第七章、盲签名及其在区块链中的应用

何德彪

武汉大学

国家网络安全学院

目　录

目　录

7.1 盲签名的概念

7.1.1 互联网的规模

7.1 盲签名的概念

7.1.1 互联网的规模

2020年4月28日，中国互联网络信息中心发布第45次《中国互联网络发展状况统计报告》.

中国网民

- 规模达9.04亿，较2018年底增长7508万，
- 互联网普及率为64.5%，较2018年底提升了4.9个百分点

中国手机网民

- 规模达 8.97亿，较2018 年底增加7992 万人
- 网民中使用手机上网人群占比由2018年底的98.6%提升至99.3%

中国农村网民

- 农村网民规模达到2.55亿，占28.2%
- 城镇网民规模7.49较2018年底增加4200万人

上网方式

- 台式机、笔记本、平板电脑接入互联网比例分别占42.7%，35.1%和29.0%
- **手机上网使用率为97.2%**

7.1 盲签名的概念

7.1.2 在线支付

在线支付：

银行APP

支付宝

微信

不具备匿名性，容易追踪到某笔交易金额的来源和购买的商品.

电子现金

具备匿名性，无法追踪电子现金来源.

- 假名
- 零知识证明
- 环签名
- 盲签名
- .....

7.1 盲签名的概念

7.1.3 电子现金

现金是通用的交换媒介，也是对其他资产计量的一般尺度. 它可以立即投入流通且具有普遍的可接受性，并可有效地用来购买商品、货物、劳务或偿还债务.

7.1 盲签名的概念

7.1.3 电子现金

电子现金 (E-Cash)：(Electronic Cash) 也可称为电子货币 (E-money) 或数字货币(Digital Cash) . 它可以看作是真实现金的模拟. 电子现金以数字信息形式存在，通过互联网流通.

7.1 盲签名的概念

7.1.3 电子现金

电子现金优于真实现金之处在于它安全、超距、迅速、低成本、匿名性、精确性等，这大大强化了现金的可移动性.

抢劫

现金易被抢劫，必须小心放置和防盗，现金越多，风险越大，其安全性投资也越大.

转运

现金转运成本高，美国转运现金费用每年高达$600亿.

伪钞

高质量彩色复制和伪造技术使现金越来越不安全，伪钞可以破坏国家的经济和政府的稳定.

7.1 盲签名的概念

7.1.3 电子现金

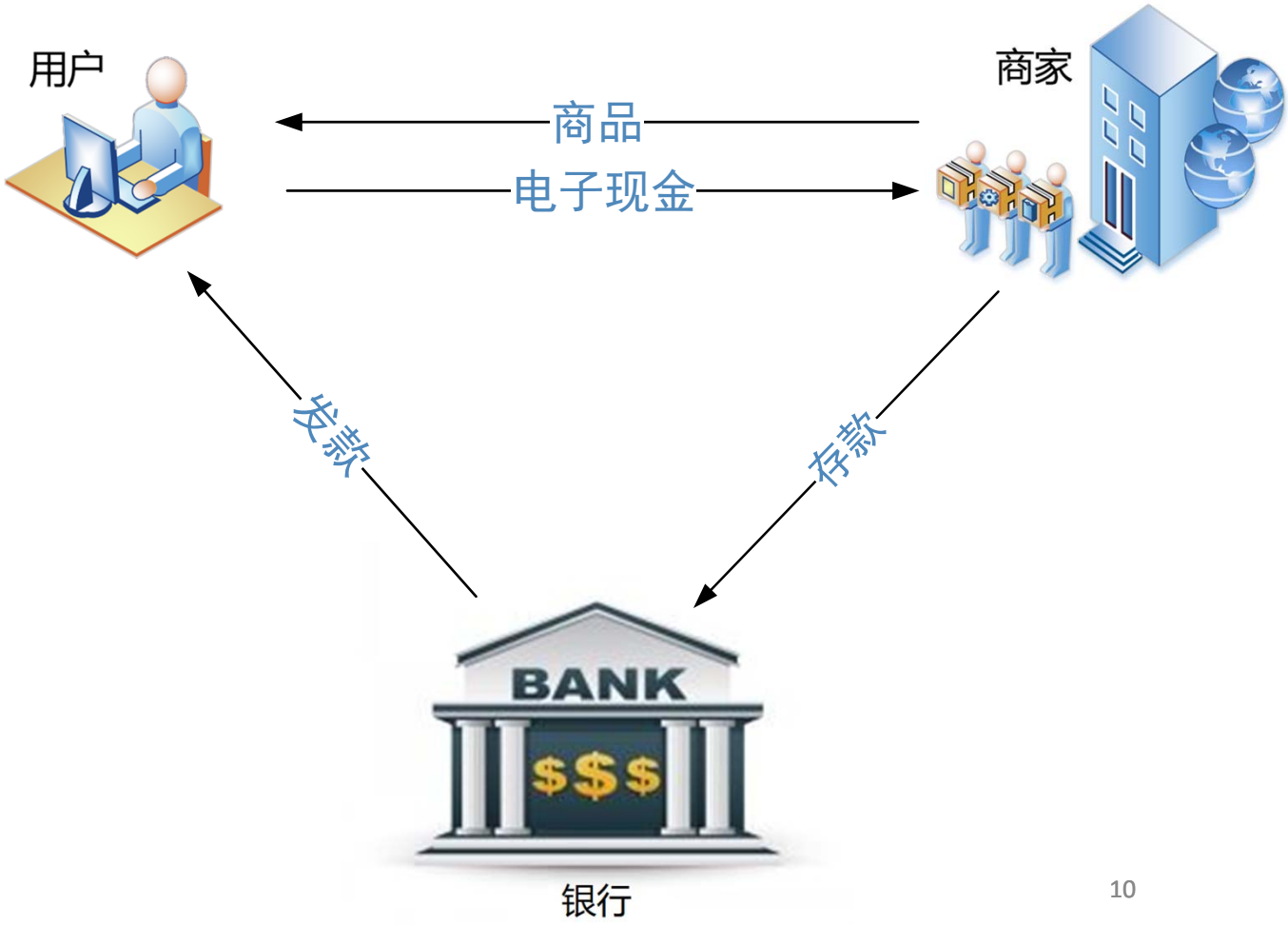
电子现金通过信息网络系统和公共信息平台实现流通、存取、支付. 在电子现金的支付中有三方参与：**银行、用户、商家.**

