第5章 密钥分配与管理

密钥的分配与管理技术所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密钥的问题，最简单的解决办法就是预先约定一个对称密钥序列并通过安全的渠道送达对方，以后按约定使用。如果密钥用量较大，且更换频繁，则密钥的传递就会成为严重的负担，而多数用户之间可能并没有安全的传输渠道，因此就需要研究在不安全的通信信道中传递密钥的方法。本章首先介绍单钥和公钥加密体制的密钥分配技术，然后介绍密钥托管及秘密分割技术，最后介绍常用的消息认证技术。

5.1 单钥加密体制的密钥分配

5.1.1 密钥分配的基本方法

两个用户在用单钥密码体制进行保密通信时，必须有一个共享的秘密密钥，而且为防止攻击者得到密钥，还必须经常更新密钥。因此，密码系统的强度也依赖于密钥分配技术。用户 A 和 B 获得共享密钥的方法基本上有以下几种。

① 密钥由 A 选取并通过物理手段发送给 B，如图 5-1 所示。

② 密钥由第三方选取并通过物理手段发送给 A 和 B，如图 5-2 所示。

图 5-1 第一种方法

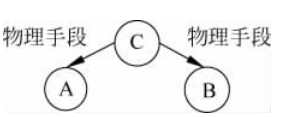

图 5-2 第二种方法

③ 如果 A、B 事先已有一密钥，则其中一方选取新密钥后，用已有的密钥加密新密钥并发送给另一方，如图 5-3 所示。

④ 如果 A 和 B 与第三方 C 分别有一保密信道，则 C 为 A、B 选取密钥后，分别在两个保密信道上发送给 A、B，如图 5-4 所示。

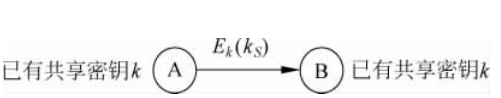

图 5-3 第三种方法

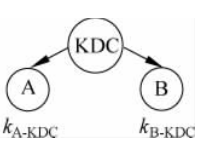

图 5-4 第四种方法

前两种方法称为人工发送，密钥的人工发送在网络的链路加密时还是可行的，因为只有该链路上的两端交换数据。而密钥的人工发送在网络的端-端加密方式中将不再可行，因为若加密是在网络层，则网络中任一对主机都必须有一共享密钥。如果有 n 台主机，则密钥数目为 $n(n-1)/2$。当 n 很大时，密钥分配的代价非常大。

第三种方法对链路加密和端-端加密方式都是可行的，但是攻击者一旦获得一个密钥就可获取以后的所有密钥。其初始密钥的分配代价仍然很大。

第四种方法广泛用于端-端加密方式时的密钥分配，其中的第三方通常是一个负责为用户分配密钥的密钥中心(Key Distribution Center，KDC)。每个用户必须和 KDC 有一个共享密钥，称为主密钥。通过主密钥分配给一对用户的密钥称为会话密钥，用于这一对用户之间的保密通信。通信完成后，会话密钥即刻被销毁。若用户数为 n 个，则会话密钥数为 $n(n-1)/2$。但主密钥数却只需 n 个，则可通过物理手段发送主密钥。

5.1.2 密钥分配的一个实例

如图 5-5 所示，假定两个用户 A、B 分别与密钥分配中心有一个共享的主密钥 k_A 和 k_B，A 希望与 B 建立一个共享的一次性会话密钥，可通过以下几步来完成：

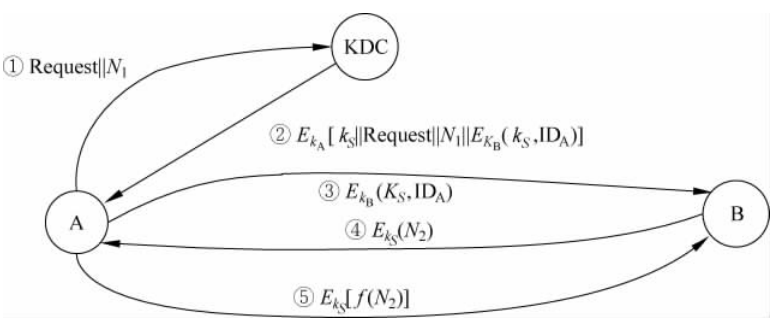

图 5-5 密钥分配实例

① A 向 KDC 发出会话密钥请求。

表示请求的消息由两个数据项组成：第一项 Request 是 A 和 B 的身份，第二项是这次业务的唯一识别符 N_1，称 N_1 为一次性随机数，可以是时间戳、计数器或随机数。每次的 N_1 都应不同，且为防止假冒，应使敌手对 N_1 难以猜测。因此用随机数作为这个识别符最为合适。

② KDC 为 A 的请求发出应答。

应答由 k_A 加密，只有 A 能成功解密，且 A 可相信这一消息的确是由 KDC 发出的。

消息中包括 A 希望得到的两项内容：

a. 一次性会话密钥 K_S。

b. A 在①中发出的请求，包括一次性随机数 N_1，目的是使 A 将收到的应答与发出的请求相比较，确定是否匹配。

- A 能验证自己发出的请求在被 KDC 收到之前，是否被他人篡改。
- A 还能根据一次性随机数相信收到的应答不是重放的过去的应答。

消息中还有 B 希望得到的两项内容：

a. 一次性会话密钥 k_S。

b. A 的身份(例如 A 的网络地址)ID_A。

- 这两项由 k_B 加密，将由 A 转发给 B，以建立 A、B 之间的连接。
- 并用于向 B 证明 A 的身份。

③ A 存储会话密钥 k_S，并向 B 转发 $E_{k_B}[k_S \| ID_A]$。

a. 因为转发的是由 k_B 加密后的密文，所以转发过程不会被窃听。

b. B 收到后，可得到会话密钥 k_S，并由 ID_A 可知另一方是 A，而且还从 E_{k_B} 知道 k_S 的确来自 KDC。

c. 这一步完成后，会话密钥就安全地分配给了 A、B。

然而还能继续以下两步工作：

④ B 用会话密钥 k_S 加密另一个一次性随机数 N_2，并将加密结果发送给 A。

⑤ A 以 $f(N_2)$作为对 B 的应答，其中 f 是对 N_2 进行某种变换(例如加 1)的函数，并将应答用会话密钥加密后发送给 B。

这两步可使 B 相信第③步收到的消息不是一个重放。

注意：第③步就已完成密钥分配，第④、⑤两步结合第③步执行的是认证功能。

5.2 公钥加密体制的密钥管理

接下来，从两个方面介绍公钥加密体制下的密钥分配：①公钥加密体制所使用的公开密钥的分配；②用公钥体制来分配单钥加密体制所需的密钥。

5.2.1 公钥的分配

公钥加密的一个主要用途是分配单钥密码体制使用的密钥。公钥密码体制所用的公开密钥的分配方法如下：

1. 公开发布

用户将自己的公钥发给每一个其他用户或向某一团体广播。这种方法虽简单却使任何人都可伪造这种公开发布。假冒者可解读发向被伪造方的加密消息，还可用伪造的密钥获得认证。

2. 公用目录表

公用目录表是指建立一个公用的公钥动态目录表，由某个可信的实体或组织承担目录表的建立、维护以及公钥的分布。这种方法比前一种安全性更高，但仍然容易受到攻击。

3. 公钥管理机构

公钥管理机构是在公钥目录表中对公钥的分配施加更严密的控制，使其安全性更强。

公钥管理机构为各用户建立、维护动态公钥目录。同时每个用户都可靠地知道管理机构的公钥，而只有管理机构自己知道相应的秘密钥。公钥的分配如图 5-6 所示。

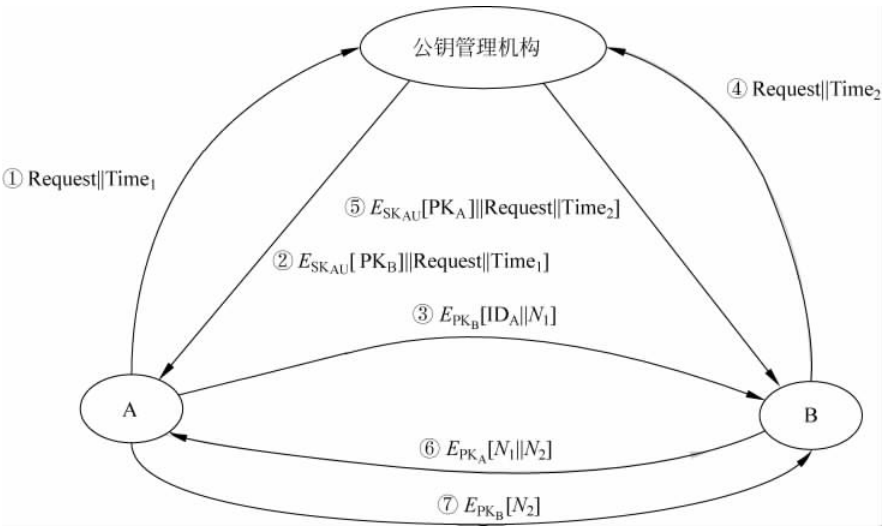

图 5-6　公钥管理机构分配公钥

① 用户 A 向公钥管理机构发送一带有时间戳的消息，请求获取用户 B 当前的公钥。

② 管理机构用自己的秘密钥 SK_{AU} 加密对 A 的请求做出应答。A 可以用管理机构的公开钥解密，并使 A 相信这个消息的确是来源于管理机构。其应答的消息有以下作用。

a. B 的公钥 PK_B，A 可用其对将要发往 B 的消息加密。

b. A 验证自己最初发出的请求在被管理机构收到以前未被篡改。

c. 时间戳使 A 相信管理机构发来的消息是 B 当前的公钥。

③ A 用 B 的公开钥对一个消息加密后发送给 B，其中一项是 A 的身份 ID_A，另一项是一个一次性随机数 N_1，用于唯一地标识本次业务。

④、⑤B 以相同的方式从管理机构获取 A 的公开钥。此时，A 和 B 都已安全地得到了对方的公钥，但仍需要进一步的相互认证。

⑥ B 用 PK_A 对一个消息加密后发送给 A，其消息有 A 的一次性随机数 N_1 和 B 产生的一个新的一次性随机数 N_2。因为只有 B 能解密③的消息，所以 A 收到的消息中的 N_1 可使其相信通信的另一方的确是 B。

