数字签名方案中的孤悬因子和冗余数据

曹正军 刘木兰
(中国科学院数学与系统科学研究院数学机械化重点实验室 北京 100080)

摘 要 提出了孤悬因子概念,并明确指出在数字签名设计中必须回避这一现象. 此外,还具体分析了 5 个数字签名协议中出现的孤悬因子及冗余数据,证明了这些协议的不安全性.

关键词 孤悬因子;冗余数据;多重签名;代理签名;盲签名
中图法分类号 TP309

Suspending Factor and Redundant Data in Signature Schemes

CAO Zheng-Jun LIU Mu-Lan
(*Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences, Beijing* 100080)

Abstract The authors introduce the concept of suspending-factor, and definitely point out that it must be avoided in designing signature schemes. Besides, they analyze five signature schemes with suspending factor or redundant data and show that the schemes are not secure.

Keywords suspending-factor; redundant data; multi-signature; proxy signature; blind signature

1 引 言

如众所知,数字签名需满足如下一些基本要求:签名是可信的,签名是不可伪造的,签名是不可重用的,签名的文件是不可改变的,签名是不可抵赖的. 随着公钥密码体制研究的深入和电子商务的迅猛发展,数字签名得到了广泛应用. 签名者用自己的私钥对文件进行签名,验证者利用签名者的公钥验证签名的有效性,相对于传统的手写签名,数字签名变得更安全. 在数字签名的实现过程中,采用公开密钥算法对长文件签名效率太低. 为了提高效率,数字签名协议经常和 Hash 函数连用,利用 Hash 函数产生签名消息的摘要,或者称之为指纹,这样做同时也使得伪造者不能把消息和它的签名剥离开来. 在签名体制中还可直接利用 Hash 函数构造挑战. 著名的 Schnoerr 签名算法就是如此.

要设计一个签名方案,在做好签名私钥隐藏的同时还必须公开足够的数据供验证者验证,这些数据便是通常所说的签名数据. 签名数据的有效性通常是通过等式来验证的,我们发现签名数据在验证等式中出现的位置是有一定要求的. 签名数据在验证等式中必须处于底数或指数位置(如在表达式 x^y 中,x 在底数上,y 在指数上. 此处 $x\neq 1, y\neq 1$),否则是不安全的. 因为攻击者要伪造出一个合乎要求的底数就面临解二次或高次剩余问题,要伪造出一个合乎要求的指数就面临解离散对数问题. 为此,我们引进一个新概念.

定义 1. 在数字签名验证等式中出现的签名数据,如果有一个签名数据既不在底数位置上也不在指数位置上,而是作为一个独立的因子,我们称这样的签名数据为孤悬因子.

收稿日期:2005-05-23;修改稿收到日期:2005-11-04. 本课题得到国家"九七三"重点基础研究发展规划项目基金(2004CB318000)和国家自然科学基金(90304012)资助.
曹正军,男,1971 年生,博士研究生,主要研究方向为信息安全与密码学. E-mail:zjcao@amss.ac.cn.
刘木兰,女,1941 年生,研究员,博士生导师,主要研究领域为信息安全、密码学、计算机代数. E-mail:mlliu@amss.ac.cn.

例 1. 在下面将述及的 2.2 方案, 其验证方程为 $y^v = u y_G^{H(m, T)} \bmod p$, 其中签名数据是 (m, u, v, T, y_G), $H(\cdot)$ 是公开的 Hash 函数, y 是另一公开参数(事实上 y_G 也是系统的公开参数, 见后文的分析). 这里的签名数据 u 就是一个孤悬因子, 攻击者要伪造签名是十分容易的, 他只需做一次扩展的欧几里德算法. 给定一个待伪造的文本 m、时间参数 T, 攻击者选取一随机数 v, 计算 $u = y^v (y_G^{H(m, T)})^{-1}$, 这样得到的 (m, u, v, T) 便能满足验证方程. 由此可见, 在签名数据中不能有孤悬因子. 值得注意的是, Nyberg-Rueppel 签名方案[1] 中的 r 不是孤悬因子. 该方案中提供的方程是 $m = g^{s^{-1}} y^{rs^{-1}} r \bmod p$. 这里 m 是待恢复的消息, (r, s) 是签名数据, g, y, p 是公开参数, s^{-1} 是相对于 g 的阶 q 取的, q 本身也作为一个公开参数.

