# 第 5 章

# 数字签名技术

学习目标与要求

1. 掌握数字签名方案设计的密码学基础。
2. 掌握基本的密码学签名方案和设计思想。
3. 掌握数字签名的安全要求和基本原理。

## 5.1 引　　言

签名（手写签名或签章）是一种古老的安全技术，它通过在文件中附加特殊风格印记的方式，实现识别签名人身份并表明签名人对文件内容认可度的功能。签名通常被用于验证签名者与文件之间的“所属”关系，作为一种书写体验证方法，带有署名的签名应该是能让人读出签名者真名的，如果它不能让人读出其名，则必须能通过某种比对手段或鉴别机制使人认可其有效性。

如同手写签名一样，数字签名（电子签名或电子签章）就是一种采用密码学技术的签名方法，它能够实现更加安全的“所有权”关系验证以及所捆绑内容的完整性验证，故在电子商务和网络环境中具有重要的地位和大量的应用。我国 2005 年 4 月 1 日起实施的《中华人民共和国电子签名法》确立了电子签名的法律效力，并对电子签名行为和有关各方的合法权益进行了规范。

本章将对数字签名的基本概念和基本安全性要求，以及几种常见数字签名的构造予以介绍，使读者熟悉数字签名的构造特点；此外，本章也将介绍一些目前依然广泛使用的数字签名所存在的缺陷。当然，安全是一种综合性的选择，这些缺陷并不一定影响签名方案的实际使用，因此需要读者对这些签名予以客观评价。

## 5.2 数字签名的概念

数字签名是一种以电子形式存储消息签名的方法，其功能是实现消息认证，即对信息系统中的数据或通信中的消息来源进行认证，也就是认证消息是否来源于某个实体（发送者），保障消息来源的正确性。

与传统手写签名或签章相比较，数字签名在传输、媒介、验证方法等方面都有显著的不同，这些不同表现在以下几方面：

① 传统手写签名或签章能成为所签署文件中物理的一部分；而数字签名没有在物理上依附于文件中，因此使用数字签名必须以某种形式将签名捆绑在所签文件上。

② 传统手写签名或签章在被验证时需要与其他签名比较来判断真伪；而数字签名通过一个公开验证算法就能够让所有人来验证签名真伪。

③ 传统手写签名或签章存在验证概率的问题，安全性并不高；而数字签名使用数学方法来防伪，并使得伪造变得困难，安全性较高。

此外，随着复制手段的提高，签名，无论手写签名还是数字签名，都应该包含日期、使用说明等附加信息，以防该签名被非法地重复使用。在本文后续内容中，除特殊说明外均将数字签名简称为签名。

在密码构造方面，数字签名方案为用户提供了一种给消息签名的方法，并且这个签名可以在之后被其他所有人验证。这种**公开可验证性**决定了签名方案需要被构造在公钥密码体制之上。更具体地讲，每个用户都可以生成一个相互匹配的公钥-私钥对，且只有用户本人使用自己的私钥才能生成针对该消息的签名，但是任何人使用签名者的公钥都可以验证该消息的签名。此外，验证者可以确信在消息进行签名之后消息的内容不会被改变，并且签名者不能否认自己曾经对某个消息进行过签名，因为只有签名者拥有自己的私钥，任何其他人伪造签名在数学上都是困难的。

