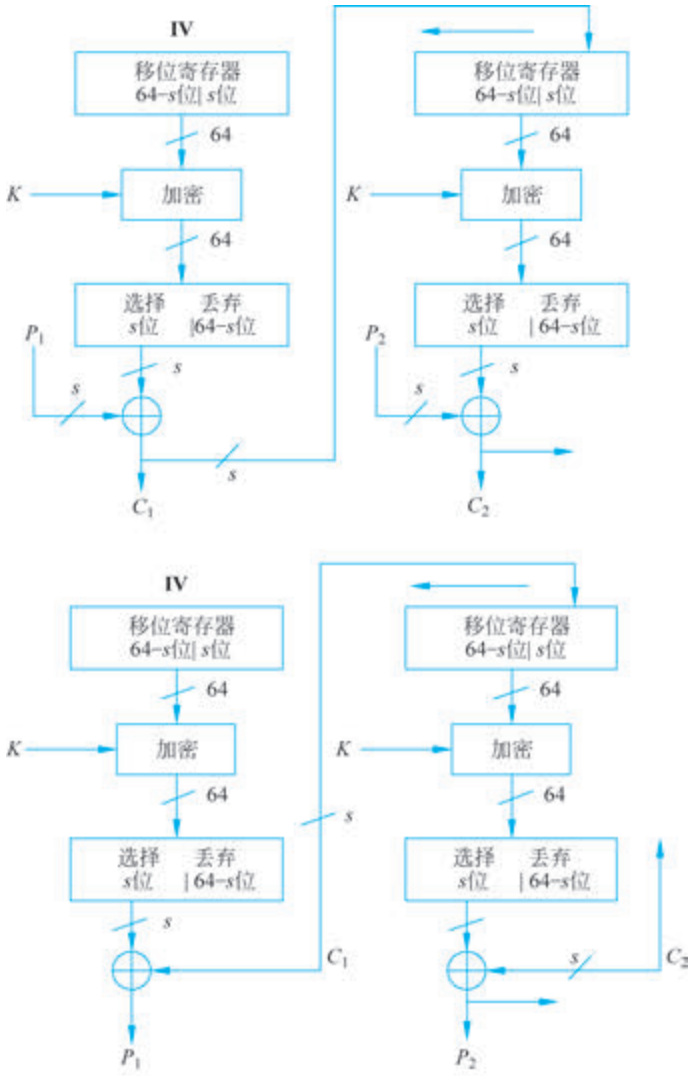

图 3-11　密码反馈的工作模式

DES(data encryption standard),同时 IBM 公司发布了同意免费使用该算法的声明。1976 年 11 月 23 日,DES 被美国国家标准局正式采纳作为商业和政府非要害信息的加密标准。DES 于 1981 年成为 ANSI 标准,并被称为 DEA,之后又有了一些面向如零售业、金融业等特定应用的变形。

按照规定,NBS 和 NSA 每隔 5 年要对 DES 进行一次重新审查和修订。第一次为 1983 年,没有进行任何修改。1987 年,NSA 认为 DES 已不够安全(尽管没有拿出证据),宣布不再对其进行认证,即不再为其安全性负责,但 NBS 仍然宣布其作为国家标准再过渡 5 年。1993 年,由于 NSA 没有提出新的民用加密算法.而且也没有 DES 被攻破的公开声明,因此它作为美国国家标准继续过渡 5 年,1998 年的情况依然如此。

DES 的密钥空间可用的密钥为 256 个(密钥 64 比特,其中 8 比特用于奇偶校验),按当时设计师的分析,使用穷举法破译需 2283 年。随着计算机技术的发展,这个时间被大幅缩短,许多专家认为它已经不够安全,相信政府的安全部门和一些大的犯罪组织已经有能力对

其进行破译。由于 DES 只用于相对不重要的场合,对于这些部门或组织来说不值得动用大量人力、物力来做破译,因此 DES 仍然还有实用价值。DES 曾被建议成为国际标准,但由于其强度不够,同时又由于加密标准与国家利益有重大关系,各国都不放心使用别人设计的算法,因此 ISO 和 ITU-T 都没有采纳其作为标准,同时已宣布不再进行任何有关数据加密算法的标准制定工作。

2. 基本原理

DES 被称为数据加密标准,属于典型的对称密码体制,同时也属于分组加密算法。DES 对数据格式有要求,一般以 64 位为分组加密,当加密数据的长度达不到 64 位倍数要求时,会使用相应规则去填充直到满足要求为止。从本质上来分析会发现 DES 并不那么安全,它依赖于其虚假的表象,从密码学的术语来说就是依赖于“混乱”和“扩散”的原则。其中,混乱是因为需要隐藏所有明文与密文、明文与密钥的关系,而扩散是因为要让明文中的有限的有效位和密钥通过排列组合方式变换出尽可能多的密文。混乱与扩散相结合的效果就是算法的安全性相对变高。

DES 算法的核心是通过对明文进行一系列的排列替换操作来将其加密,过程关键就是如何从给定的初始密钥中得到 16 个子密钥,加密和解密一般通过 16 轮迭代。整个流程可以分为两部分:一部分是密钥生成的过程;另一部分是明文加密成密文的过程。

DES 算法会通过初始密钥计算出 16 个子密钥,这里的初始密钥通常由用户自己设定的大小为 8 字节的数字或字母组成,转换为二进制之后是 64 位,这 64 位中只有 56 位是有效位,剩余的 8 位是奇偶校验位,通过 DES 中的密钥转换表打乱变化为 56 位,如表 3-2 所示。

表 3-2 DES 密钥转换表

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

根据上述方法得到 56 位密钥后就可以计算子密钥了。这里会将 56 位密钥分为两组,每组 28 位,然后每个子密钥根据 DES 子密钥旋转次数表旋转,再重新合并,根据表 3-3 对重组后的密钥进行置换,使 56 位的子密钥缩小为 48 位。

表 3-3 DES 子密钥转换表

子密钥的比特顺序	1	2	3	4	5	6	7	8	9	10	11	12
转换前对应比特	14	17	11	24	1	5	3	28	1	6	21	10
子密钥的比特顺序	13	14	15	16	17	18	19	20	21	22	23	24
转换前对应比特	23	19	12	4	26	8	16	7	27	20	13	2
子密钥的比特顺序	25	26	27	28	29	30	31	32	33	34	35	36
转换前对应比特	41	52	31	37	47	55	30	40	51	45	33	48
子密钥的比特顺序	37	38	39	40	41	42	43	44	45	46	47	48
转换前对应比特	44	49	39	56	34	53	46	42	50	36	29	32

16 个子密钥都重复一次该过程。其目的是保证将初始密钥中的不同位在每一轮排列后应用于加密的数据上。

第二部分是加密和解密数据块，我们得到子密钥就可以开始加密解密数据块了，数据块大小为 64 位，会通过一个初始置换表打乱顺序，具体操作就是查表替换，这样就会得到一个打乱的 64 位明文，然后将这 64 位分为 L_0 和 R_0 两部分，这两部分都是 32 位，然后进行 16 轮的迭代加密或解密过程。具体如图 3-12 所示。

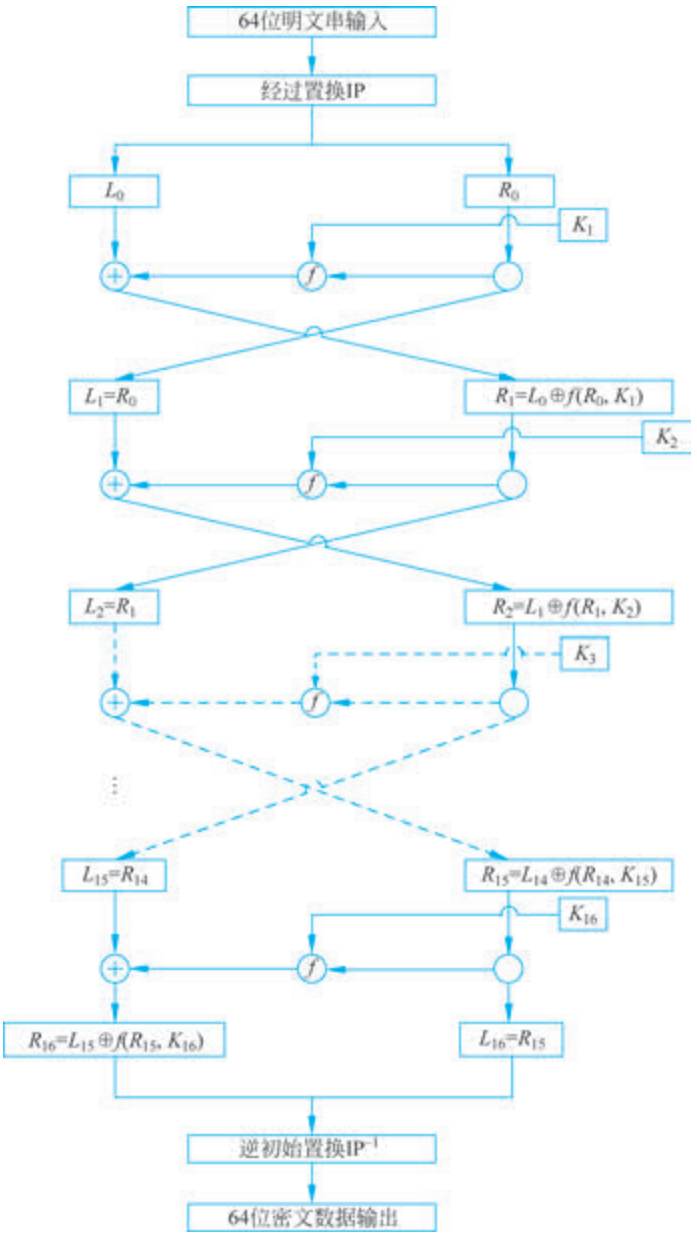

图 3-12　DES 密钥流生成过程

重点在右半部分 R_{i-1}，会进行 16 轮迭代，但是步骤都相同，先是将 R_{i-1} 从 32 位扩展为 48 位，这里也会有一个表，通过查找表进行扩展，这样做的目的是在加密数据的过程中制

造一些雪崩效应,使用数据块中的一位将在下一步操作中影响更多位,从而达到扩散效果。当完成置换后会用这 48 位去与这一轮的子密钥进行异或,这样会产生 48 位的中间值,可以记为 R_{int},可得 $R_{\text{int}}=E(R_{i-1})\oplus K$,这一步被称为扩展置换,置换表见表 3-4。

表 3-4 DES 数据块置换扩散表

密钥比特顺序	1	2	3	4	5	6	7	8	9	10	11	12	13	14
扩散前对应比特	32	1	2	3	4	5	4	5	4	5	6	7	8	9
密钥比特顺序	15	16	17	18	19	20	21	22	23	24	25	26	27	28
扩散前对应比特	8	9	10	11	12	13	12	13	12	13	14	15	16	17
密钥比特顺序	29	30	31	32	33	34	35	36	37	38	39	40	41	42
扩散前对应比特	16	17	18	19	20	21	20	21	20	21	22	23	24	25
密钥比特顺序	43	44	45	46	47	48	49	50	51	52	53	54	55	56
扩散前对应比特	24	25	26	27	28	29	28	29	28	29	30	31	32	1

然后就是进行 S 盒操作,当通过上述方法扩展到 48 位后就会将这 48 位分为 8 组,每组 6 个字符,取第一位和最后一位组成的二进制数作为行数,中间 4 位作为列数,拿到行数和列数就可以去查 S 盒置换表了,查出的数据用 4 位二进制位表示,这样就可以将 48 位缩减回 32 位了,例如第一组为$(101011)_2$,那么它的行数为 112=3,列号$(0101)_2$=5,找到 3 行 5 列,得到数据为 9,转换为 4 位二进制数为$(1001)_2$。具体 S 盒置换表如图 3-13 所示。

S盒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,	16,	3,

S盒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盒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盒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盒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盒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盒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盒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,

图 3-13 DES S 盒置换

当完成 S 盒置换得到 32 位数据后接下来就会进行 P 盒操作，同样也是查表，通过 P 盒置换表完成置换，如表 3-5 所示。

表 3-5　DES P 盒置换表

密钥比特顺序	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
置换前对应比特	16	7	20	21	29	12	28	17	1	15	23	26	5	18	31	10
密钥比特顺序	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
置换前对应比特	2	8	24	14	32	27	3	9	19	13	30	6	22	11	4	25

到了这里就快到了每一轮的最后一步，就是将 P 盒置换后的结果与原始数据的左分组 L_{i-1} 进行异或，完成后左右两个组交换进入下一轮，在最后一轮中左右不用交换，见式(3-3)、式(3-4)。最后一步就是将 R_{16} 和 L_{16} 进行逆置换，该过程同样也是通过查表操作。由于计算力的发展，1977 年提出的 DES 已不再那么安全，结合互联网强大的计算力，通过暴力破解的方式 DES，破译时间已经被缩减到了 1 小时左右。现在主流的做法是采用三重 DES 去解决这一问题，其实就是使用将 DES 算法重复三次，用三个密钥加密三次来达到提高安全性的目的，算法没有发生本质变化。

3. DES 的安全性问题

鉴于 DES 算法的重要性，美国参议院情报委员会于 1978 年曾经组织专家对 DES 的安全性进行了深入的分析。最终的分析报告是保密的，但委员会声明 DES 是安全的。IBM 公司宣布 DES 经过了 17 年的安全分析，是独立研制的。但后来课题组有人透露，DES 算法中的 S 盒曾送给 NSA 检查，且在返回时已被修改过。因此，有人认为 NSA 在 DES 中设立了陷门，但 NSA 予以坚决否认，后来又有人认为这是为了防止 IBM 公司在其中设立陷门。对 DES 的数学分析也一直在进行，例如在 1992 年有人证明了多重 DES 加密是有意义的。