签名人应该向验证人发送哪些数据? 这是数字签名协议中一个最基本的问题. 首先, 有一点十分清楚, 签名人无需向验证人发送自己的公钥, 因为这些公开参数是经公钥认证中心认证后向社会直接公布的. 建立数字签名协议的一个基本前提就是公钥是不容替换的, 验证人能够得到签名人的公钥(攻击者无法予以替换). 如果没有这一前提, 验证人从发送人处接收到一串数据, 再计算一些等式, 这起事件并不能说明这些数据的来处及其有效性. 其次, 签名数据中的任何一个数据都不能由其它签名数据结合公开参数经有效计算推导出来. 关于这一点, 可结合下面的例子加以体会.

例 2. 在下面将述及的 4.2 方案, 其验证方法是: 验证者接收到签名数据 (D, T, m) 后计算

$$h_1 = H(ID_1), \cdots, h_k = H(ID_k),$$

$$T^* = D^e (h_1 h_2 \cdots h_k)^m \bmod n, \quad m^* = H(M, T^*).$$

如果 $m^* = m$ 则接受签名. 其中 $ID_i (1 \leq i \leq k)$ 是多个签名人的身份信息, $H(\cdot)$ 是公开的 Hash 函数, e 是系统的一个公开参数, M 是消息. 这里的签名数据 T 就是多余的. 从表面上看, 验证过程中没有用到 T 这一数据. 事实上在后文的分析中, 我们可以看出 $T = T^* = D^e (h_1 h_2 \cdots h_k)^m \bmod n$. 即 T 可以由其它的签名数据 D、m 结合公开参数 e、$H(ID_i) (1 \leq i \leq k)$ 经乘法及幂运算计算出来. 因此, 发送人没必要把 T 发送给验证人.

定义 2. 我们称签名数据中那些能够由其它的签名数据结合公开参数经有效计算推导出来的数据为冗余数据.

签名协议中的冗余数据是有害的, 表现在两个方面: (1) 使协议的叙述显得混乱, 比如上面例子中出现的 T; (2) 在信道中传送冗余数据浪费了通信资源.

本文将结合 5 个具体的数字签名协议, 紧紧围绕孤悬因子和冗余数据问题, 细致地分析它们的安全性. 此外, 还澄清了个别协议中对多重签名的误解.

2 Qi-Xiao 多重签名方案及分析

在数字签名过程中, 有时一个文件需要多个用户对之进行签名和认证. 如: 一个公司的出帐, 需要经理、财务、出纳的签名, 即需要对文件进行多重签名. 能够实现多个用户对同一消息进行签名的数字签名称为多重数字签名. 根据签名过程的不同, 多重数字签名方案可分为两类: 有序多重数字签名和广播多重数字签名. 无论是有序多重数字签名还是广播多重数字签名, 都包括消息的发送者、消息的签名者和消息的验证者. 在广播签名方案中还包括签名收集者.

无论是有序多重签名还是广播多重签名, 最后的验证者都必须直接使用所有相关签名人的公开信息验证签名的合法性, 否则是无法使之相信他所得到的签名是由多个人做出的. 签名验证阶段中出现的相关签名人的公开信息表明签名人参与了签名这一事件. 如果一个非匿名签名体制的验证阶段, 验证者不利用有关签名人的公开信息, 该签名就不是真正意义上的数字签名. 关于多重数字签名验证阶段中的这一基本要求, 我们无意多加论述, 可参见文献[2~5].

Qi Xiao 多重签名方案是在 ElGamal 签名的口令认证方案基础上改进得来的, 同一篇文章中作者还提出了一个一般签名方案[6].

2.1 一般签名方案

设口令生成中心 PGC 的公钥为 $y = g^x \bmod p$, 其中 p 是一大素数, $g \in Z_p^*$ 是一生成元. 选取随机数 k, 计算 $r = g^k \bmod p$, $sx = rk + ID \bmod p - 1$, 其中 ID 为用户 A 的身份表示符, 用户 A 的口令为 $PW = (r, s)$. 若 A 要将自己签发的消息 m 送给用户 B, 则 A 和 B 实施如下步骤:

(1) A 选随机数 $t \in (1, p-1)$, 并计算 $u = y^t \bmod p$, $v = t + sH(m, T) \bmod p - 1$.

(2) A 将签名 $Sig(m) = (ID, m, r, u, v, T)$ 送给 B, T 是签名时间.

(3) B 收到签名 $Sig(m)$ 后, 计算

$$y^v = u (r^r g^{ID})^{H(m, T)} \bmod p.$$

若上述等式成立, 则签名被接受; 否则拒绝签名.