具体而言，数字签名应满足的要求包括：

① 收方能确认或证实发方的签字，像验证手写签名一样；

② 签名密钥被签名者所控制，发方发出签名后的消息理论上不能否认之前所签消息；

③ 任何人不能伪造发送方的签名；

④ 签署后，被签名消息及签名的任何变更都能够被发现。

总的来说，数字签名的功能在于以下两方面。

① **所有权验证**：能够指明消息与宣称者之间的所属关系，有时对于版权管理也可被理解为对拷贝权的验证。

② **完整性验证**：能够验证消息是否与签名时是一致的，用以保证双方之间交换的数据不被修改，并可检查出对消息的任何变更。

## 5.3 数字签名定义

本节将给出数字签名的形式化定义，其包括两部分：“功能性定义”和“安全性定义”。其中，功能性定义将给出协议所需要完成的功能，安全性定义则会给出签名所需

要的安全性要求。

设 $\mathcal{M}$ 是所有可能的消息组成的一个有限集，其可被称为消息空间。消息空间通常可被表示为无限长度的随机串，即 $\mathcal{M} \subseteq \{0,1\}^*$。以类似的方式，定义 $\mathcal{A}$ 是所有可能的签名组成的一个有限集，$\mathcal{K} = \mathcal{SK} \times \mathcal{PK}$ 是所有可能的公/私密钥对所组成的一个有限集。

根据以上符号定义，数字签名被定义如下：

**定义 5.1（数字签名）** 令给定的 $\mathcal{M}, \mathcal{A}$ 和 $\mathcal{K}$ 分别为消息空间、签名空间和密钥空间，则一个数字签名方案是满足以下条件的一个三元组 $(\mathcal{G}, \mathcal{S}, \mathcal{V})$（首字母缩写）。

① **密钥生成：** $\text{GenKey} : 1^\kappa \to \mathcal{SK} \times \mathcal{PK}$ 是一个随机密钥生成算法，它以安全参数 $\kappa$ 为输入，输出为用户公钥 $pk \in \mathcal{PK}$ 和对应私钥 $sk \in \mathcal{SK}$，即

$$\text{GenKey}(1^\kappa) = (pk, sk) \tag{5.1}$$

② **签名过程：** $\text{Sign} : \mathcal{SK} \times \mathcal{M} \to \mathcal{A}$ 是一个签名算法，给定输入私钥 $sk \in \mathcal{SK}$ 和任意大小的消息 $m$，输出一个签名 $\sigma$，也就是 $\sigma = \text{Sig}(sk, m)$，或者简写为

$$\text{Sign}_{sk}(m) = \sigma \tag{5.2}$$

③ **验证过程：** $\text{Verify} : \mathcal{PK} \times \mathcal{M} \times \mathcal{A} \to \{\text{true}, \text{false}\}$ 是一个验证算法，给定签名者公钥 $pk \in \mathcal{PK}$，使得对每一个消息 $m \in \mathcal{M}$ 和每一签名 $\sigma \in \mathcal{A}$，输出签名真实性判定——真或者假，即，$\text{Verify}(pk, m, \sigma) = \text{true/false}$，或者

$$\text{Verify}_{pk}(m, \sigma) = \text{true/false} \tag{5.3}$$

对于每一个公/私密钥对 $(pk, sk)$，签名过程是秘密的，只有签名人知道；而验证算法是公开的，任何接收方都可进行验证。此外，签名算法和验证算法都是多项式时间可计算函数。与之前介绍的消息完整性验证方法相比较，由于公钥密码体制的使用，签名私钥只为签名者所有或被其控制，因此要求签名是不可由其他人伪造的，并且签名者不能否认自己签过的消息，综上所述签名和完整性验证方法具有较大的区别。

为了保证签名方案的安全，首先必须保证交互证明系统中的两个基本安全性质：完整性和完备性。完整性是指对于一个有效的签名，验证算法必须能够确认该签名的有效性；完备性则相反：对于无效的签名，它通过验证算法验证的成功概率将是可忽略的。签名者对签名的不可否认性是通过不可伪造性实现的。

**定义 5.2（数字签名安全性）** 数字签名方案应具有以下两种性质：

① **完整性：** 对于任意有效的消息签名对 $(m, \sigma)$，$\sigma = \text{Sign}_{sk}(m)$，验证算法都将以概率 1 通过验证并输出 true，即

$$\Pr[\mathrm{Verify}_{pk}(m,\mathrm{Sign}_{sk}(m))=\mathrm{true}:(pk,sk)\leftarrow\mathrm{GenKey}(1^{\kappa})]=1 \tag{5.4}$$

