第33卷 第22期 计 算 机 工 程 2007年11月
Vol.33 No.22 Computer Engineering November 2007
·安全技术· 文章编号：1000—3428(2007)22—0172—03 文献标识码：A 中图分类号：TP309

适用于 Smart Card 的有效电子现金解决方案

吕敏芳，曹珍富
(上海交通大学计算机工程系可信任数字技术实验室，上海 200400)

摘 要：提出了一个可以在低计算能力、低存储量的硬件(如 smart card)上实现的离线电子现金方案。该方案中，电子现金的存储量和效率都比一般的方案优很多，能满足电子现金所需要的安全特性，如匿名性、不可伪造性、多次消费检测等。该文还提出了一个类似的电子票据方案以及通过代理机构代理银行发布电子货币的方案。

关键词：电子现金；哈希树；Schnorr 签名算法

Efficient Electronic Cash Scheme for Smart Card

LV Min-fang, CAO Zhen-fu
(Trusted Digital Technology Lab, Department of Computer Engineering, Shanghai Jiaotong University, Shanghai 200400)

【Abstract】This paper presents an efficient scheme for off-line electronic cash that can be applied easily on hardware with limited computing power and limited storage (e.g. smart card). This scheme has high efficiency and requires small storage size, and it also has provable security properties of E-cash. It expands the scheme to an electronic tickets scheme and an agent electronic cash scheme.

【Key words】 electronic cash; hash tree; Schnorr signature

当前，电子支付正受到越来越多的关注。电子支付方式可以分为基于账户的和非基于账户的。基于账户的电子消费有其局限性：用户无法实现消费的匿名性；每次消费必须联网到银行或消费网关，银行或消费网可能成为效率瓶颈；对于小额消费，采用这样的方案成本也很高。

基于此，越来越多的研究倾向于非基于账户的、离线的、并能提供匿名性的电子支付方案，这种电子支付方案一般称为电子现金。匿名性的电子现金概念最早由Chaum等在 1982 年提出[1]，其后，文献[2～4]分别给出了自己的方案。多数电子现金方案在安全性上都是基于非对称密码以及非交互的零知识证明，其主要算法都需要数十次以上的模指数运算，难以在smart card上被低成本和有效地实现。

本文提出了一个高效的可以在 smart card 上实现的电子现金方案，该方案通过哈希树的使用使得其存储量与基于非对称密码的电子现金方案差不多，同时证明了该方案满足电子现金的一般安全性要求，还给出了一个类似的电子票据方案以及发布电子货币的方案。

1 基础知识

1.1 电子现金方案的一般模型及安全性要求

电子现金方案的参与者一般包括：用户，银行以及商家，其模型由以下 3 个协议组成：提取现金，消费现金以及报销(见图 1)。其过程如下：

提取现金(withdrawal)：用户从银行处通过各种方法获取一些电子货币(这里一个电子货币指的是代表一个货币单位的一个二进制字符串)；

消费现金(payment)：用户通过一些交互向商家证明其电子货币的合法性，并消费一个(或一些)电子货币，获取其所需要的货物或服务。

报销(deposit)：商家将用户与其交互过程中所产生的“证据”交给银行，经验证后，银行将这些电子货币所代表的金额转换成其账户中的款项。

如果在用户消费过程中商家无需与银行在线交互，则称这样的一个电子现金方案是离线的。

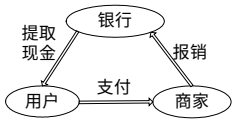

图 1 电子现金的一般模型

电子现金的安全性要求包括：

不可重复使用性(unreuseability)：如果一个电子货币被重复使用了 2 次，那么这个用户(或者是商家)可以被有效地找出并被证明；

不可伪造性(unforgeability)：无论用户与商家进行了多少次的提取、消费和报销，他们(单独或联手)都无法成功地用概率多项式图灵机算法生成一个有效的电子货币；即该伪造的电子货币在报销时不会被银行接受。