2.2　多重签名方案

p, g, x, y 如前. 系统中用户 $U_i(i=1, 2, \cdots, n)$ 的身份表示符为 ID_i，口令为 (r_i, s_i)，其中

$$r_i = g^{k_i} \bmod p,\quad s_i x = r_i k_i + ID_i \bmod p - 1,$$

其公钥为 $y_i = y^{s_i} \bmod p$. 如果系统中多个用户 U_i $(i=1, 2, \cdots, n)$ 想在时刻 T 联合签发某消息 m，如 2.1 节中的方案所述，每个 U_i 先计算 (u_i, v_i)，则 U_i 关于 m 的签名为 $(ID_i, m, r_i, u_i, v_i, T)$. U_i 将其签名送给 PGC，PGC 先验证每个签名是否满足验证方程：

$$y^{v_i} = u_i (r_i^{r_i} g^{ID_i})^{H(m, T)} = u_i (y^{s_i})^{H(m, T)} = u_i y_i^{H(m, T)} (\bmod p).$$

如果所有关于 m 的签名均满足上式，则 PG 计算

$$u = \prod_{i=1}^{n} u_i \bmod p,\quad v = \sum_{i=1}^{n} v_i \bmod p - 1,$$

$$y = \prod_{i=1}^{n} y_i = \prod_{i=1}^{n} y^{s_i} = \prod_{i=1}^{n} (r_i^{r_i} g^{ID_i}) (\bmod p).$$

然后将多重签名 (m, u, v, T, y_G) 送给验证人. 如果此多重签名满足验证方程 $y^v = u y_G^{H(m, T)} \bmod p$，则多重签名被验证人接受.

2.3　分　析

(1) 2.1 方案中的验证方程

$$y^v = u (r^r g^{ID})^{H(m, T)} \bmod p \qquad (1)$$

和 2.2 方案中的验证方程

$$y^v = u y_G^{H(m, T)} \bmod p \qquad (2)$$

并没有利用 Hash 函数构成真正的挑战，原因在于验证方程中的 u 成了一个孤悬因子，攻击者可以利用它进行伪造. 文献[7] 中已经指出了这一设计错误，并就此作出了改进，把原方案中的 $H(m, T)$ 统统替换成 $H(m, u, T)$，即在 Hash 函数的输入部分添加进 u，相应的验证方程为

$$y^v = u (r^r g^{ID})^{H(m, u, T)} \bmod p \qquad (1')$$

$$y^v = u y_G^{H(m, u, T)} \bmod p \qquad (2')$$

这样一来就有效地阻止了攻击者利用 u 进行伪造的可能性(针对原方案的具体攻击见文献[7]).

(2) 2.2 方案中验证者接收到的签名是 (m, u, v, T, y_G)，实质上 $y_G = \prod_{i=1}^{n} y_i \bmod p$，它是所有签名人的公钥的乘积，正是它反映出这个签名是个多重签名，而不是某一个人的签名. 验证者本人应该从可信的安全途径直接获得验证公钥 y_G，签名发送者在发送签名时无需把这个公开参数传送给验证人. 也就是说原方案签名中多出了一个冗余数据 y_G. 如果这个数不是由验证者本人计算获得的，而是作为签名数据的一部分，那么 2.2 方案中的验证方程(即使是文献[7] 中改进的方程(2′))都是不安全的. 我们将在下一节就文献[7] 中的改进方案给出一个攻击方法.

3　Lv-Wang 多重签名方案及分析

3.1　Lv-Wang 多重签名方案

Lv-Wang 多重签名方案是对 Qi-Xiao 多重签名方案的改进，其具体描述如下[7]：如果系统中多个用户 $U_i(i=1, 2, \cdots, n)$ 想在时刻 T 联合签发某消息 m，则实施如下步骤：

(1) $U_i(i=1, 2, \cdots, n)$ 选随机数 $t_i \in (1, p-1)$，并计算 $u_i = y^{t_i} \bmod p$，然后将其发给 PGC；

(2) PGC 计算 $u' = \prod_{i=1}^{n} u_i \bmod p$，然后将其发给所有的 $U_i(i=1, 2, \cdots, n)$；

(3) $U_i(i=1, 2, \cdots, n)$ 收到 $Rg^s = \prod_{j=1}^{t} y_j^{h(m)+R} u'$ 后，计算 $v_i = t_i + s_i H(m, u', T) \bmod p$，$T$ 是签名时间；最后将 $Sig_i(m) = (ID_i, m, r_i, u_i, v_i, T)$ 发送给 PGC；