1990 年，以色列数学家 EliBiham 和 AdiShamir 发表了可用于破译各种分组密码体制加密算法(包括 DES 算法)的差分密码分析方法，其基本思想是通过明文/密文对之间的差别分析来寻找最大可能的密钥，再辅之以穷尽分析。差分密码分析方法发现，在同一密钥的作用下，明文对之间的差别很有可能在对应的密文对之间形成特定的差别，这称为这个密钥的特性。该特性反映出这个加密系统在完善保密性方面的缺陷。这些差别在 DES 的每一轮都存在，并具有特定的概率。这样，如果能够发现密钥的这个特性，就有可能发现这个密钥的内容。EliBiham 和 AdiShamir 据此发明了一种对 DES 的选择明文攻击方法：选择具有特定差别的明文对(此处的“差别”是对异或运算而言)，以相同的密钥加密之后，分析所得到密文对的差别，给不同的密钥分配不同的概率，随着所分析的明文/密文对的增加，发现具有最大可能的密钥。

3.2.4　高级加密标准

1. 背景概述

由于 DES 已经不够安全，美国国家标准局 NIST 从 1997 年开始向全世界公开征集新的数据加密标准，通称为 AES。NIST 于 1999 年 8 月第二次 AES 会议上从提交的算法中

筛选出 5 种候选算法。最终,比利时密码专家 Joan Daeman 和 Vincent Rijmen 提交的 Rijndael 算法被提议为 AES 的最终算法,并从 2001 年 6 月起成为美国新的数据加密标准。从那时起,AES 算法开始被广泛应用在各个领域中,被网络硬件和软件供应商所采用。尽管人们对 AES 还有不同的看法,但总体来说,作为新一代的数据加密标准,AES 汇聚了强安全性、高性能、高效率、易用和灵活等优点,可以成为 DES 的替代算法。到目前为止,除了暴力破解 AES 的 256 比特密钥,没有已知的密码分析攻击可以解密 AES 密文。

对 AES 的理解需要对有限域理论有一定的了解,下面简单进行介绍。

2. 基本原理

AES 的全称为 advanced encryption standard,意思是高级加密标准,它的出现就是为了取代 DES 算法。上节中提到,随着算力的提升,可以对 DES 加密进行暴力破解。DES 密钥具有 56 位有效位,因此破解只需要遍历 2^{56} 就可以解出结果。尽管有三种 DES 算法,但是,人们需要在算法上的发展不能完全通过加密次数来实现。于是 1997 年美国国家标准技术研究所宣布收集新的加密算法以求替代 DES。通过层层选择,最后决定以 Rijndael 算法为 AES 算法。该算法有两个重要的特点:第一个是明文的分组大小为 128 位;第二个是支持三种密码标准,即密钥的长度分别为 128 位、192 位、256 位。不同密钥长度的选择会影响加密的轮数,如表 3-6 所示。

表 3-6 AES 密钥长度与加密轮数的关系

AES	密钥长度/32 位	分组长度/32 位	加密轮数
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

3. AES 加密流程

AES 加密过程涉及四种操作,分别是字节替代、行移位、列混淆和轮密钥加。解密过程分别为对应的逆操作。由于每一步操作都是可逆的,按照相反的顺序进行解密即可恢复明文。加/解密中每轮的密钥分别由初始密钥扩展得到。AES 属于分组加密算法,算法规定需要将明文划分成组,每组的数据长度为 128 位,而密钥长度可以是 128 位、192 位、256 位,这三者的主要区别就是加密轮数不一样,128 位的需进行 10 轮加密,192 位的需进行 12 轮加密,256 位的需进行 14 轮加密。本节中以 128 位密钥为例进行讲解。

AES 算法是基于代换-置换网络的迭代算法,主要流程就是循环地对矩阵和向量做查表替换、位运算和矩阵乘积,如图 3-14 所示。在加密过程中,会执行一系列的轮函数,以 AES-128 为例,算法将执行 10 次轮函数,这个轮函数的前 9 次执行操作是一样的,只有第 10 次有所不同,也就是说,一个明文分组会被加密 10 轮。AES 的核心就是实现一轮的所有操作。AES 的处理单位是字节,128 位的输入明文分组和输入密钥都被分成 16 字节。

分析图 3-14 所示的流程,首先扩展密钥是密钥矩阵元素,128 位的密钥被划分为 16 个子块,每个子块 8 位,这 16 个子块钥形成一个 4×4 的矩阵,这个矩阵被称为 State(输入状态)。然后就是变换,其为算法核心的轮函数,主要就是对数据进行“加、减、乘、除”的操作,每一轮变换的输入就是上一轮变换的输出。经过 10 轮的变换操作,明文转换为密文。流程

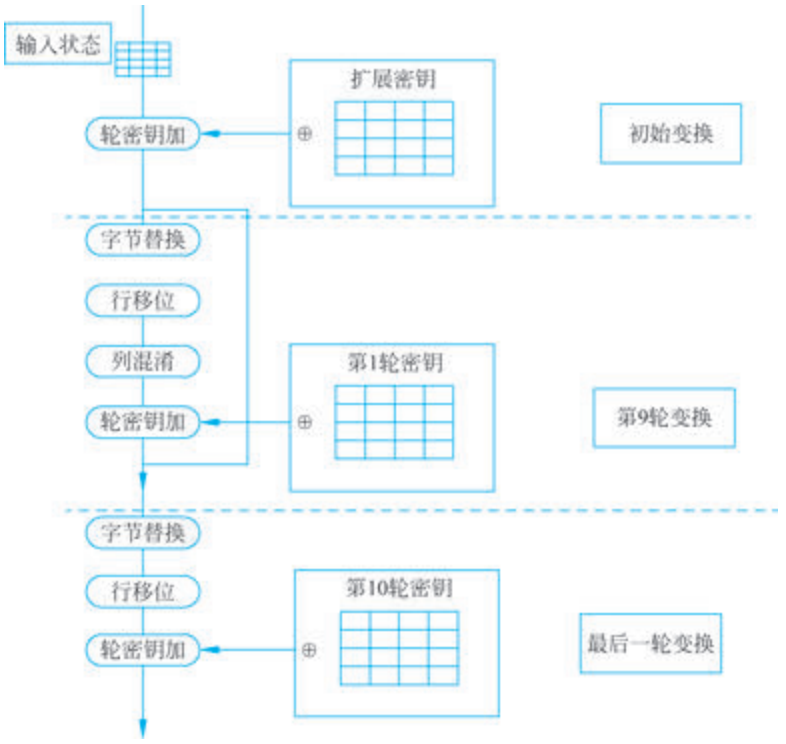

图 3-14　AES 加密流程

图右边的是密钥处理的部分，K_i 是一个子密钥，与上述的明文子块一样，密钥长度与明文块长度一样，通常把它称为轮密。每一轮函数结束后，轮密也会被修改，对应的修改算法或过程称为密钥扩展算法。下面详细分析一下每一步的过程和结果。

首先是 AES 的轮函数 Round，一个完整的 Round 函数分为 4 个步骤，它们分别是字节替换(SubBytes)、行移位(ShiftRows)、列混淆(MixColumns)和与轮密钥(RoundKey)做异或操作，如图 3-15 所示。其中，S 表示逐字节变换。

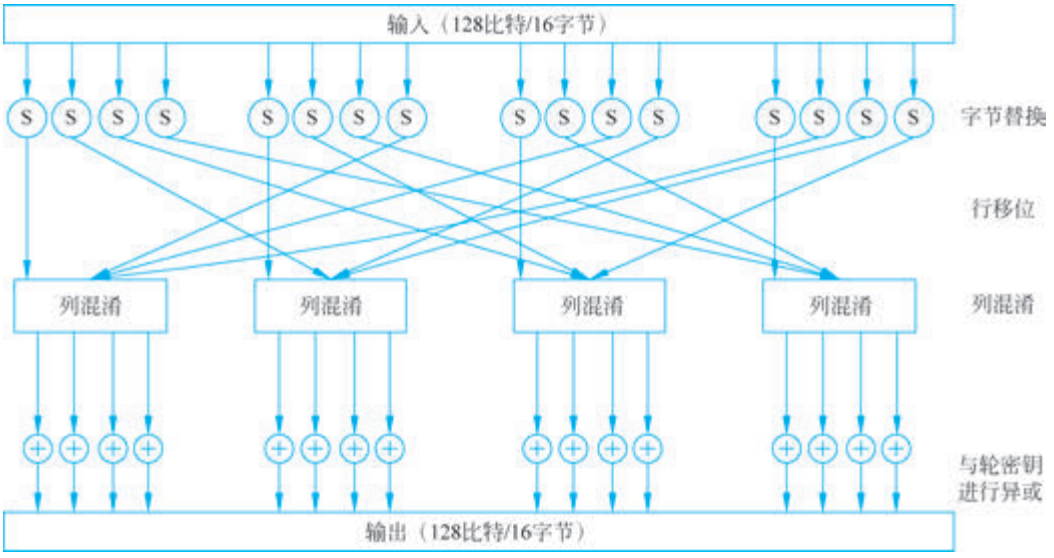

图 3-15　AES 轮函数图

字节替换主要是查找一张内置的表格，把 State 每一块(即 8 位，以下用 byte 代替)逐一替换为表格对应的 byte。这个内置的表格被称为 S 盒，是一张 16×16 的常量表格，这里 8 位会划分成两部分，每部分 4 位，每部分的 4 位转换为十六进制，分别代表 S 盒的行数和列数，然后按照这个行列数信息去查询并进行替换。

行移位就是对 State 每一行内部做平移操作。例如，当密钥长度为 128 位时，State 矩阵的第 0 行左移 0 位，第 1 行左移 1 位，第 2 行左移 2 位，第 3 行左移 3 位，如图 3-16 所示。

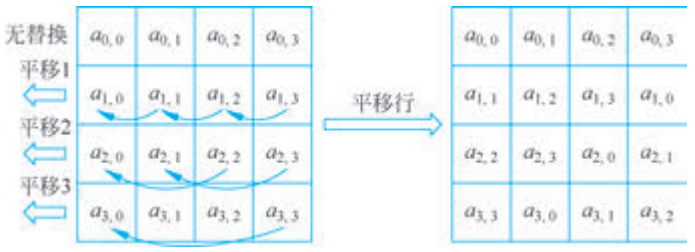

图 3-16　AES 平移行图

列混淆将输入的 4×4 矩阵左乘一个给定的 4×4 矩阵。但这里的矩阵相乘不是普通意义上的矩阵元素先相乘再相加，而是将传统矩阵相乘步骤中的元素相加运算更换成异或运算，矩阵元素的相乘也有相应变化，即乘积矩阵中的每个元素都是一行和一列中的对应元素的乘积之和。

轮密钥加产生子密钥矩阵，由最初的轮密钥矩阵扩展得来。例如 10 轮加密，每轮就会产生 1 个子密钥。

最终轮，此环节就是那个 10 轮加密的最后一轮，而它与前面 9 轮加密的不同之处在于，四个环节中少了列混淆这一环节，其他的都一样。

3.2.5　国密 SM4 分组密码算法

SM4 分组密码算法是 2012 年 3 月 21 日实施的一项行业标准。2021 年 6 月 25 日，我国 SM4 分组密码算法作为国际标准 ISO/IEC 18033-3—2010/AMD1—2021《信息技术安全技术加密算法第 3 部分：分组密码补篇 1：SM4》，由 ISO/IEC 正式发布。SM4 密码算法经过我国专业密码机构的充分分析测试，被证实可以抵抗差分攻击、线性攻击等现有攻击，因此是安全的。

SM4 算法是继 SM2/SM9 数字签名算法、SM3 密码杂凑算法、祖冲之密码算法和 SM9 标识加密算法之后，我国又一个被纳入 ISO/IEC 并正式发布的商用密码算法，标志着我国商用密码算法国际标准体系的进一步完善，展现了我国先进的密码科技水平和国际标准化能力，对提升我国商用密码产业发展、推动商用密码更好地服务“一带一路”建设具有重要意义。

1. SM4 算法基本原理

SM4 有很高的灵活性，所采用的 S 盒可以被灵活地替换，以应对突发性的安全威胁。算法的 32 轮迭代采用串行处理，这与 AES 中每轮使用代换和混淆并行地处理整个分组有很大不同。

SM4 算法是一种采用 Feistel 结构的迭代分组密码算法，由加解密算法和密钥扩展算法组成，用于无线局域网产品。该算法的分组长度为 128 比特，密钥长度为 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。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。

SM4 密码算法使用模 2 加和循环移位作为基本运算，每次迭代执行一个轮函数，该轮函数由一个非线性变换和线性变换复合而成，其中的非线性变换由 S 盒给出。轮密钥表示为 rk_i，与合成置换 T 共同构成轮函数。轮密钥的产生与图 3-15 所示的流程类似，由加密密钥作为输入生成。不同轮函数中的线性变换是不同的，同时还有些参数的区别。