匿名性(anonymous)：如果用户按照协议正确地使用电子货币，则他的消费记录不会被别人知道。

1.2 随机数生成函数与哈希树

在本文所提出的电子现金方案中需要一个随机数生成函数来构成一棵哈希树。真正意义上的随机往往是很难达到的，一般情况下人们使用伪随机数，伪随机数要满足：通过概率

基金项目：国家杰出青年科学基金资助项目“计算机通信安全”(60225007)；国家自然科学基金资助项目“基于移动代理的密码理论体系及其应用”(60572155)；上海市科委基金资助项目“安全电子贸易体系结构构建”(04DZ07067)；华为专项基金资助项目“网络信息安全技术专项研究”

作者简介：吕敏芳(1982－)，女，硕士研究生，主研方向：密码学，电子商务安全，电子现金；曹珍富，博士、教授、博士生导师

收稿日期：2006-11-26 **E-mail：**zfcao@sjtu.edu.cn

多项式图灵机算法无法区分其与真正意义上的随机数。

定义1 带密钥的伪随机数生成函数 $G_a(s,J)$。该算法接收输入 s 与计数器 J，以及密钥 a，运行一个确定性多项式算法，其输出与一个随机数生成函数的输出对于多项式概率图灵机是不可区分的。

满足条件的带密钥的伪随机数生成函数应该很多，比如 AES 也可以看成是一个伪随机数生成函数。哈希树的概念是 T. Sander 等于 1999 年在文献[2]中提出的。

定义2 (哈希树) 给定一个域 D 和一个已知的哈希函数 $hash: D\times D\to D$，一棵哈希树 (T,val) 即一棵带有加权值 V 的平衡二叉树 T，以及一个函数 $val: V\to D$。对于任一节点 v，设其两个子节点分别为 v_1,v_2，则其权值为 $val(v)=hash(val(v_1),val(v_2))$。哈希树唯一接受的操作是 $Update(leaf,w)$，该操作将叶子节点 $leaf$ 的值改为 w，并依次修改相应的内部节点以及根节点的值。

图 2 给出了一个高度为 2 的哈希树的例子。

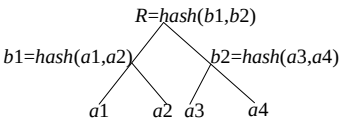

图 2 高度为 2 的哈希树

1.3 Schnorr 数字签名

这里用到的适用于 smart cards 的 Schnorr 数字签名方案是 Schnorr 在文献[6]中提出的，它由初始化、签名、验证 3 个部分组成。

初始化：输入安全参数 n，生成签名私钥 (p,q,g,u,h)，其中 q 是质数，$|q|=n$，$p=\alpha q+1(\alpha>1)$ 也是一个质数，$g\in Z_p$，$g\neq 1$ 且 $g^q\equiv 1(\bmod p)$ (即 $G_q=\{g\}$ 是 Z_p 的一个子集)，$u\in Z_q$，h 是一个哈希函数；验证公钥为 $(p,q,g,I\equiv g^u(\bmod p),h)$。

签名：对于需要签署的信息 m，随机选择 $x\in_R Z_q$，并计算：$y\equiv g^x(\bmod p)$，$e=h(m,I,y)(|e|>n)$ 以及 $r\equiv eu+m(\bmod q)$。并输出签名 $\sigma=(m,y,e,r)$。

验证：对于 $\sigma=(m,y,e,r)$，验证等式 $g^r\equiv I^e y(\bmod p)$ 和 $e=h(m,I,y)$ 是否成立。

以目前的计算能力来看，当 q 的位数为 160bit，p 的位数为 1 024bit 时可以保证数字签名的安全性。

2 适用于 Smart Card 的有效电子现金解决方案

本节中给出具体的电子现金解决方案。该方案由初始化、提取现金、消费现金、报销 4 部分组成。

2.1 初始化参数

(1)银行按照 Schnorr 数字签名的初始化方法得到合适的公钥 (p,q,g,I,h) 和私钥 (p,q,g,u,h)，并将公钥公布。

(2)银行选择并公布安全的随机数生成函数：$G: D_1\times D_2\times D_3\to\{0,1\}^l$；$D_1,D_2,D_3$ 分别是 s,a,i 的取值域。