⑦ A 用 B 的公钥对 N_2 加密后返回给 B，可使 B 相信通信的另一方的确是 A。

以上 7 个消息中的前 4 个消息是用于获取对方的公钥。当用户得到对方的公钥后保存，使之以后使用时只发送⑥、⑦确认消息即可。但还必须定期地通过密钥管理机构中心获取通信对方的公钥，以免对方的公钥更新后无法保证当前的通信。

公钥管理机构方式的优缺点：

① 每次密钥的获得由公钥管理机构查询并认证发送，用户不需要查表，提高了安全性。

② 公钥管理机构必须一直在线，由于每一用户要想和他人联系都需求助于管理机

构，所以管理机构有可能成为系统的瓶颈。

③ 由管理机构维护的公钥目录表易被敌手通过一定方式窜扰。

4. 公钥证书

公钥分配的另一种方法是公钥证书，用户通过公钥证书相互交换自己的公钥而无须与公钥管理机构联系。公钥证书由证书管理机构(Certificate Authority，CA)为用户建立，其证书的数据项有用户的公钥、身份和时间戳等，这些数据项经 CA 用自己的秘密钥签字后形成证书，其形式为 $CA=E_{SK_{CA}}[T,ID_A,PK_A]$，其中 ID_A 是用户 A 的身份，PK_A 是 A 的公钥，T 是当前时间戳，SK_{CA} 是证书管理机构的秘密钥，CA 即为用户 A 产生的证书，如图 5-7 所示。

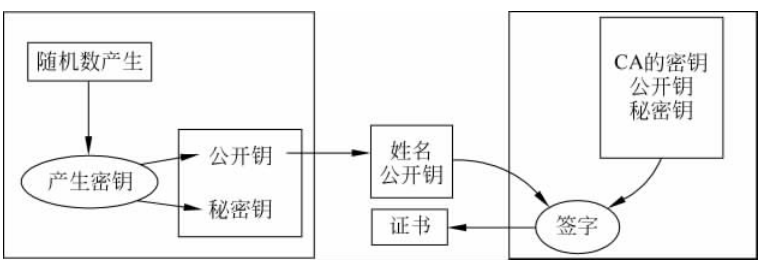

图 5-7　证书的产生过程

用户可将自己的公开钥通过公钥证书发给另一用户，接收方可用证书管理机构的公钥 PK_{CA} 对证书加以验证，即 $D_{PK_{CA}}[CA]=D_{PK_{CA}}[E_{SK_{CA}}[T,ID_A,PK_A]]=(T,ID_A,PK_A)$，因为只有用证书管理机构的公钥才能解读证书，同时获得了发送方的 ID_A 和公开钥 PK_A。时间戳用于鉴定收到的证书是否是当前的或有效的。

5.2.2　用公钥加密分配单钥密码体制的密钥

公开钥分配完成后，用户就可用公钥加密体制进行保密通信。然而由于公钥加密的速度过慢，以此进行保密通信不太合适，但用于分配单钥密码体制的密钥却非常合适。

1. 简单分配

图 5-8 简单描述了使用公钥加密算法建立会话密钥的过程，如果 A 希望与 B 通信，可通过以下几步建立会话密钥：

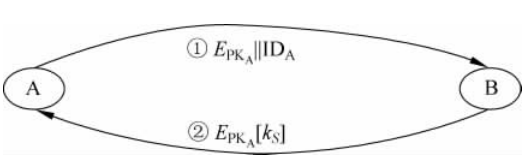

图 5-8　简单使用公钥加密算法建立会话密钥

① A 产生自己的一对密钥$\{PK_A,SK_A\}$，并向 B 发送 $PK_A \| ID_A$，其中 ID_A 表示 A 的身份。

② B 产生会话密钥 k_S，并用 A 的公开钥 PK_A 对 k_S 加密后发往 A。

③ A 由 $D_{SK_A}[E_{PK_A}[k_S]]$恢复会话密钥。因为只有 A 能解读 k_S，所以仅 A、B 知道这一共享密钥。

④ A 销毁$\{PK_A, SK_A\}$，B 销毁 PK_A。

经过以上的步骤后，A、B 就可以用单钥加密算法以 k_S 作为会话密钥进行保密通信，通信完成后，又都将 k_S 销毁。这种分配法虽然简单，但却由于 A、B 双方在通信前和完成通信后都没有存储密钥，因此密钥泄露的危险性最小，还可以防止双方的通信被攻击者监听。用公钥加密算法建立会话密钥这种协议容易受到攻击，若攻击者 E 已接入 A、B 双方的通信信道，就可通过以下不被察觉的方式截获双方的通信：

① 与上面的步骤①相同。

② E 截获 A 的发送后，建立自己的一对密钥$\{PK_E, SK_E\}$，并将 $PK_E \| ID_A$ 发送给 B。

③ B 产生会话密钥 k_S 后，将 $E_{PK_E}[k_S]$发送出去。

④ E 截获 B 发送的消息后，由 $D_{PK_E}[E_{PK_E}[k_S]]$解读 k_S。

⑤ E 再将 $E_{PK_A}[k_S]$发往 A。

现在 A 和 B 知道 k_S，但并未意识到 k_S 已被 E 截获。A、B 在用 k_S 通信时，E 就可以实施监听。

2. 具有保密性和认证性的密钥分配

此种密钥分配过程具有保密性和认证性，因此既可防止被动攻击，又可防止主动攻击，如图 5-9 所示。假定 A、B 双方已完成公钥交换，可按以下步骤建立共享会话密钥：

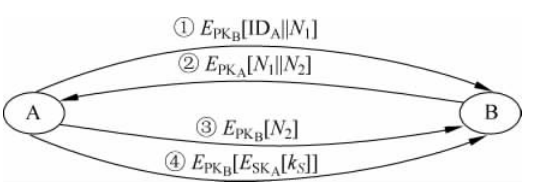

图 5-9　具有保密性和认证性的密钥分配

① A 用 PK_B 的公开钥加密 A 的身份 ID_A 和一个一次性随机数 N_1 后发往 B，其中 N_1 用于唯一地标识这一业务。

② B 用 PK_A 加密 N_1 和 B 新产生的一次性随机数 N_2 后发往 A。B 发来的消息中 N_1 的存在可使 A 相信对方的确是 B。

③ A 用 PK_B 对 N_2 加密后返回给 B，使 B 相信对方的确是 A。

④ A 选一会话密钥 k_S，然后将 $M=E_{PK_B}[E_{SK_A}[k_S]]$发给 B，其中用 B 的公开钥加密是为保证只有 B 能解读加密结果，用 A 的秘密钥加密是保证该加密结果只有 A 能发送。

⑤ B 以 $D_{PK_A}[D_{SK_B}[M]]$恢复会话密钥。

注意：这个方案其实是有漏洞的，即第 4 条消息容易被重放，假设敌手知道上次通话时协商的会话密钥 k_S，以及 A 对 k_S 的签名和加密，则通过简单的重放即可实现对 B 的欺骗，解决的方法是将第 3 和第 4 条消息合并发送。

方案修改后的过程如图 5-10 所示。

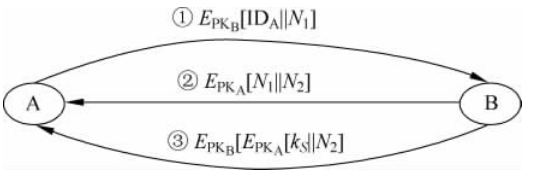

图 5-10　具有保密性和认证性的密钥分配

5.2.3　密钥管理的一个实例

DH 密钥交换算法是 W. Diffie 和 M. Hellman 于 1976 年提出的第一个公钥密码算法，已在很多商业产品中得以应用。

① 算法的唯一目的是使得两个用户能够安全地交换密钥，得到一个共享的会话密钥，算法本身不能用于加、解密。

② 算法的安全性基于求离散对数的困难性。

算法过程如图 5-11 所示，其中 p 是大素数，a 是 p 的本原根，p 和 a 作为公开的全程元素。

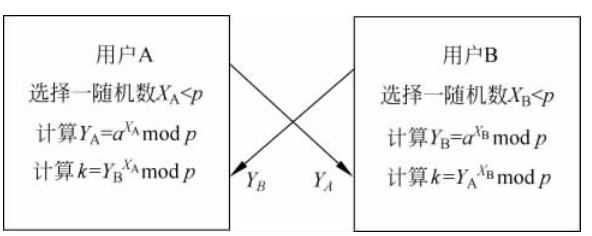

图 5-11　DH 密钥交换过程

用户 A 选择一保密的随机整数 X_A，并将 $Y_A=a^{X_A} \bmod p$ 发送给用户 B。类似地，用户 B 选择一保密的随机整数 X_B，并将 $Y_B=a^{X_B} \bmod p$ 发送给用户 A。然后 A 和 B 分别由 $k=Y_B^{X_A} \bmod p$ 和 $k=Y_A^{X_B} \bmod p$ 计算出共享密钥，这是因为：

$$\begin{aligned} Y_B^{X_A} \bmod p &= (a^{X_B} \bmod p)^{X_A} \bmod p = (a^{X_B})^{X_A} \bmod p \\ &= a^{X_B X_A} \bmod p = a^{X_A X_B} \bmod p = (a^{X_A} \bmod p)^{X_B} \bmod p \\ &= Y_A^{X_B} \bmod p \end{aligned}$$