(4) PGC 收到所有的 $Sig_i(m)$，$(i=1, 2, \cdots, n)$ 后，首先计算 $u = \prod_{i=1}^{n} u_i \bmod p$. 如果 $u = u'$，验证每个部分签名是否满足验证方程：

$$y^{v_i} = u_i (r_i^{r_i} g^{ID_i})^{H(m, u, T)} = u_i (y^{s_i})^{H(m, u, T)} = u_i y_i^{H(m, u, T)} (\bmod p),$$

否则停止协议；

(5) 如果所有关于 m 的部分签名均满足上式，则 PGC 计算：

$$v = \sum_{i=1}^{n} v_i \bmod p - 1,$$

$$y_G = \prod_{i=1}^{n} y_i = \prod_{i=1}^{n} y^{s_i} = \prod_{i=1}^{n} (r_i^{r_i} g^{ID_i}) (\bmod p),$$

然后将多重签名 (m, u, v, T, y_G) 送给验证人；否则停止协议. 如果此多重签名满足验证方程：

$$y^v = u y_G^{H(m, u, T)} \bmod p \qquad (2')$$

则多重签名被验证人接受.

3.2　分　析

(1) 方案中验证者接收到的 (m, u, v, T, y_G) 中的 y_G 是个冗余数据. 如果验证者不利用各个签名人的公钥计算得出 y_G，他就不会相信这是个多重签名；甚至 (m, u, v, T, y_G) 都不能说明这是最后的发送人 PGC 的一个普通的数字签名. 下面我们给出一种伪造方法：在 T 时刻对任一消息 m，利用 PGC 的公

钥 y, 攻击者任取两个随机数 $\alpha, \beta \in (1, p-1)$ 计算:

$$u = y^{\alpha} \bmod p, \quad y_G = y^{\beta} \bmod p,$$
$$v = \alpha + \beta H(m, u, T) \bmod p - 1.$$

把(m, u, v, T, y_G)发送给验证者.

正确性:

$$u y_G^{H(m,u,T)} = y^{\alpha} (y^{\beta})^{H(m,u,T)} = y^{\alpha+\beta H(m,u,T)} = y^{v} (\bmod p).$$

(2)该方案要求口令生成中心 PGC 参与每一个多重签名的产生过程显然是不合适的, PGC 通常只负责生成用户的口令.

4 Zhang-Wei-Wang 多重签名方案及分析

文献[8]给出了基于 RSA 的两个多重签名方案, 现叙述如下.

4.1 可验证的按序多重签名

系统的建立. 首先一个权力中心 TC 随机地选取两个不同的大素数 p, q, 并计算 $n = pq$ 作为 RSA 模数, 选取公钥 e, 计算私钥 d 使得 $ed \equiv 1 (\bmod \varphi(n))$. 权力中心 TC 公布$(n, e)$并秘密保存$(d, p, q)$. $H(\cdot)$ 是一个无碰撞的单向 Hash 函数, $U_i (i = 1, 2, \cdots, k)$ 表示签名者, M 表示待签的消息, $ID_i (i = 1, 2, \cdots, k)$ 表示签名者的身份并且是公开的. 为了使签名者 U_i 能够对消息 M 签名, 权力中心 TC 计算 $h_i = H(ID_i)$, $s_i = h_i^{-d} \bmod n$. 通过秘密安全信道把证书(ID_i, s_i)分发给每个签名者 U_i, 每个签名者通过公式 $h_i = s_i^{-e} (\bmod n)$验证证书是否正确.

消息的签名. 假设签名的顺序为$(U_1, U_2, \cdots, U_k)$, 为了使 U_i 能够对消息 M 进行签名, 需要公布每个签名者的身份 $ID_i (i = 1, 2, \cdots, k)$和签名顺序.

(1)签名者 U_1 随机选取一个数 $r_1 (1 < r_1 < n)$, 并计算

$$T_1 = r_1^e (\bmod n), \quad m_1 = H(M, T_1), \quad D_1 = r_1 s_1^{m_1} (\bmod n).$$

(2)把(T_1, m_1, D_1)传送给后序签名者 U_2, U_2 如下验证签名的正确性: 计算

$$h_1 = H(ID_1), \quad T_1^* = D_1^e h_1^{m_1} (\bmod n), \quad m_1^* = H(M, T_1^*).$$

如果 $m_1^* = m_1$, 说明该签名正确, 否则终止协议. 当验证部分签名通过后, U_2 如下继续进行协议: 随机选取一个数 $r_2 (1 < r_2 < n)$, 计算

$$T_2 = T_1 r_2^e (\bmod n), \quad m_2 = H(M, T_2),$$
$$D_2 = D_1 r_2 s_2^{m_2} (\bmod n).$$

传送(T_2, m_2, D_2, f_1)给后序签名者 U_3, 其中 $f_1 = h_1^{m_1} (\bmod n)$. 依次类推.