(3)银行选择并公布满足单向安全性的哈希函数：$H:\{0,1\}^l\to\{0,1\}^{l'}$。

2.2 提取

如图 3 所示，在提取现金的协议中，银行与用户在一个安全信道中如下交互：

(1)银行与用户首先互相认证对方的身份，然后银行发送给用户一个全局唯一的序列号 $s\in D_1$。

(2)用户通过函数 $z_i=H(G_a(s,i))(i=1,2,\cdots,k)$ 生成一系列的随机数 $\{z_i\}$，并将其发送给银行；其中 $a\in D_2$ 是用户的秘密；$i\in D_3$ 是一个计数器；k 是一个安全参数，其取值将在第 4 节中进行讨论。

(3)银行将这些 $\{z_i\}$ 作为哈希树的叶子来生成一棵哈希树，计算该哈希树的根的值，并对 s 与该根值进行签名，然后将签名 σ 发送回给用户。

这样 (s,a,σ) 即为用户获得的一个电子货币。用户将其保存；同时银行也保存 (s,σ)。

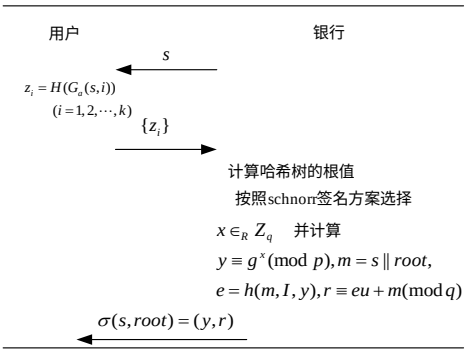

图 3 提取电子现金协议过程

2.3 消费

如图 4 所示，消费过程中，用户与商家如下交互：

(1)用户重新计算 $\{z_i\}$。

(2)商家计算由 $\{z_i\}$ 生成的哈希树的根值并验证签名的合法性。若合法，则随机生成 $\{b_i\}$，其中 $b_i\in_R\{0,1\},\sum b_i=k/2$，将 $\{b_i\}$ 发送给用户。

(3)对于 $b_i=1$ 的，用户发送相应的 $G_a(s,i)$ 给商家，以此证明自己对于该电子货币的所有权。商家检验 $G_a(s,i)$ 与 z_i 是否匹配，若匹配，则接受此次消费，并将这些 $\{z_i\}$ 以及 $G_a(s,i)$ 作为证据存储。

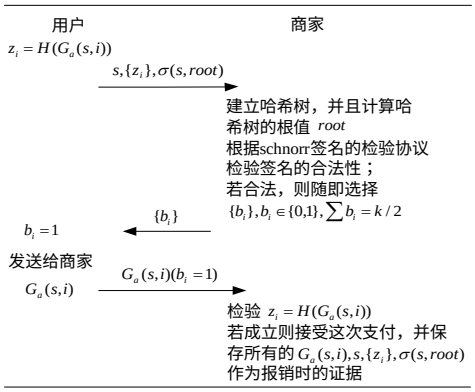

图 4 消费电子现金协议过程

2.4 报销

在报销过程中，商家与银行如下交互：(1)商家将电子货币 (s,σ) 以及消费过程中获得的所有证据提交给银行。(2)银行检测 s，若 s 存在且未被使用过则验证自己的签名，正确的话则对这个电子货币进行报销，并保存这些证据，若验证失败或 s 不存在则拒绝报销；否则检查已保存的证据：若 $\{b_i\}$ 都相等，则判断是商家重复报销；若 $\{b_i\}$ 不相等，则判断是用户重复消费了一个电子货币。