② **完备性**：对于任意无效的消息签名对 $(m^*,\sigma^*)$，即 $\sigma^*\neq\mathrm{Sign}_k(m^*)$，验证算法都将以可忽略的概率 $\varepsilon$ 通过验证且输出 true，即

$$\Pr[\mathrm{Verify}_{pk}(m^*,\sigma^*)=\mathrm{true}:(pk,sk)\leftarrow\mathrm{GenKey}(1^{\kappa}),\sigma^*\neq\mathrm{Sign}_{sk}(m^*)]<\varepsilon \tag{5.5}$$

上述定义中，完备性等价于验证算法对无效签名输出否定判决 false 的概率是压倒性的，也就是对于 $\sigma^*\neq\mathrm{Sign}_{sk}(m^*)$，则 $\Pr[\mathrm{Verify}_{pk}(m^*,\sigma^*)=\mathrm{false}]>1-\varepsilon$。签名完备性暗含着不可伪造性（unforgeability），也就是说对于任意 $m^*$，发现有效签名 $\sigma^*=\mathrm{Sign}_{sk}(m^*)$ 的概率也是可以被忽略的。具体而言，完备性可以分为下面两种情况。

**定义 5.3**　数字签名的完备性定义通常分为以下两种情况：

① **抗选择性伪造**：攻击者 $A$ 以可忽略的概率 $\varepsilon$ 对挑战者选择的消息产生一个有效的签名。换句话说，如果给攻击者 $A$ 一个消息 $m$，那么，$A$ 伪造签名 $\sigma^*$，使得 $\mathrm{Verify}_k(m,\sigma^*)=\mathrm{true}$，上述过程的概率满足

$$\Pr\left[\mathrm{Verify}_{pk}(m,\sigma^*)=\mathrm{true}:\begin{array}{c}(pk,sk)\leftarrow\mathrm{GenKey}(1^{\kappa})\\ \sigma^*\leftarrow A^{\mathrm{Sign}_{sk}(\cdot)}(pk,m)\\ m\notin\mathrm{Query}(A)\end{array}\right]<\varepsilon \tag{5.6}$$

其中，$A$ 表示攻击者，$A^{\mathrm{Sign}_{sk}(\cdot)}(pk,m)$ 表示攻击者可以通过签名预言机 $\mathrm{Sign}_{sk}(\cdot)$ 要求签名者对他提出的消息进行签名，并在查询之后输出一个消息 $m$ 的伪造签名 $\sigma^*$，$\mathrm{Query}(A)$ 表示被攻击者 $A$ 查询 $\mathrm{Sign}_{sk}(\cdot)$ 的消息集合，但 $m$ 不是 $A$ 曾经查询过的消息。

② **抗存在性伪造**：攻击者 $A$ 能以可忽略的概率至少为一个消息产生有效的签名。换句话说，攻击者能产生一个新的消息签名对 $(m^*,\sigma^*)$，使得 $\mathrm{Verify}_{pk}(m^*,\sigma^*)=\mathrm{true}$，上述过程的概率满足

$$\Pr\left[\mathrm{Verify}_{pk}(m^*,\sigma^*)=\mathrm{true}:\begin{array}{c}(pk,sk)\leftarrow\mathrm{GenKey}(1^{\kappa})\\ (m^*,\sigma^*)\leftarrow A^{\mathrm{Sign}_{sk}(\cdot)}(pk,1^{\kappa})\\ (m^*,\sigma^*)\notin\mathrm{Query}(A)\end{array}\right]<\varepsilon \tag{5.7}$$

其中，$A^{\mathrm{Sign}_{sk}(\cdot)}(pk,1^{\kappa})$ 表示攻击者可以要求签名者对他提出的任何消息进行签名并在查询之后输出一个伪造的消息签名对 $(m^*,\sigma^*)$，$\mathrm{Query}(A)$ 表示被攻击者 $A$ 查询 $\mathrm{Sign}_{sk}(\cdot)$ 的消息签名对集合，但 $m^*$ 不是 $A$ 曾经查询过的消息。