(3)当签名者 $U_i (2 < i < k)$传送消息 M 的部分签名(T_i, m_i, D_i, f_{i-1})给签名者 U_{i+1}, 其中 $f_{i-1} = f_{i-2} h_{i-1}^{m_{i-1}}$. U_{i+1} 首先验证消息 M 的签名(T_i, m_i, D_i)的正确性. 计算

$$h_i = H(ID_i), \quad T_i^* = D_i^e f_{i-1} h_i^{m_i} (\bmod n),$$
$$m_i^* = H(M, T_i^*).$$

如果 $m_i^* = m_i$ 说明该签名正确, U_{i+1} 就继续对 M 签名如下: 随机选取一个数 $r_{i+1} (1 < r_{i+1} < n)$ 计算 $T_{i+1} = T_i r_{i+1}^e (\bmod n)$, $m_{i+1} = H(M, T_{i+1})$, $D_{i+1} = D_i r_{i+1} s_{i+1}^{m_{i+1}} (\bmod n)$, $f_i = f_{i-1} h_i^{m_i} (\bmod n)$. 传送$(T_{i+1}, m_{i+1}, D_{i+1}, f_i)$给下一个后序签名者 U_{i+2}. 直到传给第 k 个签名人为止. 最后所得的消息 M 的签名为(T_k, m_k, D_k, f_{k-1}).

签名的验证. 当验证者验证消息 M 的签名(T_k, m_k, D_k, f_{k-1})时, 计算

$$h_k = H(ID_k), \quad T_k^* = D_k^e f_{k-1} h_k^{m_k} (\bmod n),$$
$$m_k^* = H(M, T_k^*).$$

如果 $m_k^* = m_k$ 则接受签名.

分析.

(1)方案 4.1 的验证阶段实质上就是验证

$$m_k = H(M, D_k^e f_{k-1} H(ID_k)^{m_k}).$$

它使用了签名数据(T_k, m_k, D_k, f_{k-1})中的 m_k, D_k, f_{k-1}(T_k是个冗余数据)以及最后一个签名人的公开信息 ID_k以外, 并无其它能反映签名人特征的公开数据, 充其量只能说明是最后一个签名人与权力中心 TC 共同作出了签名(因为 e 是 TC 的公钥), 而不能称为多重签名.

(2)该方案验证方程中的孤悬因子 f_{k-1} 为伪造者打开了方便之门, 这是一个设计上的错误. 伪造方法如下: 给定最后一个签名人的公开信息 ID_k, 对任一消息 M, 攻击者任意选取随机数 $\alpha, \beta \in (1, n)$, 令 $D_k := \alpha$ 计算

$$h_k = H(ID_k), \quad m_k = H(M, D_k^e \beta), \quad f_{k-1} = h_k^{-m_k} \beta \bmod n.$$

签名为(m_k, D_k, f_{k-1}).

正确性:

$$H(M, D_k^e f_{k-1} h_k^{m_k}) = H(M, D_k^e h_k^{-m_k} \beta h_k^{m_k})$$
$$= H(M, D_k^e \beta) = m_k.$$

4.2 广播多重签名

系统的建立与 4.1 节相同, 不过此时多了一个签名的收集者 U_r.

消息的签名.

(1)每个签名者 $U_i (i = 1, 2, \cdots, k)$ 随机选一个数 r_i, 计算 $R_i = r_i^e (\bmod n)$, 把 R_i 发送给 U_r.

(2)U_r 计算 $T = R_1 \times R_2 \times \cdots \times R_k \bmod n$, 并把 T

广播出去；

(3) 每个签名者 U_i 计算 $m=H(M,T)$，$D_i=r_i s_i^m \bmod n$ 并把 D_i 传给 U_r.

(4) U_r 计算 $D=D_1D_2\cdots D_k \bmod n$，最后的签名为 (D,T,m).

签名的验证. 验证者计算

$$h_1=H(ID_1),\ \cdots,\ h_k=H(ID_k),$$

$$T^*=D^e(h_1h_2\cdots h_k)^m \bmod n,\ m^*=H(M,T^*),$$

如果 $m^*=m$ 则接受签名.

分析. 方案 4.2 的验证阶段实质上就是验证：

$$m=H(M,D^e(h_1h_2\cdots h_k)^m),$$

其中只用到签名数据 (D,T,m) 中的 D, m, 故此 T 是个冗余数据（此处验证者直接计算得到的 $h_i=H(ID_i)(i=1,2,\cdots,k)$，表明这是一个多重签名方案）.