需要说明的是：这里都是概率多项式判断，但可以通过 k 的大小来控制这个概率，使得判断能以很高的概率获得成

功，具体的分析将在第 4 节中给出。

3 安全性证明

定义 3 (Schnorr 数字签名算法的安全性) 在 Schnorr 数字签名中伪造一个签名成功的概率 $\le \varepsilon_1$。

关于 Schnorr 数字签名算法的安全性在文献[6]中有比较详细的介绍，这里只是定义其伪造成功概率，不作证明。

定义4(哈希函数的安全性) 一个安全的哈希函数应该满足：已知 y，找到 x 满足 $y = H(x)$ 的概率 $\le \varepsilon_2$。

下面给出系统的安全性证明：

定理 1(重复消费检测) 如果一个用户重复消费了一个电子货币 2 次，则他不被银行检测到的概率是 $\le \varepsilon_1 + (C_k^{k/2})^{-1}$。

证明 一个用户想重复消费一个电子货币而又不被发现，一种方法是商家两次给其的 $\{b_i\}$ 是相同的，而这一概率 $= (C_k^{k/2})^{-1}$；另一方法就是伪造 Schnorr 签名，即对于同一个 s，用不同的 a 生成 $\{z_i\}$，并对该哈希树的根值和 s 伪造签名，而这一概率 $\le \varepsilon_1$。

定理 2(重复报销检测) 如果一个商家重复报销了一个电子货币两次，则他不被银行检测到的概率 $\le \varepsilon_1 + \varepsilon_2$。

证明 一个商家想要报销一个电子货币两次而不被发现，那么他两次提交的证据必须是不一样的，一种方法是他至少找到一个 $z_i (b_i \ne 1)$ 的原象，这一概率 $\le \varepsilon_2$；另一种方法是对于 s，商家随机选择 a 生成 $\{z_i\}$，并对该哈希树的根值和 s 伪造一个 Schnorr 签名，而这一概率 $\le \varepsilon_1$。

定理 3(无法伪造性) 一个用户或者商家伪造一个电子货币的概率 $\le \varepsilon_1$。

证明 伪造一个电子货币的方法有两种：一种是已知 s，选择另外的 a 按协议生成正常的 $G_a(s,i)$ 与 $\{z_i\}$，然后对哈希树的根值和 s 伪造一个银行的签名，显然伪造的概率 $\le \varepsilon_1$；另一种方法就是猜测一个存在且未被使用的 s，随机选择 $root$，并对它们产生一个有效的签名，同时还要能够获得该根值的一组可能的 $\{z_i\}$，即至少找到一个 z_i 的原象，显然伪造的概率小于前者。综上，一个用户或者商家伪造一个电子货币的概率 $\le \varepsilon_1$。

定理 4(匿名性) 一个用户可以匿名地进行电子货币的消费。

证明 由于商家不知道序列号所对应的用户信息，因此他无法知道用户的身份信息；同时在消费协议中并未记录用户购物信息的相关证据，因此商家在稍后也无法向别人证明用户购买商品或服务的具体信息。

4 方案分析

显然本方案的安全性建立在哈希函数的单向性、Schnorr 签名的安全性以及 $\{b_i\}$ 选择的随机性上。前两者的安全性已经在很多研究中被讨论，现在一般的结论是：密钥长度在 160bit(即 2.1 节中 $l' = 160$)及以上的哈希函数如 SHA-1 可以满足强碰撞安全；而 q 的位数为 160bit， p 的位数为 1 024bit 的 Schnorr 签名可以满足数字签名的安全性要求。这里主要讨论安全参数 k 取多大的时候可以保证该方案的安全性。假设用户要重复消费一个电子货币，则必须使得两个商家给他的 $\{b_i\}$ 相同，这个概率为 $(C_k^{k/2})^{-1}$。如果限制这个概率小于一亿分之一，即 $(C_k^{k/2})^{-1} < 10^{-10}$，则