一个签名方案不可能是无条件安全的，因为对于一个给定的消息 $m$ 而言，敌手使用公开算法 Verify 可以测试所有可能的签名 $\sigma\in\mathcal{A}$，直到他发现一个有效的签名。因此，给定足够的时间，敌手总能伪造任何消息 $m$ 的签名。因此，如同公钥加

密体制一样，设计数字签名的目标是找到在计算上不可行或可在数学上证明安全的签名方案。

## 5.4 数字签名的一般构造

在介绍 Hash 函数和 MAC 构造的前文中，已指出它们构造的前提条件是单向函数的存在。由于签名方案引入了公钥密码体制，故其构造的条件也随之增强，根据已有的研究表明，它的前提是单向陷门函数（One-way Trapdoor Function，OTF）。

数字签名构造的密码学基础是单向陷门函数 OTF 存在。

这里，所谓单向陷门函数是指该函数具有如下性质：

① 计算 $f(x)=y$ 是容易的；

② 计算 $f^{-1}(y)$ 在数学上是困难的；

③ 给定陷门 $k$，则 $f_k^{-1}(y)$ 可以为有效计算。

下面给出一个可证明安全的一般数字签名构造，它由一个单向函数（或 Hash 函数）$h$ 和一个单向陷门函数 $f$ 构成。该方案的构造如下：

① **密钥生成**：由一个可信第三方选择一个单向函数 $h$ 和一个单向陷门函数 $f$，陷门为 $k$。公布两个函数 $h$ 和 $f$，即 $pk=(f,h)$，并将陷门 $sk=(k)$ 发送给签名者。

② **签名过程**：签名者首先利用单向函数 $h$ 计算待签名消息 $m$ 的摘要 $h(m)$，然后计算摘要的签名为

$$\sigma=\mathrm{Sign}_{sk}(m)=f_k^{-1}(h(m))$$

最后将消息签名对 $(m,\sigma)$ 发送给验证者。

③ **验证过程**：验证者接收到消息签名对 $(m,\sigma)$ 后通过计算 $h(m)$ 和 $f(\sigma)$ 是否相等来验证签名的有效性，即

$$\mathrm{Verify}_{pk}(m,\sigma)=\begin{cases}\text{true} & h(m)=f(\sigma)\\ \text{false} & h(m)\neq f(\sigma)\end{cases}\tag{5.8}$$

上述一般签名构造中使用的函数 $h$ 通常是公开的密码 Hash 函数，如 SHA3。同样地，常见的签名方案也几乎总是和一种非常快的 Hash 函数结合使用。Hash 函数 $h:\{0,1\}^*\to\{0,1\}^k$ 以任意长度的消息作为输入，返回一条特定长度的消息摘要，这就要求 Hash 函数输出被转为函数 $f$ 的定义域，并在此基础上实现消息的签名。也就是说，**消息先经过 Hash 函数获取摘要，然后摘要被用于生成签名。**

**定理 5.1** 上述一般数字签名构造可满足数字签名安全性中的完整性要求。

**证明** 对于有效的消息签名对 $(m,\sigma)$，其中 $\sigma=f_{sk}^{-1}(h(m))$ 总能通过验证者的验证，因为总有 $h(m)=f(\sigma)$ 成立。 ■

**定理 5.2** 如果 $f$ 是一个具有抗二次原像攻击的单向陷门函数，$h$ 是一个碰撞稳固的单向函数，则上述一般数字签名构造满足完备性要求。

**证明** 对于无效的消息签名对 $(m,\sigma)$，其若要通过验证者的验证，则可分为存在性伪造和选择性伪造两种情况：

① **“选择性伪造”不可行**：若可对上述方案进行选择性伪造，那么任意给定一个消息 $m$，敌手可伪造出一个签名 $\sigma^*$，使得 $h(m)=f(\sigma^*)$ 成立。这意味着敌手能够有效地计算出 $\sigma^*=f^{-1}(h(m))$。由于在没有密钥 $k$ 的情况下，单向陷门函数 $f$ 是求逆困难的，$\sigma^*$ 不可在任何多项式时间内获得，所以敌手的能力与 $f$ 的求逆困难假设矛盾。因此，**在单向陷门函数 $f$ 求逆困难的假设下，上述方案可抵抗选择性伪造**。