因 X_A、X_B 是保密的，敌手只能得到 p、a、Y_A、Y_B，要想得到 k，则必须得到 X_A、X_B 中的一个，这意味着需求离散对数。因此敌手求 k 是不可行的。

例 5-1　DH 密钥交换算法示例。$p=97$，$a=5$，A 和 B 分别秘密地选 $X_A=36$，$X_B=58$，并分别计算

$$Y_A=5^{36} \bmod 97=50, Y_B=5^{58} \bmod 97=44$$

在交换 Y_A、Y_B 后，分别计算

$$k = Y_B^{X_A} \bmod p = 44^{36} \bmod 97 = 75$$
$$k = Y_A^{X_B} \bmod p = 50^{58} \bmod 97 = 75$$

5.3 密钥托管

5.3.1 密钥托管的背景

加密技术的快速发展对保密通信和电子商务起到了良好的推动作用。但是，也使得政府法律职能部门难以跟踪、截获犯罪嫌疑人员的通信，特别是在广泛采用公开密钥技术后，随之而来的是公开密钥的管理问题。对于中央政府来说，为了加强对贸易活动的监管，客观上也需要银行、海关、税务、工商等管理部门紧密协作。为了打击犯罪，还要涉及公安和国家安全部门。这样，交易方与密钥管理机构就不可避免地产生联系。为了监视和防止计算机犯罪活动，人们提出了密钥托管的概念。密钥托管与 CA 相结合，既能保证个人通信与电子交易的安全性，又能实现政法职能部门的管理介入，是今后信息安全策略的发展方向。

密钥托管(Key Escrow，KE)提供了一种密钥备份与恢复的途径，也称托管加密(Key Encryption)。从字面上理解，密钥托管就是指把通信双方的会话密钥交由合法的第三方，以便让合法的第三方利用得到的会话密钥解密双方通信的内容，从而监视双方的通信。这里所说的合法的第三方就是密钥托管的机构，一般是指政府保密部门和法律执行部门。其目的是保证对个人没有绝对的隐私和绝对不可跟踪的匿名性，即在强加密中结合对突发事件的解密能力。更确切地说，密钥托管是指为公众和用户提供更好的安全通信的同时，也允许授权者(包括政府保密部门、法律执行部门或有契约的私人组织等)为了国家、集团和个人隐私等安全利益，监听某些通信内容和解密有关密文。同时，这种技术还可以为用户提供一个备用的解密途径。所以，密钥托管也有“密钥恢复”(Key Recovery)、“数据恢复”和“特殊获取”等含义。这种技术产生的出发点是政府机构希望在需要时可通过密钥托管技术解密用户的一些特定的信息，此外当用户的密钥丢失或损坏时也可通过密钥托管技术恢复出自己的密钥。所以这个备用的手段不仅对政府部门有用，对用户自己也有用，为此，许多国家都制定了相关的法律法规。美国政府 1993 年 4 月颁布了 EES(Escrowed Encryption Standard，托管加密标准)，该标准体现了一种新思想，即对密钥实行法定托管代理的机制。该标准使用的托管加密技术不仅提供了加密功能，同时也使政府可以在实施法律许可下的监听(如果向法院提供的证据表明，密码使用者是利用密码在进行危及国家安全和违反法律规定的事，经过法院许可，政府可以从托管代理机构取来密钥参数，经过合成运算，就可以直接侦听通信)。美国政府希望用这种办法加强政府对密码使用的调控管理。

目前，在美国有许多组织都参加了密钥托管标准(KES)和 EES 的开发工作，系统的开发者是司法部门(DOJ)，NIST 和基金自动化系统分部对初始的托管(Escrow)代理都进行了研究，国家安全局(NSA)负责 KES 产品的生产，联邦调查局(FBI)被指定为最初的合法性强制用户。

自从密钥托管出现以来，特别是美国政府的 EES 公布之后，在社会上引起了极大的

反响，很多人对此项技术颇有争议。有关密钥托管争论的主要焦点在于以下两方：一方认为，政府对密钥管理控制的重要性是出于安全考虑，这样可以允许合法的机构依据适当的法律授权访问该托管密钥，不但政府通过法律授权可以访问加密过的文件和通信，用户在紧急情况时，也可以对解密数据的密钥恢复访问；另一方认为，密钥托管技术侵犯个人隐私，在他们看来密钥托管政策把公民的个人隐私置于政府情报部门手中，一方面违反了美国宪法和个人隐私法，另一方面也使美国公司的密码产品出口受到极大的限制和影响。

从技术角度来看，赞成和反对的意见也都有。赞成意见认为，应宣扬和推动这种技术的研究与开发；反对意见认为，该系统的技术还不成熟，基于密钥托管的加密系统的基础设施会导致安全性能下降，投资成本增高。

本书对这种争议不做过多评论，重点讨论这种技术本身。

5.3.2　密钥托管的定义和功能

密钥托管的实现手段通常是把加密的数据和数据恢复密钥联系起来，数据恢复密钥不必是直接解密的密钥，但由它可以得到解密密钥。理论上数据恢复密钥由所信任的委托人持有，委托人可以是政府保密部门、法院或有契约的私人组织。一个密钥也可能在数个这样的委托人中被拆分成多个分量，分别由多个委托人持有。授权机构（如调查机构或情报机构）可通过适当的程序（如获得法院的许可），从数个委托人手中恢复密钥。

从技术实现角度，可以将密码托管定义为：密钥托管是指用户向 CA 在申请数据加密证书之前，必须把自己的密钥分成 t 份交给可信赖的 t 个托管人。任何一个托管人都无法通过自己存储的部分用户密钥恢复完整的用户密码。只有这 t 个人存储的密钥合在一起才能得到用户的完整密钥。因此，密钥托管有如下重要功能。

① 防止抵赖。在商务活动中，通过数字签名即可验证自己的身份以防抵赖。但当用户改变了自己的密码，他就可抵赖没有进行过此商务活动。为了防止这种抵赖，有几种办法：一种是用户在改密码时必须向 CA 说明，不能私自改变；另一种是密钥托管，当用户抵赖时，这 t 个托管人就可出示他们存储的密钥合成用户的密钥，使用户无法抵赖。

② 政府监听。政府、法律职能部门或合法的第三者为了跟踪、截获犯罪嫌疑人员的通信，需要获得通信双方的密钥。这时合法的监听者可通过用户的委托人收集密钥片后得到用户密钥，进而进行监听。

③ 用户密钥恢复。用户遗忘了密钥想恢复密钥时，可从委托人那里收集密钥片恢复密钥。

5.3.3　美国托管加密标准简介

1993 年 4 月，美国政府为了满足其电信安全、公众安全和国家安全，提出了托管加密标准 EES，该标准所使用的托管加密技术不仅提供了强加密功能，同时也为政府机构提供了实施法律授权下的监听功能。这一技术是通过一个防窜扰的芯片（称为 Clipper 芯片）来实现的。

它有两个特性：

① 一个加密算法——Skipjack 算法，该算法是由 NSA 设计的，用于加(解)密用户间通信的消息。该算法已于 1998 年 3 月公布。

② 为法律实施提供“后门”的部分——法律实施存取域(Law Enforcement Access Field，LEAF)。通过这个域，法律实施部门可在法律授权下，实现对用户通信的解密。

1. Skipjack 算法

Skipjack 算法是一个单钥分组加密算法，密钥长 80b，输入和输出的分组长均为 64b。

可使用 4 种工作模式：电码本模式，密码分组链接模式，64b 输出反馈模式，1b、8b、16b、32b 或 64b 密码反馈模式。

算法的内部细节在向公众公开以前，政府邀请了一些局外人士对算法做出评价，并公布了评价结果。评价结果认为算法的强度高于 DES，并且未发现陷门。

Skipjack 的密钥长是 80b，比 DES 的密钥长 24b，因此通过穷举搜索的蛮力攻击比 DES 多 2^{24} 倍的搜索。所以若假定处理能力的费用每 18 个月减少一半，那么破译它所需的代价要 $1.5\times24=36$ 年才能减少到今天破译 DES 的代价。