2. SM4 加解密流程

SM4 密码算法将明文加密成密文的流程可以分解为三步，如图 3-17 所示。

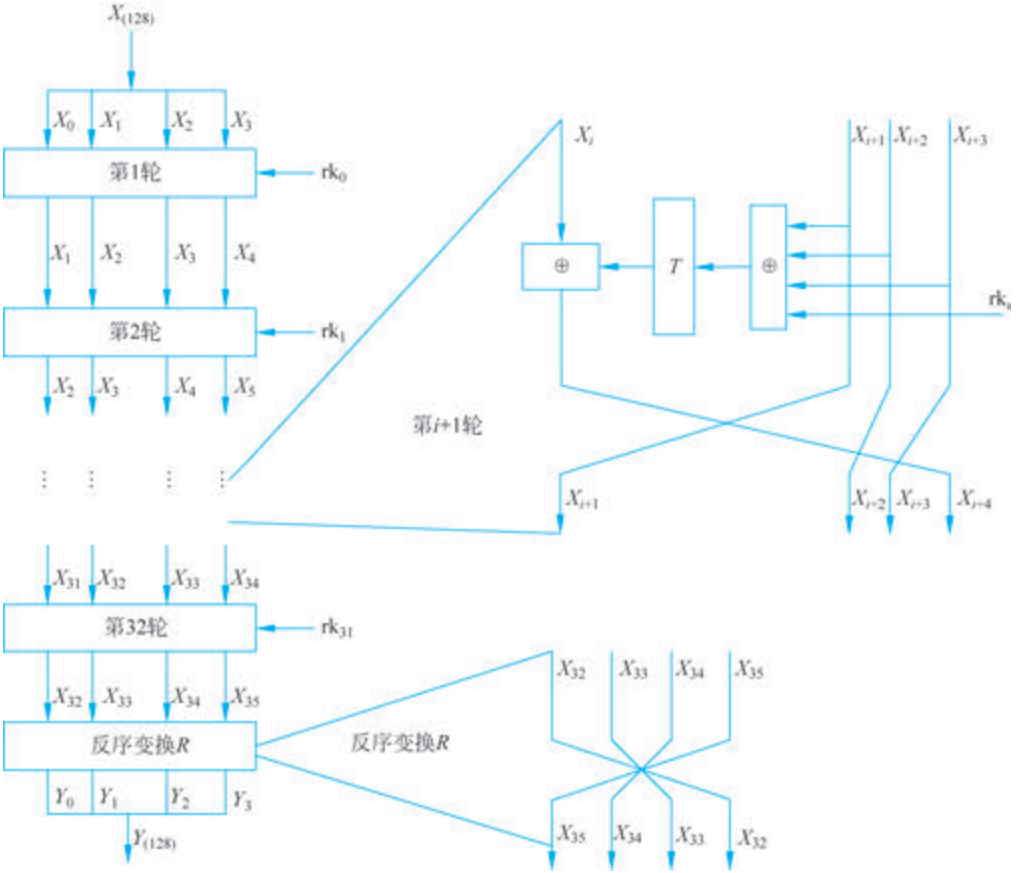

图 3-17　SM4 加密流程

(1) 将 128 比特的明文 $X_{(128)}$ 分成 4 个 32 比特的字 X_1, X_2, X_3, X_4。

(2) 将上述得到的字进行 32 轮的轮操作。

(3) 进行过 32 轮操作的 4 个字进行反序变换后组成 128 比特的密文 $Y_{(128)}$。

其中(1)密钥：加密密钥的长度为 128 比特，表示为 $MK=(MK_0, MK_1, MK_2, MK_3)$，其中 MK_i 为 32 位，轮密钥表示为$(rk_0, rk_1, \cdots, rk_{31})$，其中 rk_i 为 32 位。

(2) 轮函数 F：假设输入为(X_0,X_1,X_2,X_3)，X_i 为 32 位，则轮函数为 $F=(X_0,X_1,X_2,X_3,r_k)=X_0\oplus T(X_1\oplus X_2\oplus X_3\oplus r_k)$。

(3) 合成置换：T 函数是一个可逆变换，由一个非线性变换 r 和线性变换 L 复合而成的，即 $T(\ )=L(r(\ ))$。

(4) 非线性变换由四个并行的 S 盒构成，设输入为 $A=(a_0,a_1,a_2,a_3)$，输出为 $B=(b_0,b_1,b_2,b_3)$，其中 a_i 和 b_i 为 8 位。每个 S 盒的输入都是一个 8 位的字节，将这 8 位的前四位对应的十六进制数作为行编号，后四位对应的十六进制数作为列编号，然后用相应位置中的字节代替输入的字节。图 3-18 为 S 盒置换表。

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D6	90	E9	FE	CC	2A	BE	B7	16	B6	22	B7	16	B6	2C	33
1	2B	67	9A	76	2A	91	EF	44	AA	C3	14	E6	B1	2E	12	99
2	9C	42	50	F4	91	C9	8	33	54	0B	13	C5	E4	43	AC	62
3	E4	B3	1C	A9	Cg	F3	73	33	7A	98	0B	D1	0B	ED	3F	A6
4	47	7	A7	FC	F3	73	17	95	80	DF	94	36	43	75	4F	A8
5	68	6B	81	B2	71	64	DA	BA	83	59	3C	9B	44	FA	9D	35
6	1E	24	0E	5E	63	58	D1	8B	FB	EB	0F	2E	12	3E	78	87
7	D4	0	46	57	9F	D3	27	EE	17	22	7C	36	2E	34	C8	9E
8	EA	BE	8A	D2	40	C7	38	EF	DA	36	2	FA	21	D5	15	A1
9	E0	AE	5D	A4	9B	34	1A	13	A2	F7	F2	94	1A	93	1A	E3
A	1D	F6	E2	2E	82	66	CA	8E	52	93	32	1B	1E	32	5B	6F
B	D6	DB	37	45	DE	FD	8E	BC	60	29	23	29	72	25	5A	51
C	8D	1B	AF	92	BB	DD	BC	7B	55	FF	6A	6A	29	1A	B4	D8
D	0A	C1	31	88	A5	CD	7B	BD	2F	D9	5C	12	23	E3	39	B0
E	89	69	97	4A	0C	96	77	7E	7F	74	D0	23	C5	6E	C6	84
F	18	F0	7D	EC	3A	DC	4D	20	EE	B9	F1	5F	3E	CB	39	48

图 3-18 S 盒置换表

(5) 线性变换 L：线形变换的输入就是 S 盒的输出，即 $C=L(B)=B\oplus(B<<<2)\oplus(B<<<10)\oplus(B<<<18)\oplus(B<<<24)$，线性变换的输入和输出都是 32 位的。

经过了 32 轮的迭代运算后，最后再进行一次反序变换即可得到加密的密文，即密文 $C=(Y_0,Y_1,Y_2,Y_3)=R(X_{32},X_{33},X_{34},X_{35})=(X_{35},X_{34},X_{33},X_{32})$。

SM4 算法的解密流程和加密流程一致，只不过轮密钥的使用顺序变成了$(\mathrm{rk}_{31},\mathrm{rk}_{30},\cdots,\mathrm{rk}_0)$。

3. SM4 算法的密钥处理

1) 四个术语

加密密钥：SM4 算法的加密密钥长度为 128 比特，将其分为四项，其中每一项都为 32 位的字，表示为 $\mathrm{MK}=(\mathrm{MK}_0,\mathrm{MK}_1,\mathrm{MK}_2,\mathrm{MK}_3)$。

系统参数：其中每一项都为 32 位的字，表示为 $\mathrm{FK}=(\mathrm{FK}_0,\mathrm{FK}_1,\mathrm{FK}_2,\mathrm{FK}_3)$。

固定参数：用于密钥扩展算法。其中每一项都为 32 位的字，表示为 $\mathrm{CK}=(\mathrm{CK}_0,\mathrm{CK}_1,\cdots,\mathrm{CK}_{31})$。

轮密钥：其中每一项都为 32 位的字。轮密钥由加密密钥通过密钥扩展算法生成，表示为$(\mathrm{rk}_0,\mathrm{rk}_1,\cdots,\mathrm{rk}_{31})$。

2) 密钥扩展算法

将密钥的每个字分别与系统参数的每个字做异或运算得到(K_0,K_1,K_2,K_3)，再将得

到的后三个字与固定参数 CK_0 做异或运算，之后进行函数 T 运算得到值 C，最后将函数 T 运算得到的 C 与 K_0 做异或运算就得到了第一轮的子密钥，也是下一轮密钥运算的 K_4。所以密钥扩展算法可以总结为下面两步。

第一步：密钥与系统参数的异或。

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

第二步：获取子密钥。

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

3.2.6　祖冲之系列密码算法

祖冲之算法集(ZUC 算法)是由我国学者自主设计的加密和完整性算法，包括祖冲之算法、加密算法 128-EEA3 和完整性算法 128-EIA3。由中国科学院信息工程研究所信息安全国家重点实验室和中国科学院数据与通信保护研究教育中心(DCS 中心)联合主办的第一届祖冲之算法国际研讨会于 2010 年 12 月 2 日至 3 日在北京召开。本次国际研讨会对于加强祖冲之算法研究分析成果的国内和国际交流，扩大祖冲之算法的公开评估范围，加强祖冲之算法的安全性评估力度，进而推进祖冲之算法 4G 通信国际加密标准的进度具有重要的现实意义。祖冲之算法已经被国际组织 3GPP 推荐为 4G 无线通信的第三套国际加密和完整性标准的候选算法。

祖冲之算法是一个面向字的流密码。它采用 128 位的初始密钥和一个 128 位的初始向量(**IV**)作为输入，并输出关于字的密钥流(每 32 位被称为一个密钥字)。密钥流可用于对信息进行加密和解密操作。

1. 祖冲之算法结构

祖冲之算法在逻辑上分为上、中、下三层，如图 3-19 所示。上层是 16 级线性反馈移位寄存器(LFSR)，中层是比特重组(BR)，下层是非线性函数 F。

(1) 线性反馈移位寄存器 LFSR，包括 16 个 31 比特寄存器单元变量 $S_0,S_1,\cdots,S_{15}$，LFSR 的运行模式有两种：初始化模式和工作模式。

在初始化模式下，LFSR 接收一个 31 比特字 u。u 是由非线性函数 F 的 32 比特输出 W 通过舍弃最低位比特得到，即 $u=W>>1$(W 右移 1 位)。

工作模式相比于初始化模式，不同点在于没有输入。

(2) 比特重组(BR)。

输入为 LFSR 里面的变量 $S_0,S_2,S_5,S_7,S_9,S_{11},S_{14},S_{15}$，输出为 4 个 32 比特的字，计算过程如下。

$$X_0=S_{15H}\parallel S_{14L}$$
$$X_1=S_{11L}\parallel S_{9H}$$
$$X_2=S_{7L}\parallel S_{5H}$$
$$X_3=S_{2L}\parallel S_{0H}$$

(3) 非线性函数 F。

F 当中包含了两个 32 比特记忆单元变量 R_1 和 R_2。输入为上次经过比特重组的前三

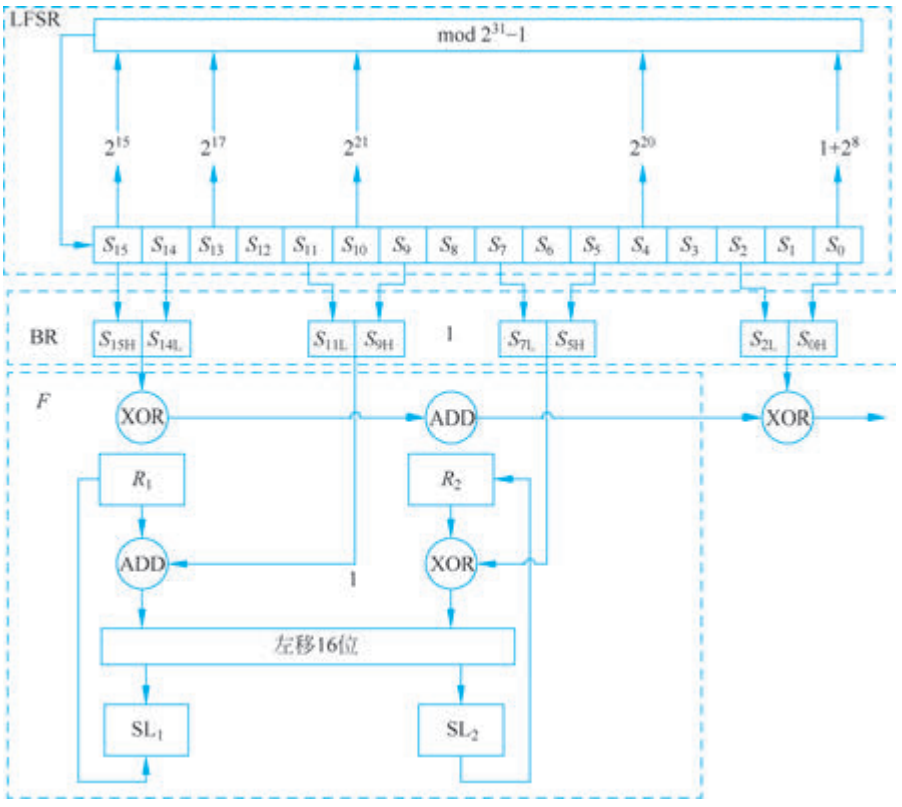

图 3-19 祖冲之算法结构

个 32 比特字 X_0,X_1,X_2,输出为一个 32 比特字 W,计算过程如下。

$$W=(X_0 \oplus R_1)+R_2$$
$$W_1=R_1+X_1$$
$$W_2=R_2 \oplus X_2$$
$$R_1=S(L_1(W_{1L} \| W_{2H}))$$
$$R_2=S(L_2(W_{2L} \| W_{1H}))$$

其中,S 为 32 比特的 S 盒变换,L_1,L_2 是 32 比特线性变换,定义如下。

$$L_1(X)=X \oplus (X <<< 2) \oplus (X <<< 10) \oplus (X <<< 18) \oplus (X <<< 24)$$
$$L_2(X)=X \oplus (X <<< 8) \oplus (X <<< 14) \oplus (X <<< 22) \oplus (X <<< 30)$$

2. 密钥装入

在密钥装入过程中,将 128 比特的初始密钥 K 和 128 比特的初始向量 **IV** 扩展为 16 个 31 比特字,作为 LFSR 寄存器单元变量 S_0,S_1,…,S_{15} 的初始状态。设 K 和 **IV** 分别为 $K_0 \| K_1 \| \cdots \| K_{15}$ 和 $\mathbf{IV}_0 \| \mathbf{IV}_1 \| \cdots \| \mathbf{IV}_{15}$。其中,$K_i$ 和 $\mathbf{IV}_i$ 均为 8 比特字节,$0 \leqslant i \leqslant 15$。密钥装入过程如下。

(1) D 为 240 比特的常量,可按如下方式分成 16 个 15 比特的子串 $D=d_0 \| d_1 \| \cdots \| d_{15}$。其中,

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

(2) 对 $0 \leqslant i \leqslant 15$，有 $S_i = K_i \| d_i \| \mathbf{IV}_i$。

3. 算法运行

祖冲之算法的运行分为两个阶段：初始化阶段和工作阶段。在第一个阶段，密钥和初始向量进行初始化，即不产生输出。第二个阶段是工作阶段，在这个阶段，每一个时钟脉冲产生一个 32 比特的密钥输出。