② **“存在性伪造”不可行**：这种情况可分为下面三种情况：

$m\neq m^*$ **但** $\sigma=\sigma^*$：敌手首先找到两条消息 $m^*\neq m$ 使得 $h(m^*)=h(m)$，然后他将消息 $m$ 发送给签名者，并让签名者对消息摘要 $h(m)$ 签名获得 $\sigma$。那么 $(m^*,\sigma)$ 是有效的签名消息，而 $\sigma$ 是消息 $m^*$ 的伪造签名。这是一种利用选择消息来实现攻击的存在性伪造；**如果 $h$ 是抗碰撞的，这种攻击就能避免**。

$m=m^*$ **但** $\sigma\neq\sigma^*$：敌手给定消息签名对 $(m,\sigma)$，他将消息 $m$ 发送给签名者，并让签名者对消息摘要 $h(m)$ 签名获得 $\sigma^*$，如果 $\sigma\neq\sigma^*$，那么 $(m^*,\sigma^*)$ 是有效伪造，且 $f(\sigma^*)=f(\sigma)$。这是典型的二次原像攻击，因此，**如果 $f$ 可抗二次原像攻击，这种攻击就能避免**。

$m\neq m^*$ **且** $\sigma\neq\sigma^*$：如果存在 $h(m^*)=f(\sigma^*)$ 成立，意味着该消息签名对与之前观察到的消息签名对都没有紧密的关系，因此，考虑两种可能性，即 $m^*=h^{-1}(f(\sigma^*))$ 和 $\sigma^*=f^{-1}(h(m^*))$。前者存在与 $h$ 的不可逆矛盾，后者存在与 $f$ 的不可逆矛盾。因此，**如果 $f$ 和 $h$ 都是不可逆的（反向计算困难），这种攻击就能避免**。

由此可知，保证签名方案的前提是 $f$ 是一个具有抗二次原像攻击的单向陷门函数，而 $h$ 是一个碰撞稳固的单向函数。问题得证。 ■

数字签名的一般构造并不是唯一的，可根据不同的需要选择不同的设计。例如，下面是一个更严格的备选方案。

① 签名函数：$\sigma=h(m)\oplus f_{sk}^{-1}(h(m))$；

② 验证函数：$f(h(m)\oplus\sigma)=h(m)$。

感兴趣的读者可验证此方案的正确性和安全性。需要说明的是，在具体数字签名方案设计中，由于密码学陷门构造不同和应用需求不同，签名方案的实际构造将与上面一般性构造差异很大，这也使得数字签名的研究丰富多彩。

# 5.5 RSA 数字签名

## 5.5.1 RSA 数字签名方案

RSA 是目前使用最广的公钥密码系统，其特点是简单、易于非密码专业人员理解和实现，故在互联网和商业密码领域一直处于统治地位。从当前密码理论研究看，RSA 密码本身有很多问题，如无法实现选择明文攻击下的语义安全（Semantic Security under Chosen Plaintext Attack）等，但是这并没有影响它的实际应用。本节将具体介绍 RSA 签名方案。

RSA 签名方案建立在 RSA 密码系统上，可以简单将其理解为 RSA 加密过程的逆过程：用私钥解密作为签名，用公钥加密作为验证过程，具体的方案构造如下（见表 5.1）所示。