电子现金在线交易一般需要以下三个基本阶段：

① 取款阶段：用户从自己的银行账户上提取数字现金.

② 支付阶段：用户使用数字现金从商店中购买商品.

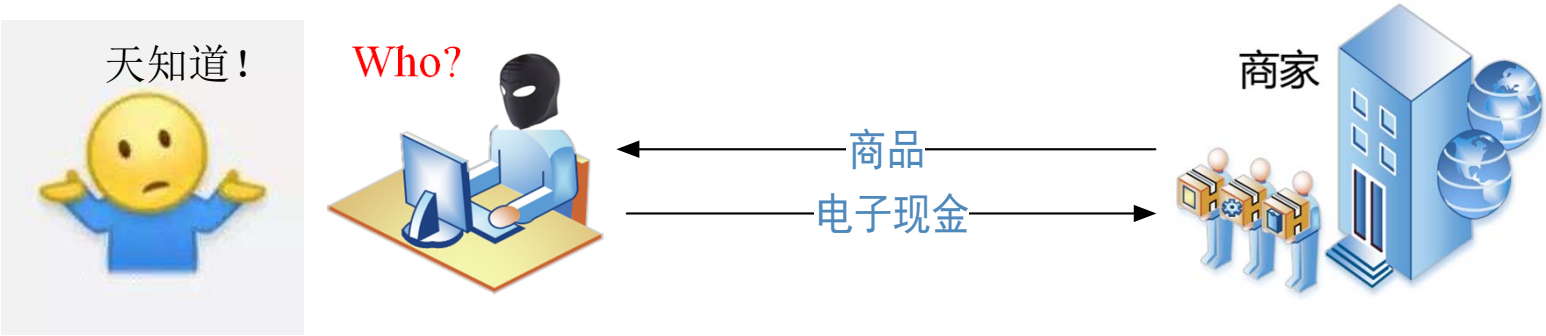
③ 存款阶段：用户及商家将数字现金存入到自己的银行账户上.