5　Tan-Liu-Tang 代理盲签名方案及分析

签名人有时候想把自己的签名权委托给其他人，利用其他人来替代自己作出签名，此种形式称为代理签名[9]. 数字签名协议的一个基本特征是文件的签署者知道他们在签署什么. 有时候，我们想要别人签署一个他们从未看过其内容的文件，这便是所谓的盲签名. 盲签名是 Chaum 在 1982 年提出的[10]. 通常协议中有一个签名者，他能作出有效的签名. 另一参与者叫做请求者，他想让签名者在他提供的文件上签名，但又不想让签名者知道文件的内容. 在文件没有公开之前签名者无法把他所签名的文件和他作出的签名联系起来.

5.1　Tan-Liu-Tang 代理盲签名方案

文献[11]提出了一个代理盲签名方案. 系统由三个部分组成：签名的接收者 R，原始签名人 A 和代理签名人 B.

系统的建立.

p,q 是两个大素数，$q\mid(p-1)$，$g\in Z_p^*$ 阶为 q，$x_A, x_B\in Z_q^*$ 分别为原始签名人 A 的私钥以及代理签名人 B 的私钥. $y_A\equiv g^{x_A} \bmod p$，$y_B\equiv g^{x_B} \bmod p$ 分别是 A 和 B 的公钥. $H(\cdot)$ 是一个公开的 Hash 函数.

代理阶段.

(1) A 任取 $\bar{k}\in Z_q^*$，使得 $\bar{r}y_A^{\bar{r}} \pmod p$ 的逆存在，其中 $\bar{r}=g^{\bar{k}} \pmod p$. A 计算 $\bar{s}=x_A\bar{r}+\bar{k} \pmod q$.

(2) A 把 $(\bar{r},\bar{s})$ 发送给代理人 B.

(3) B 检验 $g^{\bar{s}}=\bar{r}y_A^{\bar{r}} \pmod p$ 如果成立便接受. 然后 B 计算 $s'=\bar{s}+x_B \pmod q$ 作为他的代理私钥.

签名阶段.

(1) B 任取 $k\in Z_q^*$，计算 $t=g^k \pmod p$ 并把 $(\bar{r},t)$ 发送给 R.

(2) R 任取 $a,b\in Z_q^*$，计算 $r=tg^by_B^{\bar{a}-b}(\bar{r}y_A^{\bar{r}})^{-a} \pmod p$，$e=H(r\|m) \pmod q$，$u=(\bar{r}y_A^{\bar{r}})^{-e+b}y_A^{\bar{e}} \pmod p$，$e^*=e-a-b \pmod q$. 如果 $r=0$，则 R 把 e^* 发送给代理人 B.

(3) 接到 e^*，B 计算 $s''=e^*s'+k \pmod q$，s'' 发送给 R.

提取阶段.

接到 s'' 后，R 计算 $s=b+s'' \pmod q$，得到的代理盲签名为 (m,u,s,e).

验证：$e=H(g^sy_B^{\bar{e}}y_A^eu\|m) \pmod q$.

5.2　分　析

验证等式中的签名数据 u 是个孤悬因子，因而该方案是不安全的，伪造方法如下：给定原始签名人 A 的公钥 y_A 和代理签名人 B 的公钥 y_B，一则任意消息 m，攻击者只需任选一个随机数 s，计算 $e=H(g^s\|m) \pmod q$ 并令 $u=y_A^{\bar{e}}y_B^e \pmod p$.

正确性：

$$g^sy_B^{\bar{e}}y_A^eu=g^sy_B^{\bar{e}}y_A^e(y_A^{\bar{e}}y_B^e)=g^s \pmod p.$$

6　Duc-Cheon-Kim 盲签名方案及分析

6.1　Duc-Cheon-Kim 盲签名方案

2003 年，Duc 等人在文献[12]中提出了一个盲签名方案.

系统建立. 产生两个安全的随机素数 p 和 q，计算 $N=pq$，$\varphi(N)=(p-1)(q-1)$. 产生一个随机数 λ 使得它与 $\varphi(N)$ 互素，选取 $a\in Z_N^*$，阶大于 λ. 选取 $r_0\in_R Z_\lambda^*$，$s_0, e\in_R Z_\lambda^*$，计算

$$V=a^{-r_0}s_0^{-\lambda} \bmod N,\ f_1=a^e \bmod N,\ v_1=V^2a^e \bmod N,$$

$$l=(2r_0-e)\div\lambda,\ r_1=(2r_0-e) \bmod \lambda,$$

$$s_1=a^ls_0^2 \bmod N.$$

则私钥为 $SK_1=(1,r_1,s_1,v_1,f_1)$，公钥为 $PK=(N,a,V,\lambda)$.