2. 托管加密芯片

Skipjack 算法以及在法律授权下对加密结果的存取是通过防窜扰的托管加密芯片来实现的。芯片装有以下部分：

① Skipjack 算法。

② 80b 的族密钥 KF(Family Key)，同一批芯片的族密钥都相同。

③ 芯片单元识别符 UID(Unique IDentifier)。

④ 80b 的芯片单元密钥 KU(Unique Key)，它是两个 80b 的芯片单元密钥分量(KU_1，KU_2)的异或。

⑤ 控制软件。

这些部分被固化在芯片上。编程过程是在由两个托管机构的代表监控下的安全工厂中进行的，一段时间一批。编程过程如图 5-12 所示。

首先，托管机构的代表通过向编程设备输入两个参数 r_1、r_2(随机数)对芯片编程处理器初始化。芯片编程处理器对每个芯片，分别计算以上两个初始参数 r_1、r_2 和 UID 的函数，作为单元密钥的两个分量 KU_1 和 KU_2。求 KU_1 XOR KU_2，作为芯片单元密钥 KU。UID 和 KU 放在芯片中。

然后，用分配给托管机构 1 的密钥 k_1 加密 KU_1 得 $E_{k_1}(KU_1)$。类似地，用分配给托管机构 2 的加密密钥 k_2 加密 KU_2 得 $E_{k_2}(KU_2)$。(UID，$E_{k_1}(KU_1)$)和(UID，$E_{k_2}(KU_2)$)分别给托管机构 1 和托管机构 2，并以托管形式保存。以加密方式保存单元密钥分量是为了防止密钥分量被窃或泄露。

编程过程结束后，编程处理器被清除，以使芯片的单元密钥不能被他人获得或被他人计算，只能从两个托管机构获得加密的单元密钥分量，并且使用特定的政府解密设备来解密。

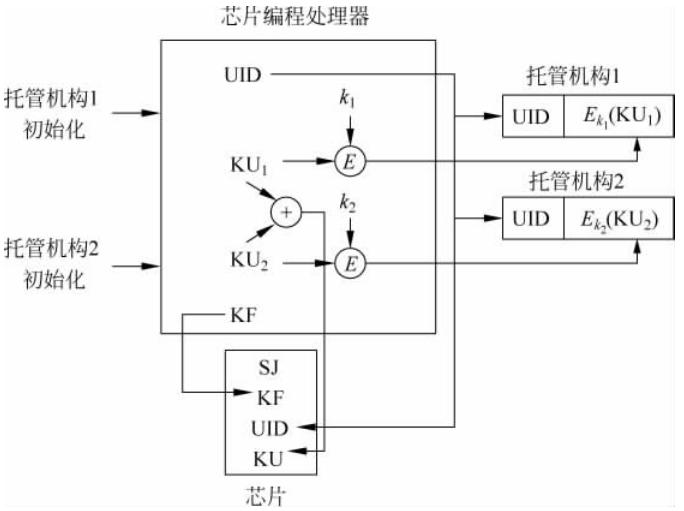

图 5-12　芯片编程过程

3. 用托管加密芯片加密

通信双方为了使用 Skipjack 算法加密他们的通信，都必须有一个装有托管加密芯片的安全的防窜扰设备。该设备负责实现建立安全信道所需的协议，包括协商或分布用于加密通信的 80b 秘密会话密钥 k_S。

例如，会话密钥可使用 DH 密钥交换算法，该算法执行过程中，两个设备仅交换公共值即可获得公共的秘密会话密钥。

80b 的会话密钥 k_S 建立后，被传送给加密芯片，用于与初始量 IV(由芯片产生)一起产生 LEAF。

控制软件使用芯片单元密钥 KU 加密 k_S，然后将加密后的结果和芯片识别符 UID、认证符 A 链接，再使用公共的族密钥 KF 加密以上链接的结果而产生 LEAF。其过程如图 5-13 所示。

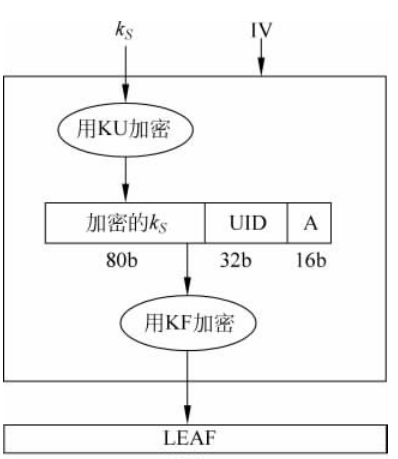

图 5-13　LEAF 过程

最后将 IV 和 LEAF 传递给接收芯片，用于建立同步。同步建立后，会话密钥就可用于通信双方的加解密。对于语音通信，消息串（语音）首先应被数字化。

图 5-14 显示的是在发送者的安全设备和接收者的安全设备之间传送 LEAF 以及用会话密钥 k_S 加密明文消息 hello 的过程。图中未显示初始量。

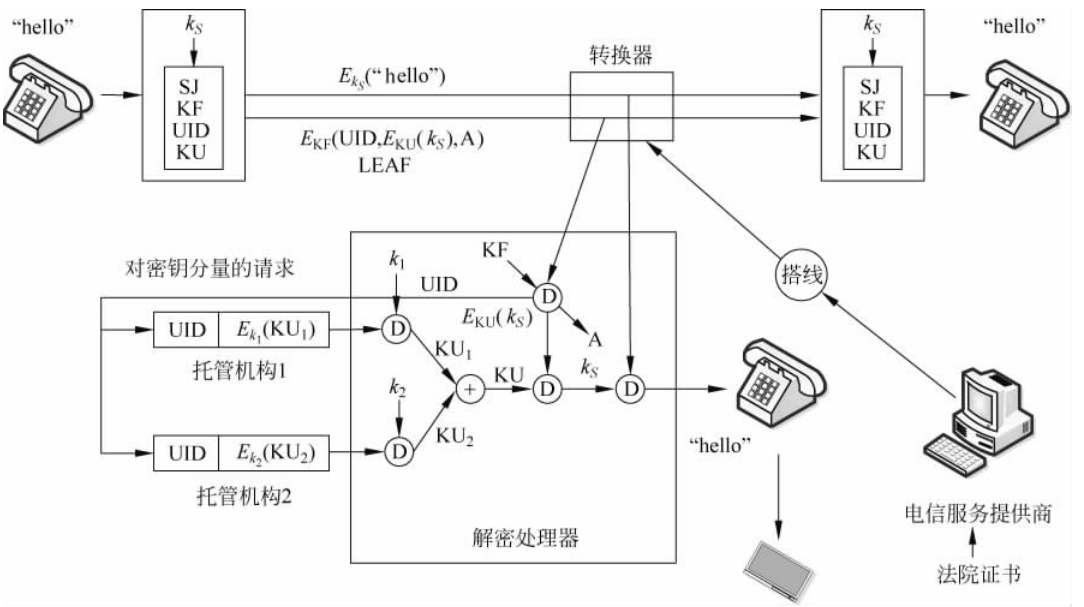

图 5-14　传送 LEAF 以及用会话密钥 k_S 加密明文消息 hello 的过程

在双向通信（如电话）中，通信每一方的安全设备都需传送一个 IV 和由其设备芯片计算出的 LEAF。然后，两个设备使用同一会话密钥 k_S 来加密传送给通信对方的消息，并解密由对方传回的消息。

4. 密钥托管的一个应用

政府机构在进行犯罪调查时，为了监听被调查者的通信，首先必须取得法院的许可证书，并将许可证书出示给通信服务的提供者（电信部门），然后从电信部门租用线路用来截取被监听者的通信。

如果被监听者的通信是经过加密的，则被截获的通信首先通过一个政府控制的解密设备。解密设备可识别由托管芯片加密的通信，取出 LEAF 和 IV，并使用族密钥 KF 解密 LEAF 以取出芯片识别符 UID 和加密的会话密钥 $E_{KU}(k_S)$。

政府机构将芯片识别符 UID、法院许可监听的许可证书、解密设备的顺序号以及政府机构对该芯片的单元密钥分量的要求一起给托管机构。

托管机构在收到并验证政府机构传送的内容后，将被加密的单元密钥分量 $E_{k_1}(KU_1)$和 $E_{k_2}(KU_2)$传送给政府机构的解密设备。

解密设备分别使用加密密钥 k_1 和 k_2 解密 $E_{k_1}(KU_1)$和 $E_{k_2}(KU_2)$以得到 KU_1、KU_2，求它们的异或 KU_1 XOR KU_2，即为单元密钥 KU。

由单元密钥 KU 解密 $E_{KU}(k_S)$，得到被调查者的会话密钥 k_S。最后解密设备使用 k_S

解密被调查者的通信。

为了实现解密，解密设备在初始化阶段，应安装族密钥 KF 和密钥加密密钥 k_1、k_2。

托管机构在传送加密的密钥分量时，也传送监听的截止时间。因此解密设备的设计应使得它到截止时间后，可自动销毁芯片单元密钥及用于得到单元密钥的所有信息。

同时，因为每一次新的会话用一新的会话密钥加密，所以解密设备在监听的截止时间之前，在截获调查者新的会话时，可不经过托管机构而直接从 LEAF 中提取并解密会话密钥。

因此，除在得到密钥时可有一个时间延迟外，对被截获通信的解密也可在监听的有效期内有一个时间延迟。这种时间延迟对有些案情极为重要，如监听进行绑架的犯罪分子或监听有计划的恐怖活动。