(1) 初始化阶段。首先把 128 比特的初始密钥 K 和 128 比特的初始向量 **IV** 按照前面介绍的密钥装入方法装入 LFSR 的寄存器单元变量 $S_0, S_1, \cdots, S_{15}$ 中作为 LFSR 的初态，并置 32 比特记忆单元变量 R_1 和 R_2 为全 0，然后执行下述操作 32 次：①比特重组；②$W = F(X_0, X_1, X_2)$；③LFSR 初始化。

(2) 工作阶段。首先执行下列过程一次，即①比特重组；②$W = F(X_0, X_1, X_2)$；③LFSR 初始化。并将 F 的输出 W 舍弃。然后进入密钥输出阶段，在密钥输出阶段，每运行一个轮次，执行下列过程一次，并输出一个 32 比特的密钥字 Z。即①比特重组；②$Z = F(X_0, X_1, X_2) \oplus X_3$；③LFSR 工作模式。

3.3　公钥密码体制

3.2 节讨论的密码体制主要是对称(单钥)密码体制，对称密码体制的一个严重的缺点就是在任何密文传输之前，通信双方必须使用一个安全渠道协商加密密钥，但在实际应用中，做到这一点是很难的。此外，如何为数字化的信息或文件提供一种类似于为书面文件手写签字的方法(即签名、认证功能)，也是对称密码体制难以解决的问题。1976 年，W. Diffie

和 M. E. Hellman 发表了《密码学中的新方向》(*New Directions in Cryptography*)一文,提出了公钥密码体制的观点,引起了密码学领域的一场变革。1977 年,Rivest、Shamir 和 Adleman 提出了第一个比较完善的公钥密码体制 RSA。公钥密码体制很好地解答了对称密码体制面临的两个问题。

3.3.1 公钥加密基础

1. 公钥加密思想

Diffie 和 Hellman 在 1976 年首次公开提出了公钥加密思想,这是有文字记载的几千年来密码学领域中第一次真正革命性的进步。公钥算法基于数学函数,但不是像对称加密算法那样基于比特模式的简单操作。公钥密码算法采用了两个相关密钥,将加密与解密能力分开。其中一个密钥是公开的,称为公钥,用于加密;另一个是用户专有的,因此是保密的,称为私钥,用于解密。公钥密码体制具有如下重要特性:已知密码算法和公钥求解私钥,在计算上是不可行的。

为了叙述方便,我们把信息发送方设为 A,信息接收方设为 B。公钥密码算法按下述步骤对信息实施保护。

(1) B 产生一对密钥 PK_B 和 SK_B,其中是 PK_B 为公钥,SK_B 为私钥。

(2) B 将 PK_B 公开,保密 SK_B。

(3) A 要想向 B 发送信息 M,则使用 B 的公钥 PK_B 加密 M,表示为 $C=E_{PK_B}(M)$。

(4) B 收到密文 C 后,则用自己的私钥 SK_B 解密,表示为 $M=D_{SK_B}(C)$。

公钥密码算法应满足如下要求。

(1) B 产生自己的公、私密钥对,在计算上是容易的。

(2) A 用 B 的公钥对信息 M 加密以产生密文 C,在计算上是容易的。

(3) B 用自己的私钥对 C 解密,在计算上是容易的。

(4) 攻击者由 B 的公钥求 B 的私钥,在计算上是不可行的。

(5) 攻击者由密文 C 和 B 的公钥,恢复明文在计算上是不可行的。

(6) 加解密次序可以交换(但不是对所有的算法都做要求),两个相关密钥中的任何一个都可以用于加密,另一个密钥用于解密。

以上要求的本质在于求一个单向陷门函数。单向陷门函数是指满足下列条件的函数 f。

(1) 给定 x,计算 $y=f(x)$是容易的。

(2) 给定 y,计算 x,使 $x=f^{-1}(y)$是困难的(所谓困难是指计算上相当复杂,耗时极大,已失去实际意义)。

(3) 存在 δ,使得在已知 δ 时,对给定的任何 y,则计算 $x=f^{-1}(y)$是容易的(若相应的 x 存在)。

我们把仅满足上述前两个条件的函数称为单向函数;满足第三条则称函数具有陷门性,δ 称为陷门信息。当用陷门函数 f 作为加密函数时,可将 f 公开,这相当于公钥;f 函数的设计者将 δ 保密,δ 相当于私钥,由于加密函数 f 是公开的,任何人都可以将信息 x 加密成 $y=f(x)$,然后发送给函数的设计者(当然可以通过不安全信道传送);由于设计者拥

有 δ，自然可以解出信息 x。另外，单向陷门函数的第二条性质表明窃听者由截获的密文 $y=f(x)$ 推测 x 是不可行的。

2. 对公钥加密的误解

在继续进行前，我们首先看看关于公钥加密的几种常见误解。

(1) 公钥加密比传统加密更抗密码分析。实际上，任何加密方案的安全性都取决于密钥长度和攻破密码所需的计算量。从抗密码分析的角度来说，没有任何原理能说明是传统加密还是公钥加密更加优越。

(2) 认为公钥加密是一种通用技术，传统加密则过时了。相反，由于当前公钥加密方案的计算开销太大，传统加密淘汰的可能性似乎还无法预测。

(3) 相对于传统加密使用密钥分发中心的相当笨拙的握手协议，使用公钥密码时的密钥分发似乎是微不足道的。事实上，公钥密码除需要某种形式的协议之外，还常常需要中心代理。所以与传统加密相比，公钥加密所需的操作并不简单或者高效。

3. 公钥密码体制的组成

公钥加密方案由以下五个部分组成。

(1) 明文。它是可读的消息或数据，用作算法的输入。

(2) 加密算法。加密算法对明文进行各种形式的变换。

(3) 公钥和私钥。它们是被选择的一对密钥，如果一个密钥用于加密，则另一个密钥用于解密。加密算法所执行的具体变换取决于输入端提供的公钥或私钥。

(4) 密文。它是输出的混乱的消息，取决于明文和密钥。对于给定的消息，两个不同的密钥将产生两个不同的密文。

(5) 解密算法。该算法接收密文和匹配的密钥，生成原始的明文。

4. 公钥密码系统的应用

公钥密码算法不仅能用于保护传递信息的保密性，而且还能对发送方发送的信息提供验证，如 B 用自己的私钥 SK_B 对明文 M 加密成密文 C，当 B 将 C 发往 A 时，A 用 B 的公钥 PK_B 对 C 解密。由于从 M 得到 C 是经过 B 的私钥 SK_B 加密的，只有 B 才能做到，因此 C 可当成 B 对 M 的数字签名。另外，任何人只要不知道 B 的私钥 SK_B 就不能对其进行篡改，所以以上过程获得了对信息来源和信息完整性的认证。以上认证过程中，由于信息是由 B 的私钥加密的，因此信息不能被他人篡改，但却能被他人窃听，这是因为任何人都能用 B 的公钥对信息进行解密。为了提供认证和保密功能，可采取双重加解密，B 首先用自己的私钥对信息 M 加密，再用 A 的公钥进行第二次加密，解密过程是 A 先用自己的私钥，后用 B 的公钥对收到的密文进行两次解密。

在广义上可以把公钥加密系统分为如下三类。

(1) 加密/解密：发送者用接收者的公钥加密消息。

(2) 数字签名：发送者用自己的私钥签名消息。把加密算法作用于消息或者消息函数的一小块数据，这样就完成了对消息的签名。

(3) 密钥交换：双方联合操作来交换会话密钥。这可以使用几种不同的方法，且需要

用到单方或双方的私钥。

3.3.2 RSA 算法

RSA 算法是在 1977 年由 Ron Rivest、Adi Shamir 和 Len Adleman 在 MIT 开发的,于 1978 年首次发表。RSA 算法从那时起便占据了绝对的统治地位,成为最广泛接受和实现的公钥加密方法。RSA 算法的数学基础是初等数论中因子分解理论和欧拉定理,并建立在大整数因子分解的困难性之上。

1. RSA 算法描述

1) 密钥对的产生

产生 RSA 密钥的过程如下所述。

(1) 选择两个大素数 p 和 q。

(2) 计算:$n=pq$,$\psi(n)=(p-1)(q-1)$。

(3) 随机选择一个整数 e,要求 e 和 $\psi(n)$ 互素。

(4) 找到一个整数 d,满足 $ed \bmod \psi(n)\equiv 1$。

只要 e 和 n 满足以上条件,d 肯定是存在的,原因如下:由于 e 和 $\psi(n)$ 互素,用 e 乘以 $\psi(n)$ 的完全余数集合中的每一个元素后再模 $\psi(n)$,得到的余数将以不同的次数涵盖 $\psi(n)$ 的完全余数集合中的所有数,则肯定有一个余数为 1。最后,e 和 n 便是 RSA 的公钥,d 是私钥。此时 p 和 q 不再被需要,它们应该被舍弃掉,但绝不可泄露。

2) 加密、解密

对明文做加密运算如下。

$$c = m^e \bmod n \tag{3-5}$$

得到的 c 即为密文,即密文 c 为 m 的 e 次方除以 n 后得到的余数。

对密文的解密运算如下。

$$m = c^d \bmod n \tag{3-6}$$

解密运算的结果是得到原来的明文 m,即明文 m 为 c 的 d 次方除以 n 后得到的余数。

3) 算法证明

RSA 算法的证明过程如下。

因为 $ed \bmod \psi(n)\equiv 1$,可将 ed 表示为

$$ed = k\psi(n) + 1 \tag{3-7}$$

其中,k 为任意整数。

将 c 用 $m^e \bmod n$ 替换后,计算 $c^d \bmod n$:

$$c^d \bmod n = (m^e)^d \bmod n = m^{ed} \bmod n = m^{k\psi(n)+1} \bmod n \tag{3-8}$$

由欧拉定理的推论可以证明:

$$m^{k\psi(n)+1} \bmod n = m \bmod n \tag{3-9}$$

由于 m 小于 n,因此

$$m \bmod n = m \tag{3-10}$$

即 $c^d \pmod n = m$,解密的结果就是原明文 m。

2. RSA 算法举例

为了让读者进一步了解 RSA 的加解密过程，我们在下面举一个简单的例子说明这一过程。

例 3.1　设 $p=43, q=59, n=pq=43\times59=2537$，则 $\phi(n)=42\times58=2436$。

取 $e=13$，解方程 $de\equiv1 \pmod{2436}$

$$2436=13\times187+5$$
$$13=2\times5+3$$
$$5=3+2$$
$$3=2+1$$

得
$$1=3-2$$
$$3=13-2\times5$$
$$5=2436-13\times187$$

推出
$$1=3-2=3-(5-3)=2\times3-5=2\times(13-2\times5)-5=$$
$$2\times13-5\times5=2\times13-5\times(436-13\times187)=$$
$$937\times13-5\times2436$$

即
$$937\times13\equiv1 \pmod{2436}$$

取
$$e=13,\quad d=937$$

若有明文：public key encryptions，先将明文分块为

pu bl ic ke ye nc ry pt io ns

如果利用英文字母表的顺序：a 为 00，b 为 01，c 为 02，…，y 为 24，z 为 25，将明文数字化得

1520 0111 0802 1004 2404
1302 1724 1519 0814 1418

加密后得密文：

0095 1648 1410 1299 1365
1379 2333 2132 1751 1289

3. RSA 算法的安全性

有两种可能的方法可以用来攻击 RSA 算法。其中一种方法是穷举攻击，即试遍所有可能的私钥。所以 e 和 d 的比特数越大，算法越安全。然而，因为密钥的生成和加密/解密所需的运算都比较复杂，所以密钥越大，系统运行得越慢。

关于分析破译 RSA 的大部分讨论都集中于如何分解 n 为两个素数。由于大数 n 具有很大的素因子，在过去对 n 进行因式分解非常困难，但是如今它已经不如以前那么困难了。发生在 1977 年的著名事件恰好反映了这种情况。RSA 的三名创始人挑战 *Scientific American* 的读者，让读者去破译他们发表在 *Martin Gardner* 的 Mathematical Games 专栏中的密文。每翻译出来一句明文，他们就提供 100 美元的奖励。他们预计在大约 4×10^6 年之内都不可能有人破译出明文。1994 年 4 月，Internet 上的一个工作组使用了 1600 多台计算机仅仅工作了 8 个月便破译出了明文并拿到了那笔奖励金。该挑战使用的公钥长度（n

的大小)为 129 比特的十进制数字或者大约 428 比特的二进制数字。这样的结果并不意味着 RSA 失效,它只是意味着必须使用更大尺寸的密钥。目前 1024 比特(大约 300 比特的十进制数)的密钥已经可以满足几乎所有应用的强度要求。

3.3.3 ElGamal 算法

ElGamal 公钥密码体制是 1984 年斯坦福大学的 Tather ElGamal 提出的一种基于离散对数问题困难性的公钥体制。该算法安全性依赖于计算有限域上离散对数难题。求解离散对数目前是困难的,但其逆运算指数运算简单。在加密过程中,生成的密文长度是明文的两倍,且每次加密后都会在密文中生成一个随机整数 K。

1. ElGamal 加密算法思想

假设有 2 个用户 Alice 和 Bob,Bob 欲使用 ElGamal 加密算法向 Alice 发送信息。对于 Alice,首先要选择一个素数 q,α 是素数 q 的本原根(本原根的概念对应模 q 乘法群中的生成元)。

首先 Alice 产生一个 X_A 作为私钥,$X_A \in [1, q-1]$,然后计算 $Y_A = \alpha^{X_A} \bmod q$,进而得到公钥$\{q, \alpha, Y_A\}$。公钥存在于某个可信公开中心目录,任何用户都可访问。

对于 Bob,首先去上述中心目录访问得 Alice 的公钥$\{q, \alpha, Y_A\}$,然后将自己欲发送的明文 M($M \in [1, q-1]$)准备好。