密钥更新. 若在 i 时刻的私钥为 (i,r_i,s_i,v_i,f_i)，选取 $e\in_R Z_N^*$，计算

$$v_{i+1}=v_i^2a^e \bmod N,\ f_{i+1}=f_i^2a^e \bmod N,$$

$$l=(2r_i-e)\div\lambda,\ r_{i+1}=(2r_i-e) \bmod \lambda,$$

$$s_{i+1}=a^ls_i^2 \bmod N,$$

则在 $i+1$ 时刻的私钥为 $(i+1,r_{i+1},s_{i+1},v_{i+1},f_{i+1})$.

签名. (1) Signer 选取 $t\in_R Z_\lambda^*$，$u\in_R Z_N^*$，计算

$x=a^t u^\lambda \bmod N$, 发送 x 给 User.

(2) User 选取盲因子 $\alpha, \gamma \in_R Z_\lambda^*$ 和 $\beta \in_R Z_N^*$, 计算

$x'=xa^\alpha \beta^\lambda v_i^\gamma \bmod N$, $c'=H(i \| f_i \| m \| x')$,

$c=c'-\gamma \bmod \lambda$,

发送 c 给 Signer.

(3) Signer 计算 $y=t+cr^i \bmod \lambda$, $\omega=(t+cr^i) \div \lambda$, $z=a^\omega u s_i^c \bmod N$, 发送 y, z 给 User.

(4) User 计算 $y'=(y+\alpha) \bmod \lambda$, $\omega'=(y+\alpha) \div \lambda$, $\omega''=(c'-c) \div \lambda$, $z'=a^{\omega'} v_i^{-\omega''} z\beta \bmod N$,

签名 $\sigma(m)=(f_i, c', y', z')$ (记号 $\div$ 表示的意义: 如果 $a=qb+r$, 则 $a \div b=q$).

验证. 给定公钥 (N, λ, a, V) 和签名 (f_i, c', y', z'), 计算 $v_i=V^{2^i} f_i \bmod N$, $x''=a^{y'} z'^\lambda v_i^{c'} \bmod N$. 如果 $c'=H(i \| f_i \| m \| x'')$, 则接受签名. 事实上,

$$x''=a^{y'} z'^\lambda v_i^{c'}=a^{y'} z'^\lambda (V^{2^i} f_i)^{c'}$$
$$=a^{y'}(a^{\omega'} v_i^{-\omega''} z\beta)^\lambda v_i^{c'}=a^{y'+\omega'\lambda}(a^\omega u s_i^c)^\lambda \beta^\lambda v_i^{c'-\omega''\lambda}$$
$$=a^{y+\omega\lambda+\alpha} u^\lambda s_i^{c\lambda} \beta^\lambda v_i^{c'-\omega''\lambda}=a^{t+cr_i+\alpha} u^\lambda s_i^{c\lambda} \beta^\lambda v_i^{c'-\omega''\lambda}$$
$$=xa^\alpha \beta^\lambda v_i^{c'-c-\omega''\lambda}=xa^\alpha \beta^\lambda v_i^\gamma=x' (\bmod N).$$

6.2 分 析

由于 f_i 和 V^{2^i} 拥有一个共同的指数 c', 从形式上看, 这似乎不是孤悬因子问题, 但实质上其破解方法和破解孤悬因子一样简单, 可以把它看作是孤悬因子的变型. 其特点是: 虽然某个签名数据位于底数上, 但同时还有别的数据和它拥有一个相同的指数. 此时, 伪造这个签名数据已经不再是面临解高次剩余问题. 具体方法如下: 给定签名人的公钥 (N, λ, a, V) 和 任意一则消息 m, 攻击者只需选取三个随机数 α, β, γ, 计算

$$f_i=V^{-2^i} a^\alpha \bmod N, \quad z'=\gamma \bmod N$$
$$c'=H(i \| f_i \| m \| a^\beta \gamma^\lambda), \quad y'=\beta-\alpha c' \bmod \lambda$$

于是所得的盲签名是 $\sigma(m)=(f_i, c', y', z')$.