7.1 盲签名的概念

7.1.3 电子现金

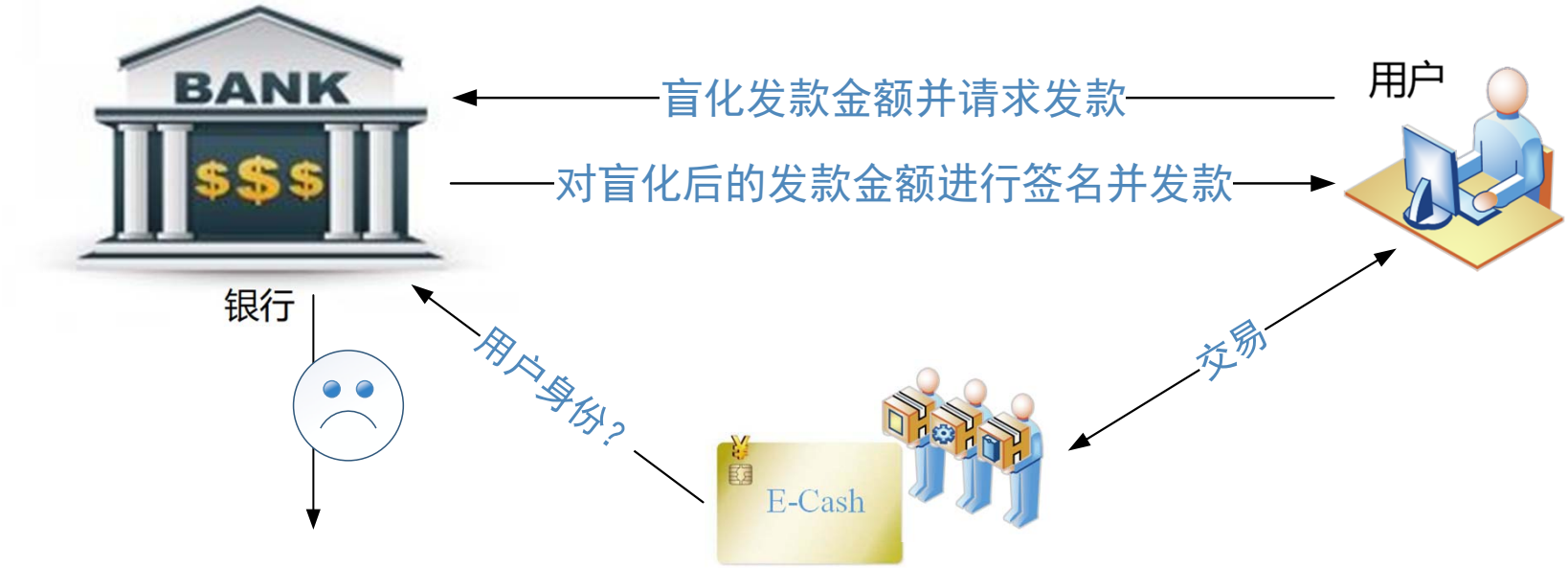
电子现金像真实现金一样，用户不希望别人知道自己购买过什么样的商品. 为了保护用户购买商品、服务时的隐私，接收方不应获得支付方的任何身份信息.