选一个随机整数 K,$K \in [1, q-1]$,计算可解密自己密文的密钥 PrivateK$= Y_A^K \bmod q$(即 $\alpha \times X_A^K \bmod q$),将 M 加密成明文对(C_1, C_2)。其中 $C_1 = \alpha^K \bmod q$,$C_2 =$ PrivateK$\times M \bmod q$。明文对(C_1, C_2)将被发送给 Alice。

Alice 收到明文对后,首先计算 PrivateK$= C_1 \times X_A \bmod q$(即 $\alpha \times K \times X_A \bmod q$),然后用 PrivateK 解密,得到的明文表示为 $M = C_2 \times$ PrivateK$^{-1} \bmod q$。其中 PrivateK^{-1} 是 PrivateK 的逆元。

2. ElGamal 加密算法流程

ElGamal 加密算法构建密钥对及加密数据传输的流程分为五个步骤,如图 3-20 所示。

(1) 构建密钥对:接收方事先要构建一对密钥,包括公钥和私钥。

(2) 公布密钥:接收方将构建的公钥发布到某个可信公开中心目录,以便所有想与之通信的用户都可以访问到。

(3) 使用公钥加密数据:发送方在发送数据(明文)前,使用接收方的公钥加密要传输的数据为密文对。

(4) 发送加密数据:发送方将生成的密文对,通过信道(可以是不安全的)传输给接收方。

(5) 使用私钥解密数据:接收方用自己的私钥对接收的数据解密,还原成明文。

3. ElGamal 加密算法举例

1) 密钥生成

对于 Bob,首先要随机选择一个大素数 p,且要求 $p-1$ 有大素数因子。再选择一个模

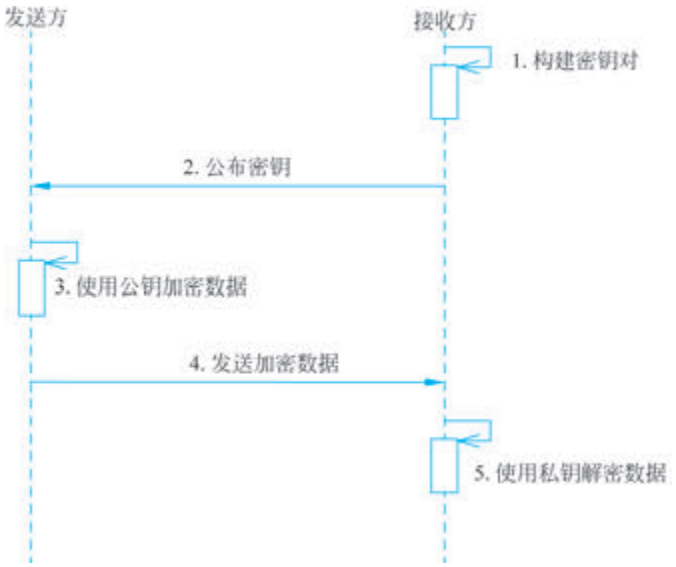

图 3-20　加密数据传输流程

p 的本原元 α，将 p 和 α 公开。例如取 $p=37$，则 p 的一个本原元为 $\alpha=2$。

随机选择一个整数 d 作为密钥，$2\leqslant d\leqslant p-2$。选择 $d=5$，计算 $\beta=\alpha^d \bmod p$，即 $\beta=2^5 \bmod 37=32$。

2）加密

首先选取随机整数 K，假设 $K=7$，则 $y_1=\alpha^K \bmod p=2^7 \bmod 37=128 \bmod 37=17$。假设 Alice 想发送消息 $x=29$，则 $y_2=x\times\beta^K \bmod p=29\times32^7 \bmod 37=33$，将密文 $y=(17,33)$发送给 Bob。

3）解密

Bob 收到密文 $y=(17,33)$后恢复明文过程如下。

$$\begin{aligned}x&=y_2(y_1^d)^{-1} \bmod p\\&=33(17^5)^{-1} \bmod 37\\&=33\times2 \bmod 37\\&=29\end{aligned}$$

4. ElGamal 加密算法应用

ElGamal 加密系统通常应用在混合加密系统中。例如：用对称密码体制来加密消息，然后利用 ElGamal 加密算法传递密钥。这是因为在同等安全等级下，ElGamal 加密算法作为一种非对称密码学系统，通常比对称密码体制要慢。对称加密算法的密钥相比于要传递的消息通常要短得多，所以相比之下使用 ElGamal 加密密钥后用对称加密来加密任意长度的消息的做法要更快一些。

3.3.4　Diffie-Hellman 算法

Diffie-Hellman 密钥交换算法（简称 Diffie-Hellman 算法）是最早的密钥交换算法之

一,由 Whitefield Diffie 和 Martin Hellman 在 1976 年公布。它是一种建立密钥的方法,而不是一个实用的数据加密方法,所以该算法必须和其他一种加密算法结合使用。这种密钥交换技术的目的在于,它使得通信的双方能在非安全的信道中安全地交换密钥,用于加密后续的通信消息。

Diffie-Hellman 密钥交换算法的安全性依赖于计算离散对数的难度,离散对数问题的困难性可以用式(3-11)说明。

$$C^d = M \bmod P \tag{3-11}$$

其中,d 为模 P 的以 C 为底的 M 的对数,通常用来作为隐蔽密钥。在已知 C 和 P 的前提下,由 d 求 M 很容易,相当于只进行一次指数运算;而由 M 反过来求 d,按照目前最快的算法也需要相当多次的运算。例如:若 P 是 1000 比特的质数,由 d 计算 M 只需 200 次 1000 比特数的乘法和除法;但要是求对数 d,大约需要 2^{100} 次运算,因此只要 P 足够大,就可以达到足够的安全性。综上所述,在离散对数密码体制中,指数 d 是需要保密的私钥,而 C、M 和 P 是可以公开的公钥。

1. 算法描述

在介绍 Diffie-Hellman 算法之前,首先介绍原根的定义:一个素数 q 的原根为其各次幂模 q 之后能够产生从 1 到 $q-1$ 的所有整数。也就是说,如果 a 是素数 q 的一个原根,那么数值 $a \bmod q$,$a^2 \bmod q$,…,$a^{q-1} \bmod q$ 是各不相同的整数,并且以某种排列方式组成了从 1 到 $q-1$ 的所有整数。

基于原根的定义及性质,可以定义 Diffie-Hellman 密钥交换算法,该算法描述如下。

(1) 两个全局公开的参数:一个素数 q 和一个整数 a。其中,a 是 q 的一个原根。

(2) A 和 B 希望交换一个密钥,用户 A 选择一个作为私钥的随机数 X_A($X_A<q$),并计算公钥 $Y_A=a^{X_A} \bmod q$。A 对 X_A 的值保密存放,而使 Y_A 能被 B 公开获得。

类似地,用户 B 选择一个私有的随机数 X_B($X_B<q$),并计算公钥 $Y_B=a^{Y_B} \bmod q$。B 对 X_B 的值保密存放,而使 Y_B 能被 A 公开获得。

(3) 用户 A 产生共享秘密密钥的计算方式是:$K=(Y_B)^{X_A} \bmod q$。同样,用户 B 产生共享秘密密钥的计算是:$K=(Y_A)^{X_B} \bmod q$。可以证明,这两个计算将产生相同的结果。因此,相当于双方已经交换了一个相同的秘密密钥。

(4) 因为 X_A 和 X_B 是保密的,敌对方可以利用的参数只有 q,a,Y_A,Y_B,所以敌对方被迫取离散对数来确定密钥。例如,要获取用户 B 的秘密密钥,敌对方必须先计算离散对数,然后再使用用户 B 采用的方法计算其秘密密钥 K。

用户 A 和 B 交换共享密钥的流程如图 3-21 所示。

2. 算法应用举例

下面举例说明用户 A 和 B 如何交换共享密钥。

(1) 密钥交换基于素数 $q=97$ 和 $a=5$(其中 a 是 q 的一个原根)。

(2) A 和 B 分别选择私有密钥 $X_A=36$ 和 $X_B=58$。

(3) 每人计算其公开密钥 $Y_A=5^{36} \bmod 97=50$,$Y_B=5^{58} \bmod 97=44$。

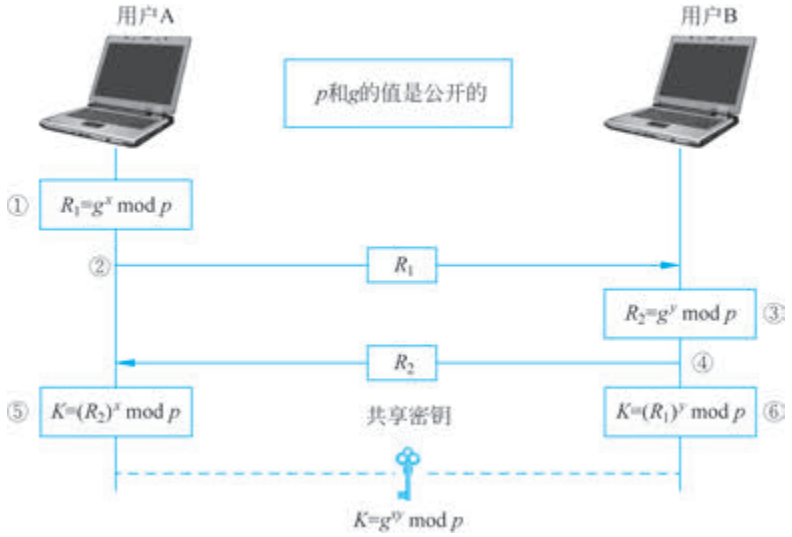

图 3-21　用户 A 和 B 交换共享密钥的流程

(4) 在 A 和 B 相互获取了公开密钥之后，各自通过计算得到双方共享的秘密密钥如下：$K=(Y_B)^{X_A} \bmod 97=44^{36} \bmod 97=75$，$K=(Y_A)^{X_B} \bmod 97=50^{58} \bmod 97=75$。其中，75 是 A 和 B 的共同密钥，已知{97,5,50,44}，攻击者很难计算出 75。

3. 算法特点

Diffie-Hellman 算法是一种确保共享密钥安全穿越不安全网络的方法，它是 Oakley 算法的一个组成部分。这个算法的巧妙在于需要安全通信的双方可以用这个方法确定对称密钥，然后利用这个密钥进行加密和解密。

Diffie-Hellman 算法具有两个吸引力的特征：一是仅当需要时才生成密钥，降低了将密钥存储很长一段时间而致使其遭受攻击的可能；二是除对全局参数的约定，密钥交换不需要事先存在的基础结构。

然而，该算法也存在许多不足。由于没有提供双方身份的任何信息，且该算法是计算密集性的，因此容易遭受阻塞性攻击，即对手请求大量的密钥(受攻击者花费了相对多的计算资源来求解无用的幂系数而不是在做真正的工作)；没办法防止重演攻击，容易遭受中间人的攻击。

Oakley 算法是对 Diffie-Hellman 密钥交换算法的优化，它保留了后者的优点，同时克服了其弱点。Oakley 算法具有五个重要特征：它采用称为 cookie 程序的机制来对抗阻塞攻击；它使得双方能够协商一个全局参数集合；它使用了现时来保证抵抗重演攻击；它能够交换 Diffie-Hellman 公开密钥；它对 Diffie-Hellman 交换进行鉴别以对抗中间人的攻击。

3.3.5　SM2 椭圆曲线密码

1. 椭圆曲线密码

目前影响最大的三类公钥密码是 RSA 密码、ElGamal 密码和椭圆曲线(elliptical curve

cryptography,ECC)密码。其中椭圆曲线密码已成为除 RSA 密码外呼声最高的公钥密码。椭圆曲线密码具有密钥短、签名短、软件实现规模小、硬件实现省电等优点。人们普遍认为,160 位长的椭圆曲线密码的安全性相当于 1024 位的 RSA 密码,因此一些标准化组织已把椭圆曲线密码作为新的信息安全标准。椭圆曲线密码体制逐步成为一个令人十分感兴趣的一个密码分支。1997 年以来形成一个研究热点,特别是其在移动通信安全方面的应用更是加快了这一趋势。

椭圆曲线密码来源于对椭圆曲线的研究,椭圆曲线指的是由魏尔斯特拉斯方程所确定的平面曲线。设(a,b)同属于一个域 F,由方程 $y^2+axy+by=x^3+cx^2+dx+e$ 的所有解(x,y),再加上一个无穷远点 O 所构成的集合 E,称为 F 域上的椭圆曲线。

ECC 所基于的椭圆曲线性质如下。

(1) 有限域上椭圆曲线在点加运算下构成有限交换群,且其阶与基域规模相近。

(2) 类似于有限域乘法群中的乘幂运算,椭圆曲线多倍点运算构成一个单向函数。

在多倍点运算(也称点乘运算)中,已知多倍点与基点,求解倍数的问题称为椭圆曲线离散对数问题(elliptic curve discrete logarithm problem,ECDLP)。对于一般 ECDLP,目前只存在指数级计算复杂度的求解方法,与大数分解问题(integer factorization problem,IFP)及有限域上离散对数问题(discrete logarithm problem,DLP)相比,ECDLP 的求解难度要大得多。因此,在相同安全程度要求下,ECC 较其他公钥密码算法所需的密钥规模要小得多。

2. SM2 算法概述

SM2 椭圆曲线密码算法(简称 SM2 算法)于 2010 年 12 月首次公开发布,2012 年成为中国商用密码标准(标准号为 GM/T 0003—2012),2016 年成为中国国家密码标准(标准号为 GB/T 32918—2016)。迄今为止,与 SM2 算法相关的研究表明,SM2 算法的可证安全性达到了公钥密码算法的最高安全级别,其实现效率相当于或略优于国际标准的同类椭圆曲线密码算法。

SM2 算法的主要内容包括三部分:数字签名算法、密钥交换协议和公钥加密算法,本小节主要介绍 SM2 的公钥加密算法。

3. SM2 算法描述

公钥加密算法规定发送者用接收者的公钥将消息加密成密文,接收者用自己的私钥对收到的密文进行解密将其还原成原始消息。

SM2 算法中,用户 B 的密钥对包括其私钥 d_B 和公钥 $P_B=[d_B]G$。

1) 密钥派生函数

SM2 算法也需要使用密钥派生函数。密钥派生函数 KDF(Z,klen)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 $2^{32}-1$ 位)。