5.4　秘密分割

秘密分割是将某一密码信息 k 分成 n 片(k_i，$i=1,2,\cdots,n$)给 n 个人，只有当这 n 个人同时给出自己的秘密分片时，才能恢复信息。它与秘密共享具有相似性，但是没有秘密共享的要求高，也很容易实现。在很多场合，为了避免权力过于集中，必须将秘密分割开来让多人掌管。只有达到一定数量的人同时合作，才能恢复这个秘密。这就是密码学中的门限方案。

门限方案：秘密 s 被分割成 n 个部分信息，每一部分信息称为一个子密钥，每个子密钥都由不同的参与者掌握，使得只有 k 个或 k 个以上的参与者共同努力才能重构信息 s；否则无法重构消息 s。这种方案就称为(k，n)门限方案，k 称为方案的门限值。

5.4.1　秘密分割门限方案

在导弹控制发射、重要场所通行检验等情况下，通常必须由两人或多人同时参与才能生效，这时都需要将秘密分给多人掌管，而且必须有一定数量的掌管秘密的人同时到场才能恢复这一秘密。由此，引入门限方案(Threshold Schemes)的一般概念。

定义 5-1　设秘密 s 被分成 n 个部分信息，每一部分信息称为一个子密钥或影子(Share or Shadow)，由一个参与者持有，使得：

① 由 k 个或多于 k 个参与者所持有的部分信息可重构 s。

② 由少于 k 个参与者所持有的部分信息无法重构 s。

则称这种方案为(k，n)秘密分割门限方案，k 称为方案的门限值。

如果一个参与者或一组未经授权的参与者在猜测秘密 s 时，并不比局外人猜秘密时有优势，即：

③ 由少于 k 个参与者所持有的部分秘密信息得不到秘密 s 的任何信息，则称这个方案是完善的，即(k，n)秘密分割门限方案是完善的。

为了可靠性也要适当控制 $n-k$ 的值，以免该值太小而使得秘密的恢复由于有大于 $n-k$ 个人员不能到场而无法恢复。所以(k，n)门限的安全性在于既要防止少于 k 个人合

作恢复秘密，又要防止对 t 个人的攻击而阻碍秘密的恢复。

下面介绍最具代表性的秘密分割门限方案。

5.4.2 Shamir 门限方案

Shamir 门限方案是典型的秘密分割门限方案。下面详细论述该方案的原理。

Shamir 门限方案基于多项式的 Lagrange 插值公式。插值是数学分析中的一个基本问题。

已知一个函数 $\varphi(x)$ 在 k 个互不相同的点的函数值 $\varphi(x_i)(i=1,2,\cdots,k)$，寻求一个满足 $f(x_i)=\varphi(x_i)(i=1,2,\cdots,k)$ 的函数 $f(x)$ 来逼近 $\varphi(x)$，$f(x)$ 称为 $\varphi(x)$ 的插值函数，也称插值多项式。

已知 $\varphi(x)$ 在 k 个互不相同的点的函数值 $\varphi(x_i)(i=1,2,\cdots,k)$，可构造 $k-1$ 次 Lagrange 插值多项式：

$$f(x)=\sum_{j=1}^{k}\varphi(x_j)\prod_{\substack{l=1\\ l\neq j}}^{k}\frac{x-x_1}{x_j-x_1}$$

也可认为是已知 $k-1$ 次多项式 $f(x)$ 的 k 个互不相同的点的函数值 $f(x_i)(i=1,2,\cdots,k)$，构造多项式 $f(x)$。

若把密钥 s 取作 $f(0)$，n 个子密钥取作 $f(x_i)(i=1,2,\cdots,n)$，那么利用其中的任意 k 个子密钥可重构 $f(x)$，从而可得密钥 s。

这种门限方案也可按如下更一般的方式来构造：

① 设 GF(q)是一有限域，其中 q 是一个大素数，满足 $q\geqslant n+1$。

② 秘密 s 是在 GF(q)\{0}上均匀选取的一个随机数，表示为 $s\in_R$GF(q)\{0}。

③ $k-1$ 个系数 $a_1,a_2,\cdots,a_i,\cdots,a_{k-1}$ 的选取也满足 $a_i\in_R$GF(q)\{0} ($i=1,2,\cdots,k-1$)。

④ 在 GF(q)上构造一个 $k-1$ 次多项式 $f(x)=a_0+a_1x+\cdots+a_{k-1}x^{k-1}$。

⑤ n 个参与者记为 $P_1,P_2,\cdots,P_i,\cdots,P_n$，$P_i$ 分配到的子密钥为 $f(i)$。

⑥ 如果任意 k 个参与者 $P_{i_1},P_{i_2},\cdots,P_{i_l},\cdots,P_{i_k}(1\leqslant i_1<i_2<\cdots i_l<\cdots<i_k\leqslant n)$ 要想得到秘密 s，可使用$\{(i_l,f(i_l))\mid l=1,2,\cdots,k\}$构造如下的线性方程组：

$$\begin{aligned}
a_0+a_1(i_1)+\cdots+a_{k-1}(i_1)^{k-1}&=f(i_1)\\
a_0+a_1(i_2)+\cdots+a_{k-1}(i_2)^{k-1}&=f(i_2)\\
&\vdots\\
a_0+a_1(i_k)+\cdots+a_{k-1}(i_k)^{k-1}&=f(i_k)
\end{aligned}$$

因为 $i_l(l=1,2,\cdots,k)$ 均不相同，所以可由 Lagrange 插值公式构造如下多项式：

$$f(x)=\sum_{j=1}^{k}f(i_j)\prod_{\substack{l=1\\ l\neq j}}^{k}\frac{x-i_1}{i_j-i_1}(\mathrm{mod}\ q)$$

从而可得秘密 $s=f(0)$。

然而参与者仅需知道 $f(x)$ 的常数项 $f(0)$ 而无须知道整个多项式 $f(x)$，所以令 $x=0$，仅需以下表达式就可以求出 s：

$$s=(-1)^{k-1}\sum_{j=1}^{k}f(i_j)\prod_{\substack{l=1\\ l\neq j}}^{k}\frac{i_1}{i_j-i_1}(\bmod q)$$

如果 $k-1$ 个参与者想获得秘密 s，他们可构造出由 $k-1$ 个方程构成的线性方程组，其中有 k 个未知量。对 GF(q)中的任一值 s_0，可设 $f(0)=s_0$，这样可得第 k 个方程，并由 Lagrange 插值公式得出 $f(x)$。因此对每一 $s_0\in$GF(q)都有一个唯一的多项式满足方程组。所以已知 $k-1$ 个子密钥得不到关于秘密 s 的任何信息，因此这个方案是完善的。

例 5-2　Shamir 门限方案示例。设 $k=3$，$n=5$，$q=19$，$s=11$，随机选取 $a_1=2$，$a_2=7$，得多项式为：

$$f(x)=(7x^2+2x+11)\bmod 19$$

分别计算

$$f(1)=(7\times1^2+2\times1+11)\bmod 19=1$$
$$f(2)=(7\times2^2+2\times2+11)\bmod 19=5$$
$$f(3)=(7\times3^2+2\times3+11)\bmod 19=4$$
$$f(4)=(7\times4^2+2\times4+11)\bmod 19=17$$
$$f(5)=(7\times5^2+2\times5+11)\bmod 19=6$$

得 5 个子密钥。

如果知道其中的 3 个子密钥 $f(2)=5$，$f(3)=4$，$f(5)=6$，就可重构出 $f(x)$：

$$5\times\frac{(x-3)(x-5)}{(2-3)(2-5)}\bmod 19=5\times(3^{-1}\bmod 19)(x-3)(x-5)$$
$$=5\times13\times(x-3)(x-5)$$
$$4\times\frac{(x-2)(x-5)}{(3-2)(3-5)}\bmod 19=4\times((-2)^{-1}\bmod 19)(x-2)(x-5)$$
$$=4\times9\times(x-2)(x-5)$$
$$6\times\frac{(x-2)(x-3)}{(5-2)(5-3)}\bmod 19=6\times(6^{-1}\bmod 19)(x-2)(x-3)$$
$$=6\times16\times(x-2)(x-3)$$

所以

$$f(x)=[65(x-3)(x-5)+36(x-2)(x-5)+96(x-2)(x-3)]\bmod 19$$
$$=7x^2+2x+11$$

从而得秘密为 $s=11$。

5.4.3　基于中国剩余定理的门限方案

设 $m_1<m_2<\cdots<m_n$ 是 n 个大于 1 的整数，满足$(m_i,m_j)=1$（对任意的 $i,j,i\neq j$）和 $m_1m_2\cdots m_k>m_nm_{n-1}\cdots m_{n-k+2}$（注意这里的条件 $m_1<m_2<\cdots<m_n$ 是必需的，在此条件下，$m_1m_2\cdots m_k>m_nm_{n-1}\cdots m_{n-k+2}$表明最小的 k 个数的乘积也比最大的 $k-1$ 个数的乘积大，从而使得 $m_1,m_2,\cdots,m_n$ 中任意 k 个数的乘积都不比 $m_1m_2\cdots m_k$ 小）。