正确性: $x''=a^{y'} z'^\lambda (V^{2^i} f_i)^{c'}=$

$$a^{\beta-\alpha c'} \gamma^\lambda (V^{2^i} V^{-2^i} a^\alpha)^{c'}=a^\beta \gamma^\lambda (\bmod N).$$

7 结束语

本文首次提出了签名数据中孤悬因子概念, 并结合几个协议具体分析了孤悬因子是如何破坏协议安全性的. 此外, 我们还明确指出了签名数据中的冗余问题. 值得强调的是, 签名中的冗余数据不仅浪费了通信系统的资源, 而且给协议留下了安全隐患.

致 谢 对审稿人提出的修改意见和建议, 我们表示衷心的感谢!

参 考 文 献

1 Nyberg K., Rueppel R.. Message recovery for signature schemes based on the discrete logarithm problem. In: Proceedings of EuroCrypto' 94, Lecture Notes in Computer Science 950, 1994, 182~193

2 Markus Michels, Patrick Horster. On the risk of disruption in several multiparty signature schemes. Advances in Cryptology ASIACRYPT, Berlin: Springer-Verlag, 1996, 334~345

3 Burmester M., Desmedt Y., Doi H., Mambo M., Okamoto E., MitsuruTada, Yoshifuji Y.. A structured ElGamal-type multisignature scheme. In: Proceedings of Public Key Cryptography2000, Berlin, 2000, 466~483

4 Boldyreva A.. Threshold signatures, multisignatures and blind signatures based on the gap diffie-hellman-group signatures scheme. In: Proceedings of Public Key Crypto2003, Berlin, 2003, 31~46

5 Kei Kawauchi, Mitsuru Tada. On the exact security of multi signature schemes based on RSA. In: Proceedings of Information Security and Privacy' 2003, Berlin, 2003, 336~349

6 Qi Ming, Xiao Guo Zheng. Security improvement of password authentication scheme and corresponding signature schemes. Journal of China Institute of Communications, 1998, 19(6): 61~64(in Chinese)
(祁 明, 肖国镇. 口令认证方案的安全性改进及其相应的数字签名方案. 通信学报, 1998, 19(6): 61~64)

7 Lv Ji Qiang, Wang Xin Mei. Improvement of two ID-based digital signature schemes. Journal of China Institute of Communications, 2003, 24(9): 128~131(in Chinese)
(吕继强, 王新梅. 两个基于身份的数字签名方案的安全性改进. 通信学报, 2003, 24(9): 128~131)

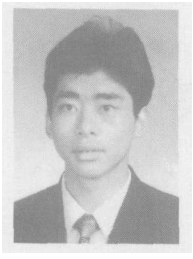
8 Zhang Jian-Hong, Wei Yong-Zhuang, Wang Yu-Min. Digital signature schemes based on RSA. Journal of China Institute of Communications, 2003, 24(8): 150~154(in Chinese)
(张键红, 韦永庄, 王育民. 基于 RSA 的多重数字签名. 通信学报, 2003, 24(8): 150~154)

9 Mambo M., Usuda K., Okamoto E.. Proxy signature: delegation of the power to sign messages. IEICE Transactions on Fundamentals, 1996, E79-A(9): 1338~1353

10 Chaum D.. Blind signature for untraceable payments. Advances in Cryptology: Crypto' 82, Berlin, 1982, 199~203

11 Tan Zuo-Wen, Liu Zhou-Jun, Tang Chun-Ming. A proxy blind signature scheme based on DLP. Journal of Software(China), 2003, 14(11): 1931~1935

12 Duc N.D., Cheon H.I., Kim K.. A forward-secure blind signature scheme based on the strong RSA Assumption. In: Proceedings of Information and Communications Secureity' 2003, Berlin, 2003, 11~21



CAO Zheng Jun, born in 1971, Ph. D. candidate. His research interests include information security and cryptography.

LIU Mu Lan, born in 1941, professor, Ph. D. supervisor. Her research interests include cryptography, information security and computer algebra.

Background

Digital signature is an important component of modern cryptography. It has greatly promoted the development of E-commerce. Lots of researchers have paid attentions to the design and analysis of signature schemes. But it's difficult to propose any common rules for designing signature schemes. In this paper, the authors introduce a concept, suspending factor, and definitely point out that it must be avoided in designing signature schemes. Besides, they analyze five signature schemes with suspending factor or redundant data and show that the schemes are not secure. It should be observed that redundant data in signature schemes not only confuses the description of protocol but also occupies the bandwidth. The research is supported by National Basic Research Program of China (973 Program) under grant No. 2004CB318000 and the National Natural Science Foundation of China under grant No. 90304012. The research group is with Institute of Systems Sciences, Chinese Academy of Sciences. The group members have published a number of papers in international and internal journals and conferences about applied mathematics, cryptography and computer sciences, etc.