$$(C_k^{k/2})^{-1} = (\frac{k(k-1)\cdots k/2}{1\cdot 2\cdots k/2})^{-1} = \frac{k}{k/2}\cdot\frac{k-1}{k/2-1}\cdots\frac{k/2}{1} \ge 2^{-k/2}$$

$$2^{-k/2} \le 10^{-10} \le 2^{-31} \Rightarrow k \ge 62$$

即 k 取 64 即可充分满足安全性要求。

而一个电子现金方案的有效性包括：一个电子货币的长度，运行提取，消费以及报销协议时所需要的计算量及通信量。这里主要考虑用户参与的过程，即提取与消费协议，因为商家与银行的设备通常没有计算能力与存储量上的限制。

方案中一个电子货币由 (s,σ) 组成，而 a 作为用户使用随机数生成函数的密钥每次都相同，只需存储一遍即可，对于 Schnorr 数字签名来说，签名长度在(1 024+160) bit 即可获得可靠的安全性，为了保证 s 的全局唯一性，这里假设 s 的长度也为 1 024 bit，则该方案所需要的长度为 2 208bit，只有 276B 长。

在运行速度上，在提取现金时银行需要做一个数字签名，用户需要验证该签名，即双方都需要 1 次指数运算；在消费现金时，只需要商家做 2 次指数运算验证银行的签名。

在通信量方面，在提取现金时，需要传送 k 个哈希值和 1 个数字签名，假设方案采用 SHA-1 作为哈希函数，则一共要传输 $160k$+2 208b。现在的芯片读写速度一般都在几十兆字节每秒；假设芯片的传输速度为 10Mb/s，取 k=64，则在用户提取 1 000 个电子货币的情况下，所需时间约为(160*64+2 208)*1 000/10M=1.19s；在消费时通信量为一个电子货币加上 k 个哈希值加上 k/2 个随机数值，总共为 $160k$+1 024* k/2+2 208。还是假设芯片的传输速度为 10Mb/s，取 k=64，则在用户消费 100 个电子货币的情况下，所需时间约为(160*64+1 024*32+2 208)*100/10M=0.431s。显然这个速度是可以接受的。

5 扩展方案一——有效的电子票据方案

在电子票据的解决方案中参与者可以缩减为商家→用户→商家，即等于电子现金方案中的银行与商家是一个实体。这样的话无需对哈希树的根值进行数字签名，商家可以在自己的数据库中记录下该根值，在用户使用电子票据的时候进行比对即可。

由于与电子现金相比，电子票据只需要记录一个“ 记录 ”，因此有足够的空间可以记录下所有的 $\{z_i\}$ 以及 $G_a(s,i)$，即一个电子票据由 $\{z_i\}$, $G_a(s,i)$ 组成，大小约为(128*k)B。这样在使用电子票据的时候，不需要再进行任何运算。显然这样可以节省用户制造电子票据的成本，而且这类电子票据也很容易在各种移动设备上实现，如手机、PDA。

6 扩展方案二——代理银行发布电子货币的方案

在前文所提到的电子现金方案中银行需要使用自己的公私钥对做数字签名。在实际应用中银行可以对一些代理机构进行授权，给予他们一对公私钥对使得他们可以代理银行发售电子货币；只需要在电子货币的存储信息中增加一些信息来注明是哪一个代理结构，商家就可以使用相应的公钥来验证该电子货币的合法性，银行也可以通过此与相应的代理机构进行结算。

7 结束语

本文提出了一个有效的电子现金解决方案，该方案在哈希函数的单向安全性及 Schnorr 签名的安全性假设下满足电子现金的安全性要求，同时计算量足够小，所需的空间足够小，使得它可以方便地应用到 Smart Card 等计算能力不强，以及存储量有限的硬件设备上，并给出了该方案的两个扩展。 (下转第 189 页)