又设 s 是秘密数据，满足：

$$m_nm_{n-1}\cdots m_{n-k+2}<s<m_1m_2\cdots m_k$$

这意味着对任意 k 个数的乘积 T，$s=s \bmod T$，而任意的 $k-1$ 个数的乘积 R，则 s

mod R 在数值上不等于 s。

计算 $M=m_1m_2\cdots m_k$，$s_i=s(\mathrm{mod}\ m_i)(i=1,2,\cdots,n)$，以$(s_i,m_i,M)$作为一个子密钥，集合$\{(s_i,m_i,M)\}_{i=1\sim n}$即构成了一个$(k,n)$门限方案。

这是因为，在 k 个参与者中，每个 i、j 计算：

$$M_{ij}=M/m_{ij},\quad N_{ij}=M_{ij}^{-1}(\mathrm{mod}\ m_{ij}),\quad y_{ij}=s_{ij}M_{ij}N_{ij}$$

结合起来根据中国剩余定理可求得：

$$s=\sum_{j=1}^{k}y_{ij}\left(\mathrm{mod}\ \prod_{j=1}^{k}m_{ij}\right)$$

由于任意 k 个或 k 个以上的模数相乘都比 s 大，它们恢复出来的 s 必然相同，而少于 k 个参与者则不行。

5.5 消息认证

5.5.1 消息认证的基本概念

消息认证就是验证所收到的消息确实来自真正的发送方且是未被修改的消息，也可以验证消息的顺序和及时性，即消息认证是使接收者能够检验收到的消息是否真实的方法。因此，消息认证具有三层含义：一是检验消息来源的真实性，即对消息的发送者进行身份认证，确定信息的发送者是真的而不是冒充的；二是检验消息的完整性，即验证消息在传送或存储过程中是否被篡改、删除或插入等；三是检验消息的时效性，即验证消息在传送过程中是否被重传或延迟等。

消息认证实际上是对消息本身产生一个冗余的信息——消息认证码(Message Authentication Code，MAC)。消息认证码是利用密钥对要认证的变长消息和收发双方共享的密钥的一个函数值产生的带密钥的消息摘要函数，它对于要保护的信息来说是唯一的和一一对应的，可以有效地保护消息的完整性以及实现发送方消息的不可抵赖性和不能伪造性。消息认证码的安全性取决于两个方面：①采用的加密算法，即利用公钥加密算法对块加密，以保证消息的不可抵赖性和完整性；②待加密数据块的生成方法。

当需要进行消息认证时，仅有消息作为输入是不够的，需要加入密钥 k，消息认证码 MAC 是与这个密钥 k 相关的单向杂凑函数。消息认证码通常表示为：

$$\mathrm{MAC}=C_k(M)$$

其中，M 是长度可变的消息，k 是收、发双方共享的密钥，函数值 $C_k(M)$ 是定长的认证码，即密码校验和。

5.5.2 消息加密认证

1. 在对称加密体制下

在对称加密体制下的消息加密认证如图 5-15 和图 5-16 所示。由于攻击者不知道密钥 k，他也就不知道如何改变密文中的信息位才能在明文中产生预期的改变。

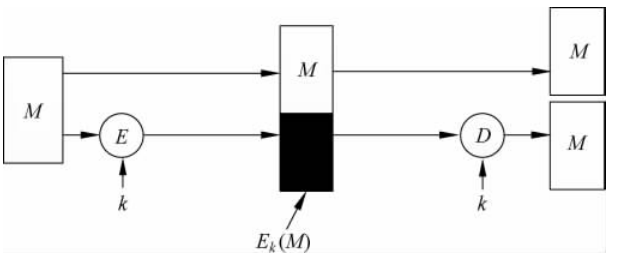

图 5-15　在对称加密体制下的消息加密认证

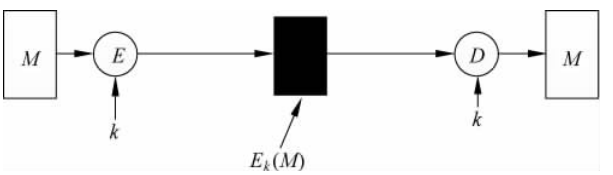

图 5-16　在对称加密体制下的消息加密认证方式 2

接收方可以根据解密后的明文是否具有合理的语法结构来进行消息认证。但有时发送的明文本身并没有明显的语法结构或特征，例如二进制文件，因此很难确定解密后的消息就是明文本身。

2. 在公钥加密体制下

在公钥加密体制下的消息加密认证如图 5-17 所示。由于只有 A 有用于产生 $E_{SK_A}(M)$ 的密钥，所以此方法提供认证。

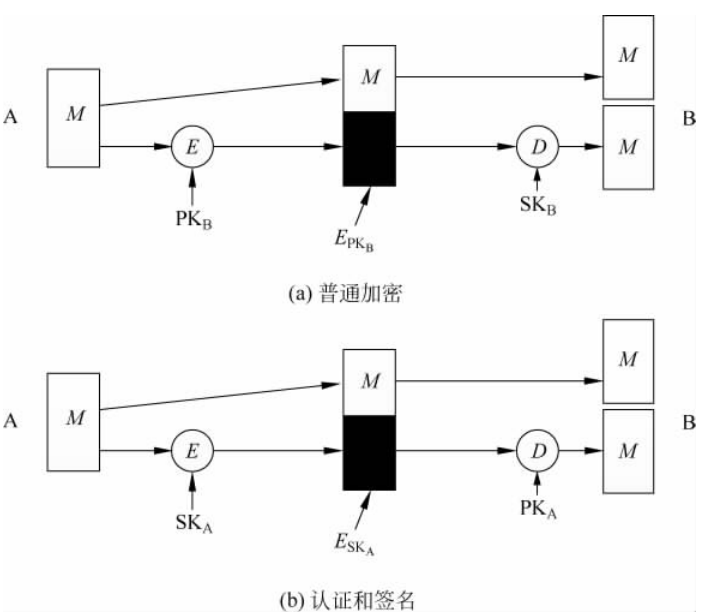

图 5-17　在公钥加密体制下的消息加密认证

5.5.3 Hash 函数

密码学上的 Hash 函数也称杂凑函数或报文摘要函数等，是一种将任意长度的消息 m 压缩（或映射）到某一固定长度的消息摘要 $H(m)$ 的函数。$H(m)$ 也称消息 m 的指纹。一个 Hash 函数是一个多对一的映射。

1. Hash 函数的分类

Hash 函数按是否需要密钥可分为以下两类。

① 不带密钥的 Hash 函数，它只有一个被通常称为消息的输入参数。

② 带密钥的 Hash 函数，它有两个不同的输入，分别称为消息和密钥。

按设计结构，散列算法可以分为三大类：标准 Hash、基于分组加密 Hash 和基于模数运算 Hash。

标准 Hash 函数有两大类：MD 系列的 MD4、MD5、HAVAI、RIPEMD、RIPEMD-160 等；SHA 系列的 SHA-1、SHA-256、SHA-384、SHA-512 等，这些 Hash 函数体现了目前主要的 Hash 函数设计技术。

2. Hash 函数的性质

从应用需求上来说，Hash 函数 H 必须满足以下性质。

① H 能够应用到任何大小的数据块上。

② H 能够生成大小固定的输出。

③ 对任意给定的 x，$H(x)$ 的计算相对简单，使得硬件和软件的实现可行。

从安全意义上来说，Hash 函数 H 应满足以下特性。

① 对任意给定的散列值 h，找到满足 $H(x)=h$ 的 x 在计算上是不可行的。

② 对任意给定的找到满足 $H(x)=H(y)$ 而 $x\neq y$ 的对 (x,y) 在计算上是不可行的。

③ 要发现满足 $H(x)=H(y)$ 而 $x\neq y$ 的对 (x,y) 是计算上不可行的。

满足以上前两个性质的杂凑函数称为弱 Hash 函数，或称杂凑函数 $H(x)$ 为弱无碰撞的；如果还满足第③个性质，就称为强 Hash 函数，或称杂凑函数 $H(x)$ 为强无碰撞的。

第①个特性是单向性的要求，通常也称抗原像性。第②个特性是弱无碰撞性，也称抗第二原像性，目的是防止伪造，即将一份报文的指纹伪造成另一份报文的指纹在计算上是不可行的。第③个特性是强无碰撞性，也称抗碰撞性，它防止对杂凑函数实施自由起始碰撞攻击或称生日攻击。

杂凑函数各特性之间的关系如下。

性质 5-1 杂凑函数的强无碰撞性（抗碰撞性）隐含弱无碰撞性（抗第二原像性）。

证明 5-1 假设杂凑函数 $H(x)$ 不具有弱无碰撞性，则对于一个 x，就可以找到一个 $y(\neq x)$，使得 $H(x)=H(y)$，这时 (x,y) 是不同输入杂凑为同一输出，而这与强无碰撞性矛盾。

3. 常用 Hash 算法

(1) MD5 Hash 算法

MD4 是 MD5 杂凑算法的前身，由 Ronald Rivest 于 1990 年 10 月作为 RFC 提出，1992 年 4 月公布的 MD4 的改进（RFC 1320，1321）称为 MD5。