输出:位长为 klen 的密钥数据比特串 K。

2) 加密算法

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,作为加密者的用户 A 应实现如图 3-22 所示的运算步骤。

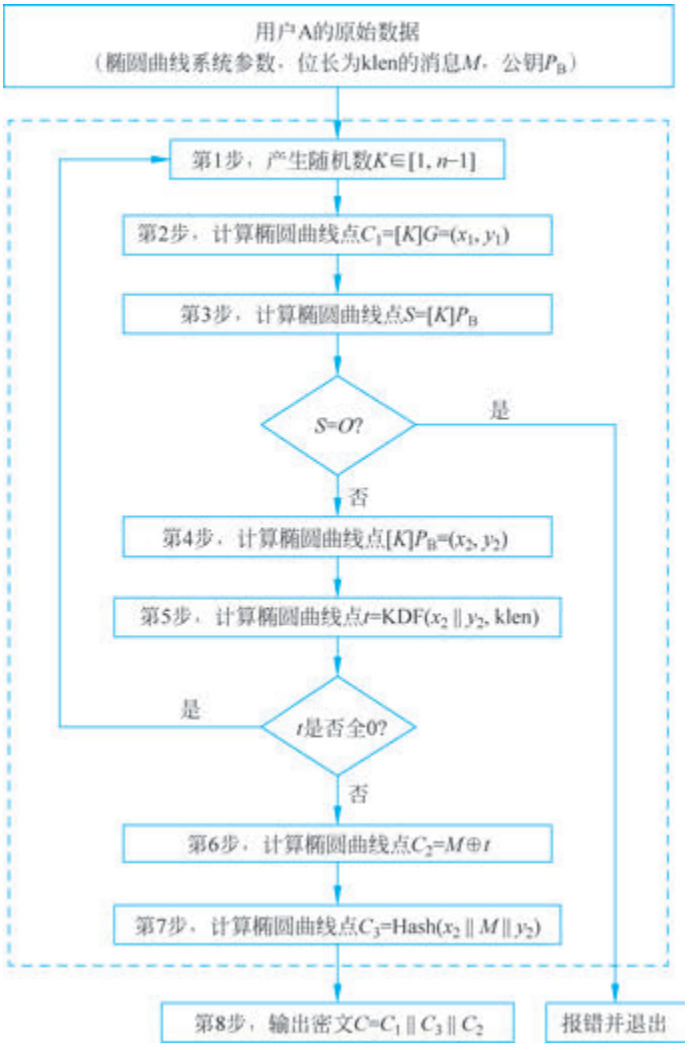

图 3-22　SM2 加密流程

3）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，作为解密者的用户 B 应实现如图 3-23 所示的运算步骤。

4. SM2 公钥加密算法的安全性

攻击者对公钥加密算法的攻击行为包括如下。

（1）选择明文攻击（chosen plaintext attack，CPA）。攻击者可以访问加密预言机（encryption oracle），获得一定的明文/密文对，但他不能访问解密预言机（decryption oracle），攻击者根据所掌握的信息和资源对他想破解的密文给出一个答案。

（2）选择密文攻击（chosen ciphertext attack，CCA1）。攻击者可以访问加密预言机和解密预言机，但在获得一定的明文/密文对后，便不能再访问解密预言机了，攻击者根据所掌握的信息和资源对其想破解的密文给出一个答案。

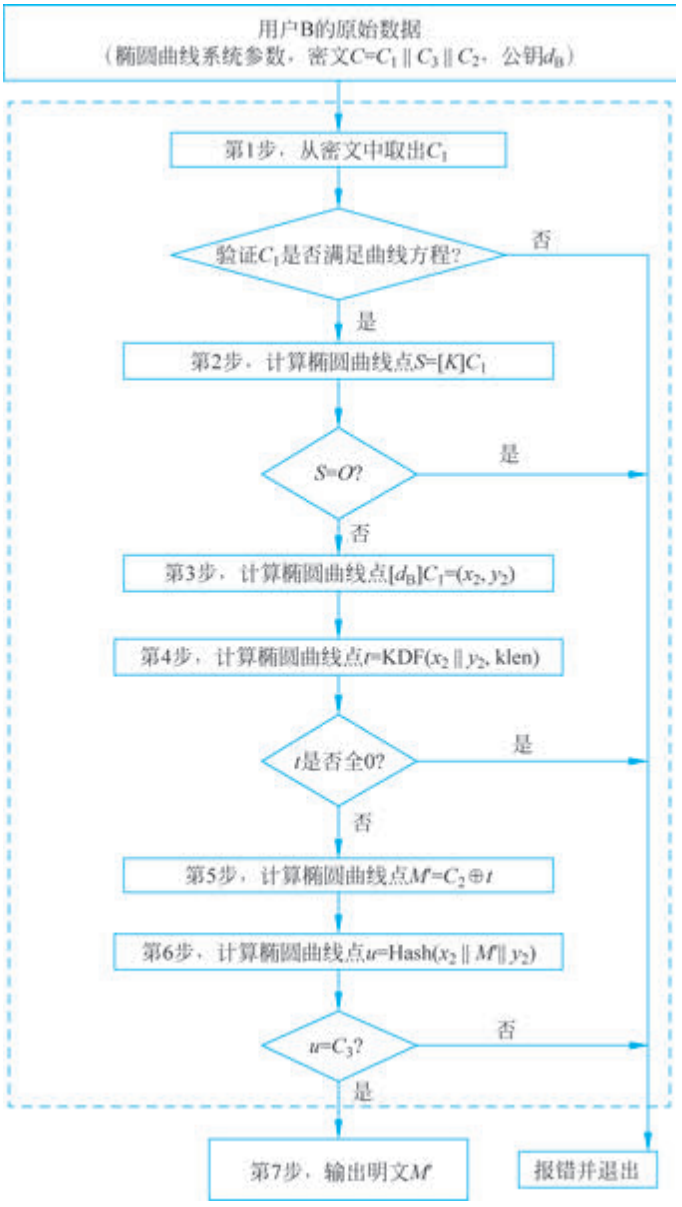

图 3-23 SM2 解密流程

(3) 自主选择密文攻击(adaptively chosen ciphertext attack,CCA2)。攻击者任何时候都可以访问加密预言机和解密预言机,唯一限制是不能直接将其想破解的密文输入解密预言机进行解密,攻击者根据所掌握的信息和资源对其想破解的密文给出一个答案。

显然,上述三种攻击行为中 CCA2 是最强的,公钥加密算法的安全性体现在密文所具备的一些安全属性,这些属性包括:

(1) 单向性(one-wayness,OW)。攻击者在不拥有私钥的前提下,不能计算出任何密文所对应的明文。

(2) 不可区分性(indistinguishability,IND)。攻击者选择两个不同的明文 m_1 和 m_2 输

入加密预言机，加密预言机随机选择其中一个明文加密并返回密文 C，攻击者无法以明显区别于 1/2 的概率正确判断 c 为 m_1 或 m_2 的密文。

(3) 不可延展性(non-malleability，NM)。攻击者无法通过密文 c(对应明文为 m)构造出另一合法密文 c'(对应明文为 m')，使得 m 和 m' 之间存在某种有意义的关系(即有利于破解的关系)。

公钥加密算法的安全性定义为：在攻击者的某种攻击行为下密文具备的某种安全属性。如：IND-CCA2 是指自主选择密文攻击下，密文具备不可区分性；NM-CCA2 是指自主选择密文攻击下，密文具备不可延展性。事实上 IND-CCA2 和 NM-CCA2 是等价的，具有国际密码学界公认的公钥密码算法的最高级别的安全性。

SM2 算法是基于广义 ElGamal 加密算法进行设计的，但广义 ElGamal 加密算法的安全性级别不高，达不到 IND-CCA2 的安全性。对公钥加密算法进行安全性增强，以使其达到 IND-CCA2 的方法比较多，可以概括为以下几类。

(1) OAEP 方法。OAEP 实际上是一种增强的对明文信息的 Padding 规则，比一般 RSA 算法的 Padding 规则更安全，比较适用于增强 RSA 算法的安全性。

(2) 签名加密方法。对每次加密所涉及的秘密信息或者是密文本身签名，攻击者无法通过一个密文构造相关联的另一个合法密文，而解密预言机一旦遇到非法密文则拒绝输出任何信息，从而抵抗 CCA2 攻击。

(3) 混合加密方法。将公钥密码和对称密码结合起来，譬如采用密钥封装机制(key encapsulate mechanism，KEM)将对称密钥用公钥密码密文的形式封装起来，真正对明文加密的是使用该对称密钥的对称加密算法，利用公钥加密和对称加密共同生成密文的认证信息，解密预言机若验证认证信息有误则拒绝输出任何信息，从而抵抗 CCA2 攻击。比较有代表性的混合加密方法是 DHAE。

(4) 使用 Hash 函数。通过安全 Hash 函数产生和验证 MAC，对公钥密码算法中涉及的秘密信息和明文信息进行验证，一旦解密预言机发现验证 MAC 有误，则拒绝输出任何信息，从而抵抗 CCA2 攻击。

SM2 公钥加密算法选择了方法(4)以增强安全性，其密文包括 C_1，C_2，C_3，$C_3 = \text{Hash}(x_2 \| M \| y_2)$，也就是 MAC，计算 C_3 所使用的密钥(x_2, y_2)是根据一次性的秘密随机数据进行 Diffie-Hellman 密钥协商生成的，由于在解密时需要验证 C_3 的正确性，使得 CCA2 攻击者只能通过访问加密预言机获得有效的密文，除此之外不能获得或伪造出任何有效的密文，从而保证了 SM2 公钥加密算法抵抗 CCA2 的安全性。

3.4　密钥管理技术

密钥的管理是整个加密系统中最薄弱的环节，密钥的泄露将直接导致加密系统的失效。从密钥管理的途径窃取机密比从破译的方法花费的代价要小得多，所以对密钥的管理和保护格外重要。一般来说，好的密钥管理系统应做到系统是坚固的，密钥难以被窃取；损失是可控的，在一定条件下窃取了密钥也没用，即密钥有使用范围和使用时间的限制；过程是安全的，密钥的分配和更换过程对用户透明，用户不一定要亲自掌管密钥。

3.4.1 密钥管理的内容

密钥管理就是指对密钥进行如加密、解密、破解等一系列管理的行为。其过程一般可划分为生成密钥、分发密钥、验证密钥、更新密钥、存储密钥、备份密钥、销毁密钥等步骤。

1. 生成密钥

密钥长度应该足够长。一般来说,密钥长度越大,对应的密钥空间就越大,攻击者使用穷举猜测密码的难度就越大。

选择好密钥,避免弱密钥。例如,由自动处理设备生成的随机的比特串是好密钥。

对公钥密码体制来说,生成密钥更加困难,因为密钥必须满足某些数学特征。生成密钥可以通过在线或离线的交互协商方式实现,如密码协议等。

2. 分发密钥

采用对称加密算法进行保密通信,需要共享同一密钥。通常系统中的一个成员先选择一个秘密密钥,然后将它传送给另一个成员或别的成员。ANSI X9.17 标准描述了两种密钥:密钥加密密钥和数据密钥。密钥加密其他需要分发的密钥;而数据密钥只对信息流进行加密。密钥加密密钥一般通过手工分发。为增强保密性,也可以将密钥分成许多不同的部分然后用不同的信道发送出去。

3. 验证密钥

密钥附着一些检错和纠错位来传输,当密钥在传输中发生错误时,能很容易地被检查出来,并且如果需要,密钥可被重传。

接收端也可以验证接收的密钥是否正确。发送方用密钥加密一个常量,然后把密文的前 2～4 字节与密钥一起发送。在接收端,做同样的工作,如果接收端解密后的常数能与发端常数匹配,则传输无错。

4. 更新密钥

当密钥需要频繁地改变时,频繁进行新的密钥分发的确是困难的事,一种更容易的解决办法是从旧的密钥中产生新的密钥,有时称为密钥更新。可以使用单向函数进行更新密钥。如果双方共享同一密钥,并用同一个单向函数进行操作,就会得到相同的结果。

5. 存储密钥

密钥可以存储在人脑、磁条卡、智能卡中。也可以把密钥平分成两部分,一半存入终端,另一半存入 ROM。还可采用类似于密钥加密密钥的方法对难以记忆的密钥进行加密保存。

6. 备份密钥

密钥的备份可以采用密钥托管、密钥分割、密钥共享等方式。

最简单的方法是使用密钥托管中心。密钥托管要求所有用户将自己的密钥交给密钥托

管中心，由密钥托管中心备份保管密钥（如锁在某个地方的保险柜里或用主密钥对它们进行加密保存），一旦用户的密钥丢失（如用户遗忘了密钥或用户意外死亡），按照一定的规章制度，可从密钥托管中心索取该用户的密钥。另一个备份方案是用智能卡作为临时密钥托管。如 Alice 把密钥存入智能卡，当 Alice 不在时就把它交给 Bob，Bob 可以利用该卡进行 Alice 的工作，当 Alice 回来后，Bob 交还该卡，由于密钥存放在卡中，所以 Bob 不知道密钥是什么。

密钥分割把密钥分割成许多碎片，每一片本身并不代表什么，但把这些碎片放到一块，密钥就会重现出来。

一个更好的方法是采用一种密钥共享协议。将密钥 K 分成 n 块，每部分叫作它的“影子”，知道任意 m 个或更多的块就能够计算出密钥 K，知道任意 $m-1$ 个或更少的块都不能够计算出密钥 K，这叫作（m，n）门限（阈值）方案。目前，人们基于拉格朗日内插多项式法、射影几何、线性代数、孙子定理等提出了许多密钥共享方案。