**表 5.1 RSA 数字签名方案**

| 步 骤 | 签 名 者 | 认 证 者 |
|---|---|---|
| 密钥生成 | 选择大素数 $p$ 和 $q$，计算 $N=pq$。选择验证指数 $e$，使得 $\gcd[e,\phi(N)]=1$。计算签名指数 $d$，使 $d$ 满足 $ed\equiv 1[\bmod \phi(N)]$。公钥为 $(N,e)$ | |
| 签名过程 | 计算消息 $m$ 的签名 $\sigma\equiv m^e(\bmod N)$。发送消息和签名对 $(m,\sigma)$ 给验证者 | |
| 验证过程 | | 根据消息签名对 $(m,\sigma)$ 和签名者的公钥 $(N,e)$，计算 $m'\equiv\sigma^e(\bmod N)$。如果 $m'=m$，则签名有效；否则，签名无效 |

①**密钥生成：**秘密选择大素数 $p$ 和 $q$，计算 $N=pq$。选择验证指数 $e$，使得 $\gcd(v,\varphi(N))=1$，其中，$\varphi(N)=(p-1)(q-1)$ 为欧拉函数，签名者计算签名指数 $d$，使其满足 $ed\equiv 1[\bmod \varphi(N)]$，即 $sk=(d)$。最后，公开大整数和验证指数为公钥 $pk=(N,e)$。

② **签名过程：**用签名指数 $d$ 对消息 $m$ 进行签名得到 $\sigma\equiv m^d \mod N$，将消息签名对 $(m,\sigma)$ 发送给验证者。

③ **验证算法：**验证者收到消息签名对 $(m,\sigma)$ 之后，首先利用发送方的公开参数 $(N,e)$ 验证签名得 $m'\equiv\sigma^e \mod N$。如果 $m'=m$，表示签名有效；否则，签名无效。也就是

$$\mathrm{Verify}_{pk}(m,\sigma)=\begin{cases}\text{true} & m\equiv\sigma^e(\bmod N)\\ \text{false} & m\neq\sigma^e(\bmod N)\end{cases} \tag{5.9}$$

### 5.5.2 RSA 数字签名方案的安全性

RSA 签名方案的有效性可以直接通过欧拉定理得到，任何人都可以获取签名者的公钥来对签名进行有效性验证。但这并不意味着 RSA 签名是安全的，下面将证明 RSA 数字签名方案存在选择性伪造和存在性伪造的风险。

**定理 5.3（选择性伪造）** RSA 签名方案在任意消息攻击下可能被选择性伪造。

**证明** 为了产生消息 $m$ 的签名，先选择一个随机数 $r \in \mathbb{Z}_N^*$。定义 $m_1$ 和 $m_2$ 分别为 $m_1 \equiv mr(\mathrm{mod}\, N)$，$m_2 \equiv r^{-1}(\mathrm{mod}\, N)$。遵循选择性伪造攻击中的规则，可请求签名者生成两个消息的签名分别为 $\sigma_1 \equiv m_1^d(\mathrm{mod}\, N)$ 和 $\sigma_2 \equiv m_2^d(\mathrm{mod}\, N)$。进而生成这两个消息乘积的签名，即

$$\sigma = \sigma_1\sigma_2 = m_1^d \cdot m_2^d = (mr)^d \cdot (r^{-1})^d = (mr \cdot r^{-1})^d \equiv m^d(\mathrm{mod}\, N)$$

$\sigma$ 也就是消息 $m$ 的签名，即 $m_1m_2 = (mr)r^{-1} \equiv m(\mathrm{mod}\, N)$，伪造成功，定理得证。■

**定理 5.4（存在性伪造）** RSA 签名方案在已知消息攻击下可进行存在性伪造。

**证明** 如果能够分别产生两个消息的签名，那么两个消息乘积的签名就是两个消息签名的乘积。设定 $m_1$ 和 $m_2$ 是两个消息，遵循选择性伪造攻击中的规则，可请求签名者生成两个消息的签名分别为 $\sigma_1 \equiv m_1^d(\mathrm{mod}\, N)$ 和 $\sigma_2 \equiv m_2^d(\mathrm{mod}\, N)$。现在产生两个消息乘积的签名：

$$\sigma = (m_1m_2)^d = m_1^d \cdot m_2^d \equiv \sigma_1\sigma_2(\mathrm{mod}\, N)$$