MD5 算法的输入消息可任意长，压缩后输出为 128b。MD5 算法框图如图 5-18 所示。

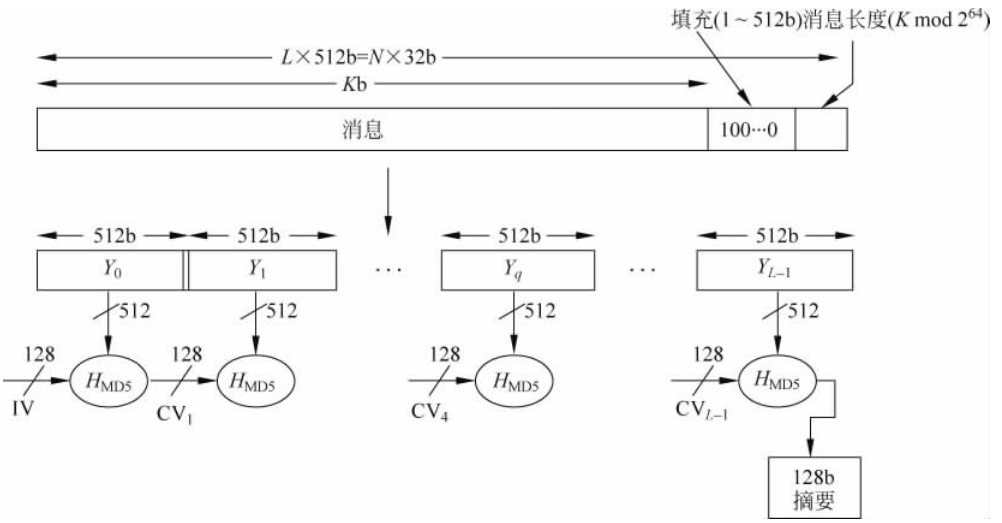

图 5-18　MD5 算法框图

MD5 算法的步骤如下。

① 分组填充，如图 5-19 所示。

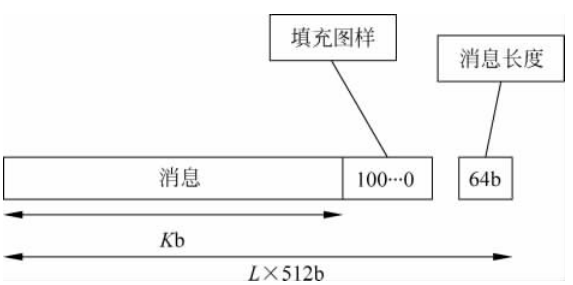

图 5-19　MD5 算法分组填充

如果消息长度大于 2^{64}，则取其对 2^{64} 的模。

执行完后，消息的长度为 512 的倍数(设为 L 倍)，则可将消息表示为分组长为 512 的一系列分组 $Y_0, Y_1, \cdots, Y_{L-1}$，而每一分组又可表示为 16 个 32b 长的字，这样消息中的总字数为 $N=L\times16$，因此消息又可按字表示为 $M[0, \cdots, N-1]$。

② 缓冲区初始化。

Hash 函数的中间结果和最终结果保存于 128 位的缓冲区中，缓冲区用 32 位的寄存器表示。可用 4 个 32b 字表示：A、B、C、D。初始存数以十六进制表示为：

A = 01234567

B = 89ABCDEF

C = FEDCBA98

D = 76543210

③ H_{MD5} 运算。

以分组为单位对消息进行处理每一分组 $Y_q(q=0, \cdots, L-1)$ 都经一压缩函数 H_{MD5} 处

理。H_{MD5}是算法的核心，如图 5-20 所示，其中又有 4 轮处理过程。

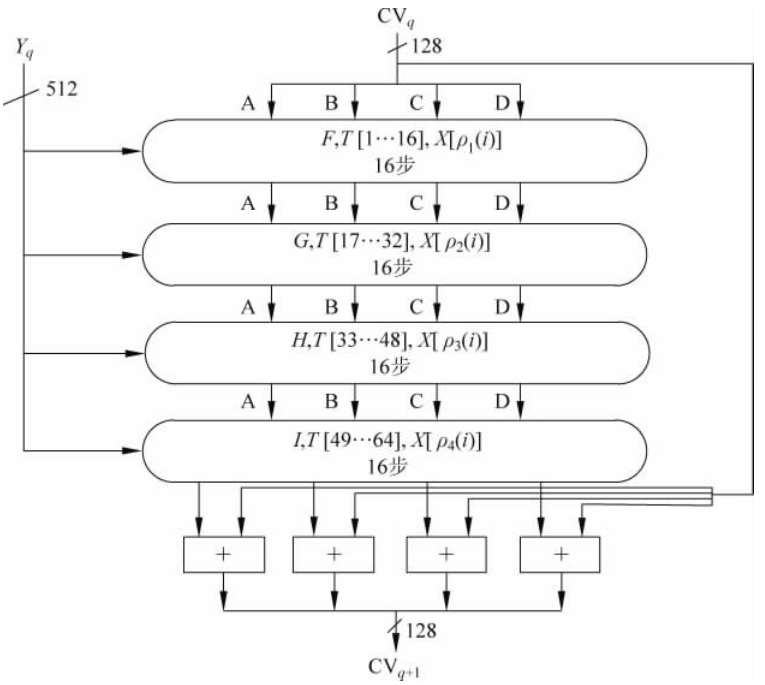

图 5-20　H_{MD5}运算示意图

H_{MD5}的 4 轮处理过程结构一样，但所用的逻辑函数不同，分别表示为 F、G、H、I。每轮的输入为当前处理的消息分组 Y_q 和缓冲区的当前值 A、B、C、D，输出仍放在缓冲区中以产生新的 A、B、C、D。

每轮又要进行 16 步迭代运算，4 轮共需 64 步完成。

第四轮的输出与第一轮的输入相加得到最后的输出。

压缩函数中的一步迭代过程如图 5-21 所示。

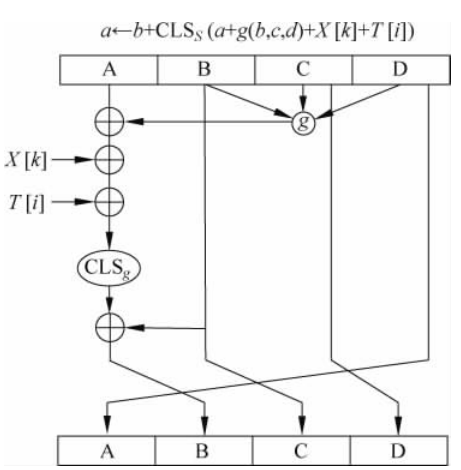

图 5-21　压缩函数中的一步迭代过程

其中基本逻辑函数 g 定义如表 5-1 所示。

表 5-1　基本逻辑函数 g 定义

轮	基本逻辑函数	$g(b,c,d)$
1	$F(b,c,d)$	$(b \wedge c) \vee (\overline{b} \wedge d)$
2	$G(b,c,d)$	$(b \wedge d) \vee (c \wedge \overline{d})$
3	$H(b,c,d)$	$b \oplus c \oplus d$
4	$I(b,c,d)$	$c \oplus (b \vee \overline{d})$

MD5 的输出为 128b，若采用纯强力攻击寻找一个消息具有给定 Hash 值的计算困难性为 2^{128}，用每秒可试验 1 000 000 000 个消息的计算机需时 1.07×10^{22} 年。

采用生日攻击法，找出具有相同杂凑值的两个消息需执行 2^{64} 次运算。

如果两个输入串的 Hash 函数的值一样，则称这两个串是一个碰撞(Collision)。既然是把任意长度的字符串变成固定长度的字符串，所以，必有一个输出串对应无穷多个输入串，碰撞是必然存在的。

2004 年 8 月 17 日，美国加利福尼亚州圣巴巴拉正在召开国际密码学会议，山东大学王小云教授公布了快速寻求 MD5 算法碰撞的算法。

(2) SHA 算法

① 算法简介。

- 由 NIST 设计。
- 1993 年成为联邦信息处理标准(FIPS PUB 180)。
- 基于 MD4 算法，与之非常类似。
- 输入为小于 2^{64}b 的任意消息。
- 分组 512b 长。
- 输出 160b。

② 算法描述。

- 消息填充：与 MD5 完全相同。
- 附加消息长度：64b 长度。
- 缓冲区初始化。

$$A = 67452301$$
$$B = EFCDAB89$$
$$C = 98BADCFB$$
$$D = 10325476$$
$$E = C3D2E1F0$$

③ 分组处理，如图 5-22 所示。

④ SHA-1 压缩函数(单步)，如图 5-23 所示。

⑤ SHA-1 的基本逻辑函数 f_t 如表 5-2 所示，其中 $\wedge$、$\vee$、$-$、$\oplus$分别表示与、或、非、异或 4 个逻辑运算。