拉格朗日插值多项式方案是一种易于理解的密钥共享（m，n）门限方案。密钥共享解决了两个问题：一是若密钥偶然或有意地被暴露，整个系统就易受攻击；二是若密钥丢失或损坏，系统中的所有信息就不能用了。

7. 销毁密钥

如果密钥必须替换，旧钥就必须销毁。密钥必须物理销毁。

加密密钥不能无限期使用，有以下几个原因：密钥使用时间越长，它泄露的机会就越大；如果密钥已泄露，那么密钥使用越久，损失就越大；密钥使用越久，人们花费精力破译它的诱惑力就越大，甚至采用穷举攻击法；对用同一密钥加密的多个密文进行密码分析一般比较容易。因此，不同密钥应有不同有效期。

数据密钥的有效期主要依赖数据的价值和给定时间里加密数据的数量。价值与数据传输率越大，所用的密钥更换越频繁。

密钥加密密钥无须频繁更换，因为它们只是偶尔用作密钥交换。在某些应用中，密钥加密密钥仅一月或一年更换一次。

用来加密保存数据文件的加密密钥不能经常地变换。通常每个文件用唯一的密钥加密，然后用密钥加密密钥把所有密钥加密，密钥加密密钥要么被记忆下来，要么保存在一个安全地点。当然，丢失该密钥意味着丢失所有的文件加密密钥。

公开密钥密码应用中的私钥的有效期是根据应用的不同而变化的。例如，用作数字签名和身份识别的私钥必须持续数年（甚至终身），用作抛掷硬币协议的私钥在协议完成之后就应该立即销毁。即使期望密钥的安全性持续终身，也要考虑两年更换一次密钥。旧密钥仍需保密，以防用户需要验证从前的签名。但是新密钥将用作新文件签名，以减少密码分析者所能攻击的签名文件数目。

3.4.2　密钥的组织结构

从信息安全的角度看，密钥的生存期越短，破译者的可乘之机就越少（从这个角度看，一次一密体制最安全）。因此，在实际使用中，尤其是在网络环境下，多采用层次化的密钥管理结构。用于数据加密的工作密钥平时不在加密设备中，需要时动态产生，并由其上层的密钥

加密密钥进行加密保护；密钥加密密钥可根据需要由其上一级的加密密钥进行保护，以此类推。层次的划分根据实际的使用环境确定，与环境的规模有关，规模越大，层次越多。最高层的密钥称为主密钥，它构成整个密钥管理系统的核心。在多层密钥管理系统中，通常下一层的密钥由上一层按照某种密钥协议来生成，因此掌握了主密钥，就有可能找出下层的各个密钥。

工作密钥常称为会话密钥，建立会话密钥的必要性在于以下四方面。

(1) 密钥重复使用容易导致泄露，因此应经常更换。

(2) 若使用相同的密钥，攻击者可将以前截获的信息插入当前的会话，而通信双方不一定能发现。

(3) 密钥一旦被破译，则使用这一密钥加密的信息都会失密，若使用会话密钥，则只有当前会话的信息失密。

(4) 如果对方不可靠，则更换会话密钥可防止对方以后窃取信息。

多层密钥体制极大增强了密码系统的安全性。由于用得最多的工作密钥经常更换，而高层密钥则用得较少，因此破译者可用的信息变得很少，增加了攻击的难度。

另外，多层密钥体制为自动化管理带来了方便，因为下层密钥可由计算机系统自动产生和维护，并通过网络自动分配和更换，减少了接触密钥的人数，也减轻了用户的负担。

在层次结构的框架下，可以对密钥作连通和分割定义。密钥的连通是指在用户之间共享密钥的范围；密钥的分割是指对这个范围的限制(细化)。按空间分割密钥，可区分不同的用户群，例如不同密级的数据之间的密钥分割，不同业务部门、业务系统之间的密钥分割，上下级机关之间的密钥分割，应用系统和管理系统之间的密钥分割，等等。

按时间分割密钥可实现让各个用户在不同的时期使用不同的密钥，使用户的使用权具有时间限制，分割的实现有两种方式。

(1) 静态分割：在给用户的加密设备注入密钥时就给定了用户的密码连通范围，即用户只能使用注入的密钥。

(2) 动态分割：密钥分配中心定期向规定范围内的用户加密传送一个用于控制分割范围的通播密钥(向指定用户广播的密钥)，即收到什么，使用什么。

3.4.3 密钥分配技术

密钥的分配技术解决的是网络环境中需要进行安全通信的端实体之间建立共享的对称密钥问题，在做法上可以由一方生成，再传递给另一方，或者由双方协作产生。如果已经存在一个安全上下文，例如已预先约定一个对称密钥序列，并通过安全渠道送达对方，则可以按约定使用并更换密钥，在这个序列使用完成之前不存在后续的密钥分配问题。或者在正常的情况下，原来的密钥仍然有效，则可以使用滚动的方法，用原有的密钥加密新的密钥并送达给对方，这个传送过程受旧密钥的保护，所以是安全的，密钥分配也不成问题。如果没有这样的安全上下文，就需要考虑如何安全可靠地分配密钥。

1. 密钥分配中心

密钥分配中心(KDC)是一种运行在物理安全服务器上的服务。采用 KDC 是当前的一种主流方式。每个节点或用户只需保管与 KDC 之间使用的加密密钥，而 KDC 为每个用户

保管一个互不相同的加密密钥。当两个用户需要通信时，需向 KDC 申请密钥。KDC 动态生成一个工作密钥(或称会话密钥)，用这两个用户的加密密钥分别加密后传送给这两个用户(通常是通过其中一个用户传送)。这种方式下，用户不必保存大量的工作密钥，而且可实现每次通信都使用不同的密钥；但缺点是通信量大，而且需要有较好的鉴别功能，以识别 KDC 和用户。

KDC 方式还可变形成电话号码本方式，适用于非对称密码体制。通过建立用户的公开密钥表，在密钥的连通范围内散发，也可采用目录方式动态去查询，后面介绍的公开密钥管理体制采用的就是这种思想。用户在进行保密通信前，首先产生一个工作密钥并使用对方的公开密钥加密传送，对方获悉此工作密钥后，使用对称密码体制与其进行保密通信。

使用 KDC 的优点是显而易见的，即通信方只需保存一个长期的同 KDC 进行通信的密钥和一个临时产生的短期会话密钥。当通信实体增加时，密钥的分发复杂度也极低。但 KDC 并未解决对称密码体制在最初的密钥共享的安全信道的问题(公钥密码体制解决了这个问题)，同时，密钥的集中管理分发决定了 KDC 安全的重要程度，一旦对 KDC 攻击得手，整个系统的安全通信便失去了保障。此外，大量的单点访问会使得 KDC 系统负载过大，而若尝试增加 KDC 数量，则又为整个系统增加了易受攻击的目标。

2. 智能卡方法

这个方法的基本思想是将 RSA 与 Diffie-Hellman 协议相结合，将通过 KDC 分配的密钥保存在智能卡中，这样平时使用密钥时不需要 KDC。整个系统有一个主密钥对 S_K/P_K，其中 P_K 是公钥，通信各方均可使用，S_K 是私钥，只有发卡的 KDC 使用。每个持卡的用户也拥有自己的一个密钥对，由系统的主密钥生成。需要通信时，使用智能卡产生一个工作密钥，具体做法如下。

(1) 磁卡发卡时，在对应的 RSA 体制下，找出整数 g 为 p 或 q 的本原元。对于用户 I，设定他的身份标识为整数 ID_i，计算

$$S_i = \mathrm{ID}_i S_K \bmod r \tag{3-12}$$

将整数组(r, g, P_K, S_i)存储在磁卡中发给用户，其中 S_i 需要保密，因为它是该用户的私钥，而其他数据是可公开的。

(2) 需要通信时，基于 Diffie-Hellman 协议，用户 I 和用户 J 要建立一个共同的工作密钥时，用户 I 产生一个随机数 R_i 并计算

$$X_i = S_i g^{R_i} (\bmod r) \tag{3-13}$$

将 X_i 连同自己的 ID_i 一起发给用户 J。同理，用户 J 也产生一个随机数 R_j 并计算

$$X_j = S_j g^{R_j} (\bmod r) \tag{3-14}$$

将 X_j 并连同自己的 ID_j 发给用户 I，然后双方分别计算出共享的工作密钥

$$K_i = (X_j^{P_K}/\mathrm{ID}_j)^{R_i} (\bmod r) = ((S_j g^{R_j})^{P_K}/\mathrm{ID}_j)^{R_i} (\bmod r) \tag{3-15}$$

$$K_j = (X_i^{P_K}/\mathrm{ID}_i)^{R_j} (\bmod r) = ((S_i g^{R_i})^{P_K}/\mathrm{ID}_i)^{R_j} (\bmod r) \tag{3-16}$$

因为 $S_i P_K/\mathrm{ID}_i (\bmod r) = \mathrm{ID}_i^{S_K P_K}/\mathrm{ID}_i (\bmod r) = 1$，所以 $K_j = K_i$。

(3) 需要验证身份时,利用 RSA 体制确认通信的对方拥有正确的 S_i。

用户 I 产生一个随机数 R_i,并计算 $X_i = g^{P_K R_i} \pmod r$;$Y_i = S_i g^{R_i} \pmod r$,并将 X_i、Y_i 和 ID_i 发给用户 J。

用户 J 也类似地产生一个随机数 R_j,并计算 $X_j = g^{P_K R_j} \pmod r$;$Y_j = S_j g^{R_j} \pmod r$,并将 X_j、Y_j 和 ID_j 发给用户 I。

因为在模 r 的情况下,有 $Y_i P_K / X_i = (S_i g^{R_i})^{P_K} / g^{P_K R_i} = S_i P_K = ID_i S_i P_K = ID_i$,所以用户 I 和 J 均可以用此方法验证对方的身份。

在智能卡方法中,整个系统的私钥只在发卡时使用,在智能卡的使用过程中并不涉及,因此它的生命周期不受智能卡生命周期的限制。

另外,Diffie-Hellman 密钥交换算法也是密钥分配的有效方法。

3.4.4 公钥基础设施

公钥体制主要是针对开放型的大型互连网络的应用环境而设计的,这种网络环境中需要有一个权威的、协调的公钥管理机制,以保证公钥的可靠性和可用性。公钥的管理一般基于可信的第三方机制,即需要一个通信的 A、B 双方都信任的第三方 N 来分别证明 A 和 B 的公钥的可靠性,这需要 N 分别对 A 和 B 的公钥进行数字签名,并构造一个证明该公钥可靠性的证书。从服务的可靠性和有效性角度出发,在一个大型网络中,这样的 N 应当有多个,称为公钥管理中心。如果这些公钥管理中心之间存在信赖关系,则用户可通过对应的信任链去设法验证任一个公钥管理中心签发的证书。体现公钥全局管理结构的公钥管理体制称为公钥基础设施(public key infrastructure,PKI)。

1. 数字证书及 X.509 标准

Diffie 和 Hellman 在 1976 年提出非对称密钥概念时,为了解决公钥的分配问题,他们对应提出了公众文件(public file)的概念,用一个中央数据库作为可信的第三方来保存公钥。为了解决这种中央数据库访问的性能问题,MIT 的 Kohnfelder 于 1978 年在他的本科毕业论文中提出了证书的概念,建议通过数字签名来保护命名证书(名字/密钥对),从而可将这些公钥分散存放和访问。然而直到 10 年之后,他的建议才变为现实,出现了 X.509 标准。

X.509 标准是目录服务标准 X.500 的一部分,对应的 ISO 标准是 ISO/IEC 9594-8。第一版发表于 1988 年,1993 年修订为第二版,并应用于 PEM(internet privacy enhanced mail,一种带安全功能的电子邮件)系统中。经过使用,X.509 于 1996 年 6 月改进为第三版。

为使用户在不可靠的网络环境中获得真实的公钥,PKI 引入了公认可信的第三方;同时为避免在线查询集中存放的公钥带来的性能瓶颈,PKI 引入了数字证书。数字证书是 PKI 的核心数据结构,它是数字世界中信任关系的载体,依赖证书上第三方的数字签名,用户可以离线地确认一个公钥的真实性。

PKI 的概念与数字证书是密不可分的。数字证书的形式有很多种,不同证书所携带的属性也不同。本小节要讨论的是以 X.509 证书为基础的 PKI,不涉及其他证书机制。作为一种网络基础设施,PKI 以证书为手段保证公钥的可靠分发,但如何使用公钥进行安全通

信，如交换会话密钥、构造数字信封等，并不在 PKI 讨论的范围内。也就是说，除了 PKI 所提供的服务，应用程序应有另外的协议机制进行基于公钥的安全交互。

2. PKI 的结构模型

PKI 是公钥的一种管理机制，宏观上呈现为管理域结构。每个管理域都有一定的覆盖范围，通过交叉证书所表达的信任关系相互关联，构成更大的管理域，并最终形成全局性的公钥管理体系。不同的管理域对公钥的使用具有不同的要求，通过政策来表达，从而具有相对的独立性。对于互联网而言，可以存在多个 PKI，每个 PKI 都包含政策审批中心（policy approval authority，PAA）、证书管理中心（certificate authority，CA）、单位注册中心（organizational registry authority，ORA）等功能，这些功能的数量随管理域的规模而变化。由于对证书的使用要附加特定的政策，因此不同的应用要使用不同的公钥证书。

（1）PAA。PAA 负责制定 PKI 的全局安全政策和所有下级机构都需遵循的规章制度，主要是证书政策（certificate policies）和证书使用规定（certification practice statements）。证书政策是一组规定，指出一个证书对一组特定用户和/或应用的可适用性，表明它对于一个特定的应用和目的是否是可用的。证书使用规定比证书政策更详细，它综合描述了 CA 对证书政策的各项要求的实现方法。

证书政策是信息管理和信息基础设施的一个组成部分。通过定义恰当的证书政策，可以使这个基础设施能够从整体上安全地实现开放环境中的服务提供、管理和通信；用电子形式的认证替代书面形式的认证，从而可加快信息流通速度，提高效率，降低成本，促进开放标准技术的使用，并有助于建立与其他开放安全环境的互操作。