显然，$\sigma$ 也就是消息 $m_1m_2$ 的签名，因此伪造成功，定理得证。 ■

尽管从可证明安全的角度而言 RSA 签名并不优秀，但是上述两种攻击的实施都需要敌手欺骗签名者进行某些看似无害的签名，只是这些签名的明文一般是没有实际意义的，因此当 RSA 签名者是人类时敌手很难实施欺骗，这也是 RSA 签名被广泛使用的原因；此外，实际应用中通常还会采用消息的 Hash 值进行签名，使上述攻击均可被避免。但是，随着大量网络签名者由人类变为程序，这种情况下 RSA 签名攻击变得更容易被实施，因此，在未来应用中应尽量采用更加安全的签名方案。

### 5.5.3 RSA 数字签名实例

**例 5.1** 假定选择两个素数，$p = 1223$，$q = 1987$。计算 $N = p*q = 1223 \times 1987 = 2\ 430\ 101$。进而可计算 $\varphi(N) = (p-1)(q-1) = 1222 \times 1986 = 2\ 426\ 892$。选择 $e$ 使其与 $\varphi(N)$ 互素且小于 $\varphi(N)$，这里选择 $e = 1\ 051\ 235$。公开参数是 $(e = 1\ 051\ 235, N = 2\ 430\ 101)$。

签名者计算 $d$ 并使得 $ed \equiv 1[\mathrm{mod}\, \varphi(N)]$，计算可得 $d = 948\ 047$。对于输入消息 $m = 1\ 070\ 777$，签名者计算签名如下：

$$\sigma = m^s(\mathrm{mod}\, N) = 1\ 070\ 777^{948\ 047}(\mathrm{mod}\, 2\ 430\ 101) \equiv 1\ 473\ 513$$

并将消息签名对 $(m,\sigma)=(1\ 070\ 777,1\ 473\ 513)$ 发送给验证者。

任何人可用公钥对该签名进行验证如下：

$$m'=\sigma^d(\mathrm{mod}\,N)=1\ 473\ 513^{1\ 051\ 235}(\mathrm{mod}\,2\ 430\ 101)=1\ 070\ 777$$

可得 $m'=m$ ，则签名有效。

## 5.6 ElGamal 数字签名

### 5.6.1 ElGamal 数字签名方案

本节将介绍另一类重要的签名方案，它是由 ElGamal 在 1985 年提出的，并被命名为 ElGamal 数字签名。该方案的变形已被美国国家标准技术研究所（NIST）采纳为签名标准。此签名建立在 ElGamal 加密系统之上，以离散对数问题为基础，但是其签名本身与加密方案并没有直接的关系，它是为数字签名而专门设计的，这与 RSA 方案有显著的不同。

ElGamal 签名的另一个特点是签名不唯一，这意味着对任何给定的消息能够产生许多有效的签名，并且验证算法能够将它们中的任何一个作为有效签名而接受。

ElGamal 签名方案是基于有限域内求解离散对数问题困难的假设下的，明文空间为 $\mathcal{M}=\mathbb{Z}_p^*$，签名空间为 $\mathcal{A}=\mathbb{Z}_p^*\times\mathbb{Z}_{p-1}$，其中 $p$ 为一个安全素数。方案包括三个过程：密钥生成、签名算法和认证算法。具体方案构造（见表 5.2）如下。