7.1 盲签名的概念

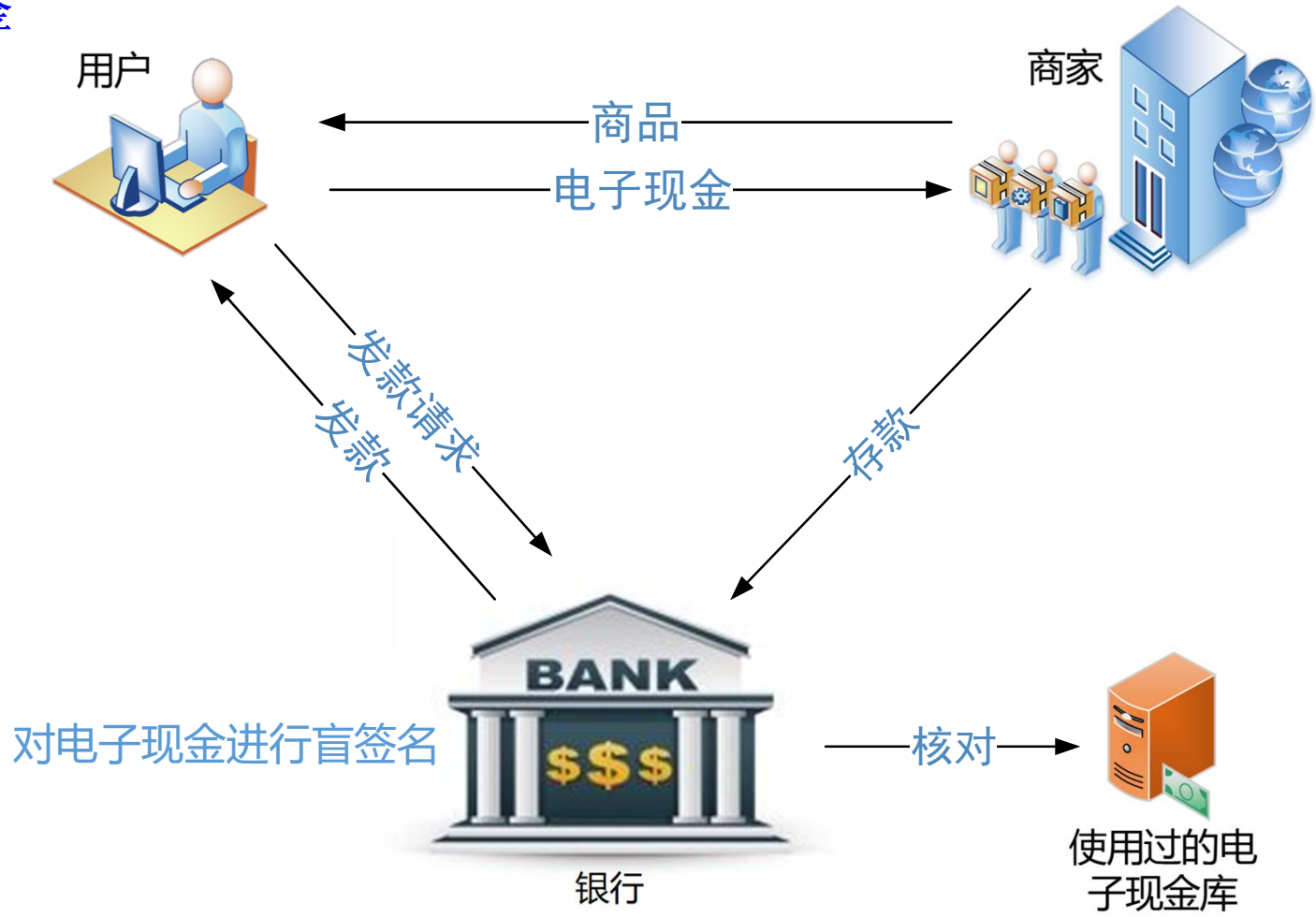
7.1.3 电子现金

盲签名技术为解决上述问题而生. 盲签名可以实现银行向合法用户发放有效的电子现金，使用户可以使用有效的电子现金进行交易，银行本身也无法通过电子现金追踪到该电子现金是属于哪个用户的.



7.1 盲签名的概念

7.1.3 电子现金



7.1 盲签名的概念

7.1.4 盲签名的概念

盲签名的概念首先由 David Chaum 于1982年提出，盲签名实现了**签名者对发送者的消息进行签名，却不能知道签名者消息的具体内容**.

所谓盲签名，就是将文件放入带有复写纸的信封，签名者在信封上对文件进行签名而不知道文件的具体内容.

众所周知，电子货币是比特串，容易被复制和多次使用. 为了避免这种双重支付问题，David Chaum提出了一个**基于RSA算法的盲签名方案**，并用它构造了一个基于盲签名的在线电子银行系统，银行可以在线实时检查Coin是否已经花掉.

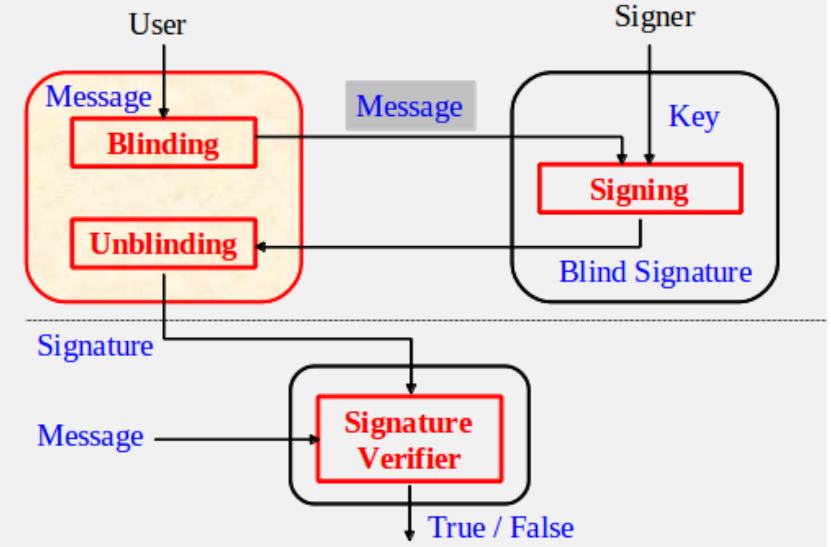
虽然这个方案可以解决双重支付问题，但是实用性不高，因为银行数据库相当大，实时搜索需要很多时间.