（2）CA。CA 是可信任的第三方，负责具体的证书颁发和管理。公钥证书的含义就是用户实体和公钥之间一个符合证书政策的绑定。交叉证书实际上意味着一个 CA 对另一个 CA 的证书政策的承认，从而可以进行跨越 CA 管理域的认证。注意，每个交叉证书只是证明了一个 CA 对另一个 CA 的信任关系，反之不然。CA 既包括基本的证书发放和管理功能，还可包括一些辅助功能。它可以具有层次结构，在自身直接管理一些具体的证书之外，还可管理一些下级证书管理中心。PAA 通常也作为根证书管理中心（root CA），它向下一级证书管理中心发放证书。

（3）ORA。ORA 有时也简称为 RA，银行的营业网点可以帮助远离 CA 的端实体在 CA 处注册并获得证书。ORA 可以视作 CA 的代理，但本身并没有 CA 的功能。

（4）端实体。端实体是结构模型中的最底层，是使用证书的端系统或应用的抽象。

通常 PKI 的最高管理是通过一个政策管理中心（policy management authority，PMA）来体现的。这个机构主要用来对 PKI 进行宏观管理，其作用有点像公司的董事会，而 PAA 则像公司的行政班子。

3. PKI 的操作模型

PKI 结构模型中有两种管理实体：CA 和 RA。CA 是 PKI 框架中唯一能够发布和撤销证书的实体，它维护着证书的生存周期；RA 负责处理用户请求，在验证了请求的有效性后，代替用户向 CA 提交。严格地说，CA 与 RA 之间是存在区别的，CA 是一个组织的注册机构，而 RA 没有这个限制，它表示的范围更广。RA 可以单独实现，也可以合并在 CA 中实

现。作为管理实体,CA/RA 以证书方式向端实体提供公钥的分发服务。

PKI 框架中有两种端实体:持证者(holder)和验证者(verifier)。持证者是证书的拥有者,是证书所声明事实的主体。持证者向管理实体申请并获得证书,也可以在需要时请求撤销或更新证书。持证者可以借助证书来认证自己的身份,从而获取相应的权力。验证者通常是授权方,它需要确认持证者所提供的证书的有效性以及对方是否为该证书的真正拥有者,只有在成功认证之后才会授予对方相应的权力。

实体的划分不是绝对的,有些实体往往同时兼有其他实体的功能,例如 RA 需要使用证书向 CA 鉴别自己,这时它就是持证者;大多数持证者同时也是验证者,但一些验证者可能并不是持证者。

4. PKI 的实现模型

从前面的叙述可以看出,完整地实现 PKI 服务是一个非常复杂的问题,不仅要考虑技术层面的因素,还要考虑管理和法律等方面的因素,实现的复杂性大大限制了 PKI 服务的推广普及,尽管发达国家从 20 世纪 90 年代中后期就开始了这方面的实践尝试,也有相应的商业服务推出,但距商业成功还有很大的距离。

鉴于在商业领域实现 PKI 服务的困难性,在教育科研领域进行实验实现是一个可行的选择。美国的 Internet2 曾就 PKI 在教育网的应用推广进行了广泛的研究和探讨,建立了相应的研究委员会,并提出了自己的实现思路,颇有参考价值。按照 Internet2 的思路,PKI 的实现可以有以下几种模式。

(1) PKI 全集。这是完整的 PKI 实现,提供多用途,广泛覆盖多领域,桥接层次型 CA 结构的证书服务。多用途指的是可提供不同安全等级的证书。

(2) PKI 简单全集。这是针对单一领域的多用途 PKI 服务,具有统一的证书政策(CP)和层次型 CA 结构,例如专门面向教育领域。

(3) 轻型 PKI。这类实现包含 PKI 的所有关键部分,例如 CA 和 RA 等,但在功能实现上进行简化,例如只提供低安全等级的证书或支持的密码体制有限,因此仅限于一些特定的应用使用,例如只支持对 Web 访问的身份认证。

(4) 超轻型 PKI。该类简化 PKI 功能的实现,例如只实现 CA 功能,不提供 RA 服务。这类 PKI 实现往往是研究小组的成果,用于科研实验目的,起宣传示范作用。简化的方面包括 PKI 的保障措施、支持的签名算法、支持的应用范围、证书撤销列表(CRL)的功能、主体命名方式、对移动性的支持等。

3.5 扩展阅读:国产密码算法

国产密码算法简称国密,即国家密码局认定的国家商用密码算法。它是由国家密码管理局认定和公布的密码算法标准及其应用规范,是指用于商业的、不涉及国家秘密的密码技术,其中部分密码算法已经成为国际标准。

3.5.1 国密基础

密码算法是国之重器,是各国之间战略竞争的制高点,必须掌握自主密码技术,才能够

赢得安全和发展的主动权。自 20 世纪末发布《商用密码管理条例》以来，我国的密码研究和密码应用水平快速发展。目前，我国已发布了 40 多项密码标准，公布了 SM2、SM3、SM4 等系列密码算法，形成了较为完整的密码标准体系。国产的祖冲之密码算法已被采纳为第四代移动通信国际密码标准，是我国第一个成为国际标准的密码算法。

2015 年 3 月，中央办公厅、国务院办公厅联合印发了《关于加强重要领域国产密码应用的指导意见》，进一步推动了我国国产自主密码算法及相关产品在重要领域的积极应用。文件明确了六项主要任务：一是在推进基础信息网络密码应用方面，提出电信网、广播电视网、互联网等基础信息网络，要将国产密码应用纳入信息化建设整体规划。二是在规范重要信息系统密码应用方面，提出了能源、教育、公安、社保、测绘地理信息、环保、交通、卫生计生、金融等涉及国计民生和基础信息资源的重要信息系统，要将国产密码应用纳入信息化建设整体规划。统筹推动金融服务相关领域的国产密码应用标准建设，建立健全基于国产密码的身份认证、访问控制、数据保护、可信服务、安全审计等安全防护措施。三是在促进重要工业控制系统密码应用方面，提出了核设施、航空航天、先进制造、石油石化、油气管网、电力系统、交通运输、水利枢纽、城市设施等重要工业控制系统要将国产密码应用纳入信息化建设整体规划，实现国产密码在数据采集与监控、分布式控制系统、过程控制系统、可编程逻辑控制器等工业控制系统中的深度应用。四是在面向社会服务的政务信息系统密码应用方面，要求党政机关和使用财政性资金的事业单位、团体组织使用的面向社会服务的信息系统，要加快推进基于国产密码的网络信任、安全管理和运行监管体系的建设。五是在提升密码基础支撑能力方面，要求建设完善密码基础技术、应用技术、标准规范和检测评估体系。加强面向云计算、物联网、大数据、移动互联网和智慧城市等新方向的密码应用技术研究，完善身份认证、授权管理、责任认定、可信时间、电子签章等密码基础设施，科学分布密码应用系统集成、运营、监理等服务机构。六是在建立健全密码应用安全性评估审查制度方面，提出要加强密码检测能力建设，全面提升密码产品和系统检测效能，健全密码检测认证体系。

为适应我国国家安全面临的新形势和密码广泛应用带来的新挑战，有必要制定一部密码领域综合性、基础性法律。为此，国务院立法工作计划将密码法确定为全面深化改革急需项目。按照国家密码立法工作部署，2014 年 12 月，国家密码管理局成立起草小组，着手密码法起草工作，在国家有关部门的大力支持和帮助下，起草小组深入研究新形势下密码事业发展面临的新形势、新任务，认真总结我国密码工作中形成的好经验、好做法，积极参考借鉴国外做法，广泛深入调研，系统梳理我国密码事业发展中存在的突出问题，认真研究对策措施。在此基础上，制定完成了《中华人民共和国密码法(草案征求意见稿)》，并于 2017 年 4 月向社会公开征求意见。2019 年 10 月 26 日，十三届全国人大常委会第十四次会议审议通过《中华人民共和国密码法》，自 2020 年 1 月 1 日起施行。

密码法是总体国家安全观框架下，国家安全法律体系的重要组成部分，其颁布实施将极大提升密码工作的科学化、规范化、法治化水平，有力促进密码技术进步、产业发展和规范应用，切实维护国家安全、社会公共利益以及公民、法人和其他组织的合法权益。这表明，我国在构建与国家治理体系和治理能力现代化相适应的密码法律制度体系的道路上迈出重要一步，将为确保密码使用优质高效，确保密码管理安全可靠提供坚实的法治保障。

3.5.2 国密系列算法

1. SM1 算法

SM1 算法是对称加密算法,其加密强度与 AES 相当。该算法不公开,仅以 IP 核的形式存在于芯片中,需要通过加密芯片的接口进行调用。

2. SM2 算法

SM2 算法是非对称算法,其实现基于 ECC 算法。SM2 椭圆曲线密码算法是我国自主设计的公钥密码算法,包括 SM2-1 椭圆曲线数字签名算法、SM2-2 椭圆曲线密钥交换协议、SM2-3 椭圆曲线公钥加密算法,分别用于实现数字签名密钥协商和数据加密等功能。SM2 算法与 RSA 算法不同的是,SM2 算法是基于椭圆曲线上点群离散对数难题,相对于 RSA 算法,256 位的 SM2 密码强度已经比 2048 位的 RSA 密码强度要高。SM2 以其高安全性和运算快速的特点在数据安全领域应用越来越广泛。

3. SM3 算法

SM3 算法为摘要算法,可以用 MD5 作为对比理解。它的校验结果为 256 位,适用于商用密码应用中的数字签名和验证消息认证码的生成与验证以及随机数的生成,可满足多种密码应用的安全需求。

4. SM4 算法

SM4 算法是无线局域网标准的分组数据算法。对称加密,密钥长度和分组长度均为 128 位。由于 SM1、SM4 加解密的分组大小为 128 比特,故对消息进行加解密时,若消息长度过长,需要进行分组;若消息长度不足,则要进行填充。

5. SM7 算法

SM7 算法是一种分组密码算法,该算法没有公开,分组长度为 128 比特,密钥长度为 128 比特。SM7 适用于非接触式 IC 卡,应用包括身份识别类应用(门禁卡、工作证、参赛证)、票务类应用(赛事门票、展会门票)、支付与通卡类应用(校园一卡通、企业一卡通等)。

6. SM9 算法

SM9 算法是一种基于椭圆曲线的公钥密码体制,由中国密码学家于 2016 年提出。目前,在全球范围内尚未出现针对椭圆曲线密码的有效破解方法。因此,SM9 算法具有较高的安全性能。传统的公钥密码体制需要使用数字证书来验证公钥的合法性,而 SM9 算法直接使用用户标识作为公钥,省去了数字证书的颁发和验证过程,降低了系统的复杂性和成本。相较于传统的公钥密码体制,SM9 算法在密钥生成和加密过程中具有更高的计算效率,且不需要复杂的证书链验证。这使得 SM9 算法特别适合应用于计算资源受限的场景。SM9 算法具有良好的兼容性和可扩展性,可以与现有的加密技术和安全协议无缝集成。同时,SM9 算法支持多种应用场景,如安全通信、数字签名、身份认证等。

7. 祖冲之密码算法

祖冲之密码算法是一种流加密算法，它是中国自主研究的流密码算法，该机密性算法可适用于 3GPP LTE 通信中的加密和解密，该算法包括祖冲之算法（ZUC 算法）、加密算法（128-EEA3）和完整性算法（128-EIA3）三部分。目前已有对 ZUC 算法的优化实现，有专门针对 128-EEA3 和 128-EIA3 的硬件实现与优化。祖冲之密码算法由国家密码管理局于 2012 年 3 月 21 日发布。

3.5.3　国密展望

伴随云计算技术、物联网、大数据分析与数据挖掘、人工智能等先进信息技术的快速发展，特别是“互联网＋”的推广，信息安全性成为现代信息技术生产和信息安全服务的基础要求，数据加密逐渐成为不可或缺的重要手段，密码应用领域将不断深入和拓展。

1. 应用拓展

随着商用密码领域的不断发展，商用密码产业将会形成一条完整的、可控制的产业链和良好的生态系统，整个产业的综合能力将会得到极大提高，从而形成一批商用密码领域的龙头企业，对中国商用加密产业产生重要的影响和促进作用。

2. 性能提升

加大技术研发力度，在新型密码算法、量子密钥、生物密钥等方面突破一系列重要的技术难题；在可信计算、区块链等新技术研究，在云计算、大数据、密码管理等方面，在生物密码管理技术与生物特征数据结合方面，在核心技术领域，如网络管理与应用技术、网络身份管理技术等方面都达到世界领先水平。

3. 标准体系完善

加速发展若干基础共性技术、应用技术和技术标准，为我国关键技术应用提供强有力的支持。全面建设科学、先进的密码标准和检测制度，建立完善的密码评价规范与认证体制机制，推动国密算法、产品、服务的高质量发展。

4. 确保政策落地

政府要加强宣传和监管，确保系列国产密码鼓励政策真正落地。当前，我国已经出台了一系列相关政策法规，通过简化流程、税收优惠等手段规范和推广国产密码应用。但是在实际应用中存在政策执行不力、监管不到位、细则规范不明晰等问题，因此，要积极制定可操作的细则，同时加强政策宣传，使国产密码政策真正落地。

本章小结

数据加密技术为数据信息在网络中的传输提供了有力的技术支撑，可以说，数据加密技术的价值在网络空间安全的各领域中是不可估量的。数据加密技术不仅有力保障了网络用

户的个人信息安全,同时也保障了电子商务等行业的交易过程和交易信息的安全,同样也为数据信息的安全存储提供了有力保障。

本章首先介绍了数据加密技术的基础知识,包括基本概念、密码学发展历史、密码攻击、数据加密基本方式等。接着详细介绍了传统(对称)密码体制和公钥密码体制,重点讲述了DES、AES、SM4、祖冲之系列密码、RSA、ElGamal、SM2 椭圆曲线密码等加密算法。然后介绍了密钥管理的相关知识,其中重点讲述了密钥分配技术。最后扩展阅读介绍了国密算法,包括国密算法的发展与展望。