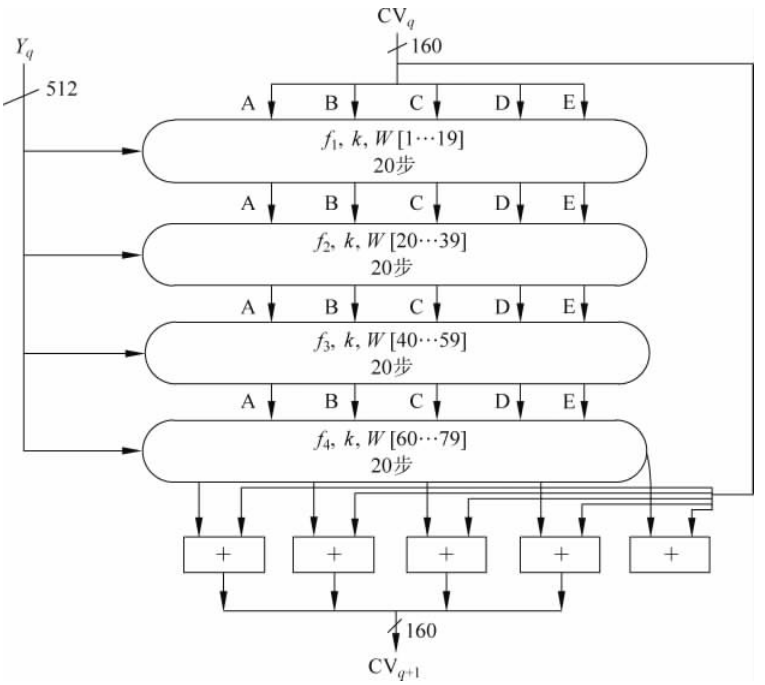

图 5-22　分组处理示意图

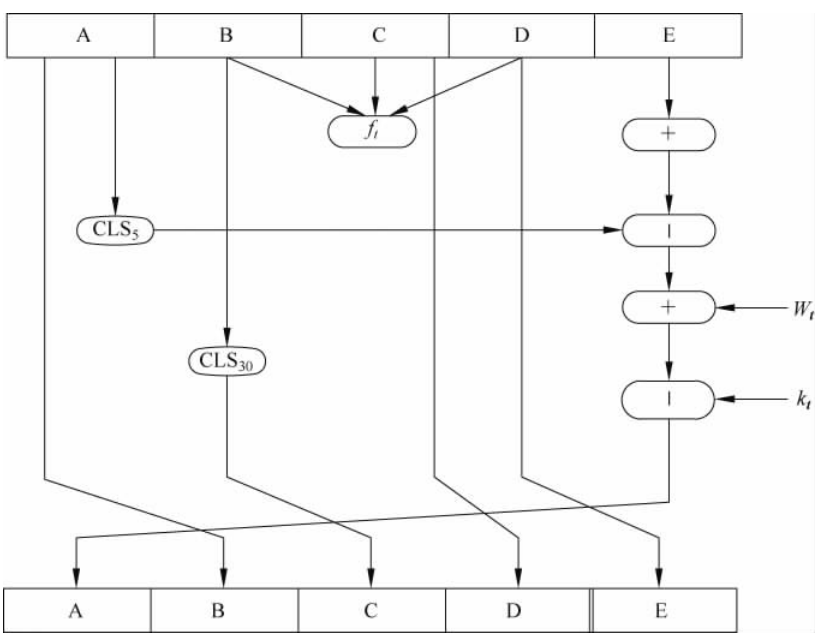

图 5-23　SHA-1 压缩函数示意图

表 5-2　SHA 中基本逻辑函数的定义

轮	基本函数	函数值
1	$f_1(B,C,D)$	$(B \wedge C) \vee (\overline{B} \wedge D)$
2	$f_2(B,C,D)$	$B \oplus C \oplus D$
3	$f_3(B,C,D)$	$(B \wedge C) \vee (B \wedge D) \vee (C \wedge D)$
4	$f_4(B,C,D)$	$B \oplus C \oplus D$

⑥ W_t：从当前 512 位输入分组导出的 32 位字。

如图 5-24 所示，下面说明如何由当前的输入分组(512b)导出 W_t(32b)。前 16 个值(即 $W_0, W_1, \cdots, W_{15}$)直接取为输入分组的 16 个相应的字，其余值(即 $W_{16}, W_{17}, \cdots, W_{79}$)取为：

$$W_t = \mathrm{CLS}_1(W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$$

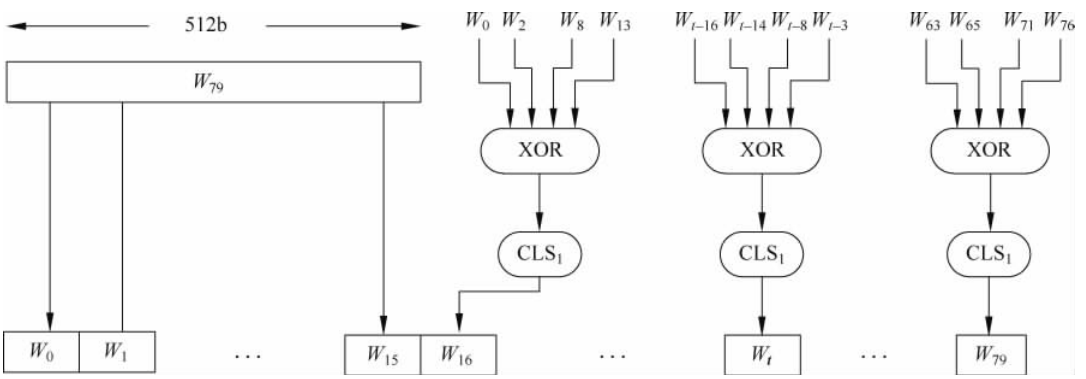

图 5-24　W_t：从当前 512 位输入分组导出的 32 位字示意图

与 MD5 比较，MD5 直接用一个消息分组的 16 个字作为每步迭代的输入，而 SHA 则将输入分组的 16 个字扩展成 80 个字以供压缩函数使用，从而使得寻找具有相同压缩值的不同的消息分组更为困难。

⑦ k_t：加法常量，如表 5-3 所示。

表 5-3　加法常量

步骤	十六进制
$0 \leqslant t \leqslant 19$	k_t=5A827999
$20 \leqslant t \leqslant 39$	k_t=6ED9EBA1
$40 \leqslant t \leqslant 59$	k_t=8F1BBCDC
$60 \leqslant t \leqslant 79$	k_t=CA62C1D6

(3) SHA 与 MD5 的比较

① 抗穷举搜索能力。

- 寻找指定 Hash 值，SHA 为 $O(2^{160})$，MD5 为 $O(2^{128})$。
- 生日攻击，SHA 为 $O(2^{80})$，MD5 为 $O(2^{64})$。

② 抗密码分析攻击的强度，SHA 似乎高于 MD5。

③ 速度。SHA 较 MD5 慢。

④ 简捷与紧致性。描述都比较简单，都不需要大的程序和代换表。

(4) 其他 Hash 算法

① MD4。

- MD4 使用 3 轮运算，每轮 16 步；MD5 使用 4 轮运算，每轮 16 步。
- MD4 的第一轮没有使用加法常量，第二轮运算中每步迭代使用的加法常量相同，第三轮运算中每步迭代使用的加法常量相同，但不同于第二轮使用的加法常量；MD5 的 64 部使用的加法常量 $T[i]$均不同。
- MD4 使用 3 个基本逻辑函数，MD5 使用 4 个。
- MD5 中每步迭代的结果都与前一步的结果相加，MD4 则没有。
- MD5 比 MD4 更复杂，所以其执行速度也更慢，Rivest 认为增加复杂性可以增加安全性。

② RIPEMD-160。

- 欧共体 RIPE 项目组提出。
- 输入可以是任意长的报文，输出 160 位摘要。
- 对输入按 512 位分组。以分组为单位处理。
- 算法的核心是具有 10 轮运算的模块，10 轮运算分成两组，每组 5 轮，每轮 16 步迭代。

习题 5

1. 密钥管理的原则是什么？

2. 对称密码体制的密钥管理策略是什么？

3. 对称密钥体制下密钥分配的方法有哪些？

4. 公钥密码体制的秘密密钥的分配方法有哪些？

5. 公钥密码体制的公钥管理策略是什么？

6. 公钥密钥体制如何实施公开密钥的分配？

7. 随机数在密码算法中的使用有哪些？简述 DES 的输出反馈模式产生随机数的工作原理。

8. 密钥托管的目的是什么？简述密钥托管的工作原理。

9. 在公钥体制中，每一用户 U 都有自己的公开钥 PK_U 和秘密钥 SK_U。如果任意两个用户 A、B 按以下方式通信，A 发给 B 消息($E_{PK_B}(m)$，A)，B 收到后，自动向 A 返回消息($E_{PK_A}(m)$，B)以通知 A、B 确实收到报文 m。

(1) 问用户 C 怎样通过攻击手段获取报文 m？

(2) 若通信格式变为：

A 发给 B 消息：$E_{PK_B}(E_{SK_A}(m), m, A)$

B 向 A 返回消息：$E_{PK_A}(E_{SK_B}(m), m, B)$

这时的安全性如何？分析 A、B 这时如何相互认证并传递消息 m。

10. DH 密钥交换算法在实施时易受中间人攻击，即攻击者截获通信双方通信的内容后可分别冒充通信双方，以获得通信双方协商的密钥。详细分析攻击者如何实施攻击。

11. 在 DH 密钥交换过程中，设大素数 $p=11$，$a=2$ 是 p 的本原根。

(1) 用户 A 的公开钥 $Y_A=9$，求其秘密钥 X_A。

(2) 设用户 B 的公开钥 $Y_B=3$，求 A 和 B 的共享密钥 k。

12. 线性同余算法 $X_{n+1}=(aX_n) \bmod 24$，问：

(1) 该算法产生的数列的最大周期是多少？

(2) a 的值是多少？

(3) 对种子有何限制？

13. 在 Shamir 门限方案中，设 $k=3$，$n=5$，$q=17$，5 个子密钥分别是 8、7、10、0、11，从中任选 3 个，构造插值多项式并求秘密数据 s。