**表 5.2 ElGamal 数字签名方案**

| 步　骤 | 签　名　者 | 认　证　者 |
|---|---|---|
| 密钥生成 | (1) 选择大素数 $p$ 和阶 $p-1$ 随机元 $g$。<br>(2) 选择秘密的签名指数 $s$，然后计算<br>$v\equiv g^s(\mathrm{mod}\quad p)$。<br>公开验证参数 $(v,p,g)$ | |
| 签名过程 | 选择一个随机数 $e$，计算 $m$ 的签名为：<br>$S_1\equiv g^e\ (\mathrm{mod}\ p)$，<br>$S_2\equiv(m-sS_1)e^{-1}\ (\mathrm{mod}\ p-1)$。<br>输出消息签名对 $(m,(S_1,S_2))$ | |
| 认证过程 | | 计算 $v_1\equiv v^{S_1}S_1^{S_2}\ (\mathrm{mod}\ p)$ 和 $v_2\equiv g^m$ $(\mathrm{mod}\ p)$。如果 $v_1=v_2$，表示签名有效；否则，签名无效 |

① **密钥生成：**签名者选择一个大素数 $p$ 和在 $\mathbb{Z}_p^*$ 下阶为 $p-1$ 的一个随机元素 $g$。

选择一个秘密的签名指数 $s$ 并计算 $v \equiv g^s \pmod{p}$。公开验证参数 $pk = (p, g, v)$，签名私钥为 $sk = (s)$。

② **签名算法**：签名者对消息 $m$ 进行签名（$m \in \mathbb{Z}_p^*$），首先选择一个随机数 $e$（一个临时密钥且 $1 < e < p$）计算签名为

$$\begin{cases} S_1 \equiv g^e \pmod{p} \\ S_2 \equiv (m - sS_1)e^{-1} \pmod{p-1} \end{cases}$$

然后将消息和签名 $(m, \sigma)$ 发送给验证者，其中 $\sigma = (S_1, S_2)$。

③ **验证算法**：验证者收到消息签名信息 $(m, \sigma)$ 之后，计算 $v^{S_1} S_1^{S_2} \pmod{p}$，然后将其与 $g^m \pmod{p}$ 进行比较，即

$$\mathrm{Verify}_{pk}(m, \sigma) = \begin{cases} \text{true} & g^m \equiv v^{S_1} S_1^{S_2} \pmod{p} \\ \text{false} & g^m \neq v^{S_1} S_1^{S_2} \pmod{p} \end{cases} \tag{5.10}$$

如果相等，表示签名有效；否则，签名无效。

### 5.6.2 ElGamal 数字签名方案的安全性

首先，假设如果签名被有效构造出来，那么验证一定会成功。分析从等式 $S_2 \equiv (m - sS_1)e^{-1} \pmod{p-1}$ 开始，可知

$$m \equiv eS_2 + sS_1 \pmod{p-1} \tag{5.11}$$

上式被载入到 $g$ 的指数上，可得到

$$g^m = g^{eS_2 + sS_1} \equiv (g^e)^{S_2} \cdot (g^s)^{S_1} \pmod{p} \tag{5.12}$$

将两个等式 $v \equiv g^s \pmod{p}$ 和 $S_1 \equiv g^e \pmod{p}$ 代入上式可得

$$g^m \equiv v^{S_1} \cdot S_1^{S_2} \pmod{p} \tag{5.13}$$

可见最终的验证等式成立。

其次，分析一下 ElGamal 数字签名方案的安全性。从协议构造上看，假定敌手在不知道加密指数 $s$ 的情况下想对给定的消息 $m$ 伪造签名。如果敌手选择一个值 $S_1$，然后试图找出相应的 $S_2$，那么他必须计算离散对数 $S_2 = \log_{S_1} g^x v^{-S_1}$，根据离散对数问题的求解困难程度可知，这是不可行的。另一方面，如果首先选择 $S_2$，然后试图找到 $S_1$，那么必须“求解”等式 $v^{S_1} S_1^{S_2} \equiv g^m \pmod{p}$，以便获得这个未知的 $S_1$，这是一个还没有已知可行方法来求解的问题。

另一方面，如果敌手先选择 $S_1$ 和 $S_2$，然后去解 $m$，那么他又一次面临着求解离散对数问题的一个实例，也就是计算 $m = \log_g v^{S_1} S_1^{S_2}$，因此，敌手也不能使用这种方法对给定的消息 $m$ 伪造签名。总之，上述这些都不是可行的方式。