7.1 盲签名的概念

7.1.4 盲签名的概念

一个盲签名方案一般包括四个过程：

- **系统初始化**：产生盲签名方案中的所有系统参数；
- **密钥对生成**：产生用户的私钥和公钥；
- **签名**：用户利用签名算法对消息签名，签名过程可以公开也可以不公开，但一定包含仅签名者才拥有的私钥：
  - ✓ **盲化**：用户将盲化因子注入待签名的消息；
  - ✓ **盲签名**：签名者对盲化过的消息进行签名；
  - ✓ **去盲**：用户从盲化签名(对盲化过的消息)去除盲化因子，获得去盲后的签名(待签名消息)；
- **验证**：验证者利用公开的系统参数、验证方法和签名者的公钥对给定消息的签名进行验证.



7.1 盲签名的概念

7.1.5 盲签名的性质

一个盲签名除了满足一般的数字签名性质外，还需要满足下面的4个性质：

- **完备性**：如果签名者和接收者都遵守盲签名生成算法，那么盲签名验证算法将始终接受签名生成算法输出的签名(即它将始终输出“Accept”).
- **不可伪造性**：在没有私钥的情况下，任何人不能伪造一个合法的签名.
- **盲性**：签名者对其所签署的消息是不可见的，即签名者不知道他所签署消息的具体内容.
- **不可追踪性(广义的盲性)**：当签名消息被公布后，签名者无法知道这是他哪次的签署的，既使保存了他所签的每一个盲签名的记录.

7.1 盲签名的概念

7.1.6 盲签名的分类

根据算法的功能，盲签名可以分为以下5类：

- **完全盲签名**：由David Chaum于1987年提出，完全盲签名实现了签名的完全匿名性. 任何人无法准确的追踪到签名.
  - ✓要求银行实时在线；
  - ✓导致完美犯罪；
- **限制性盲签名**：解决了电子现金的二次花费问题，限制性盲签名将用户的身份信息嵌入到签名中，当用户二次花费时可以准确的追踪到用户的身份信息.
  - ✓防止重花费；
  - ✓实现了离线银行系统；

7.1 盲签名的概念

7.1.6 盲签名的分类

- **公平盲签名**：“公平”是区别于普通盲签名的不可追踪性而言的，相对于一般盲签名，公平盲签名引入一可信的第三方，以便在需要对签名进行追踪的情况下，为签名者准确地追踪到签名.
  - ✓防止了不法分子进行敲诈勒索以及洗黑钱等非法行为；
- **部分盲签名**：在盲签名的基础上增加一个公共信息，该公共信息由签名者和用户共同在签名之前商议得到，由签名者在签名过程中把这些公共信息嵌入签名中，且该公共信息不能被删除或非法修改.
  - ✓签名前需要签名者和用户共同协商一个公共信息；
  - ✓公共信息可以包括电子货币的有效期、面值等；

7.1 盲签名的概念

7.1.6 盲签名的分类

➢ **群盲签名**：群签名和盲签名的有机结合,兼有群签名和盲签名的特性.

✓可以实现多个银行匿名地分发不可跟踪的电子现金；

✓签名者无法将消息签名对与某个具体的签名协议联系起来；

目　录

7.2 几个经典盲签名算法

7.2.1 基于RSA签名的盲签名算法

David Chaum 于1982年提出盲签名的概念，并利用RSA算法设计了第一个盲签名方案. 该方案的安全性基于大整数分解问题，这里做简介介绍，具体见：

David Chaum. "Blind Signatures for Untraceable Payments." Proc Crypto (1982):199-203.

1. 密钥生成

签名者执行以下步骤生成密钥对：

① 签名者选择两个大素数p, q，计算$n = pq$，$\phi(n) = (p-1)(q-1)$；

② 签名者选择两个大整数e, d，满足$ed \equiv 1 \, mod \ \phi(n)$，$gcd\big(e, \phi(n)\big) = 1$；

③ 签名者保存私钥(d, n)，并公开公钥(e, n)和安全哈希函数$H: \{0,1\}^* \rightarrow Z_n^*$.

7.2 几个经典盲签名算法

7.2.1 基于RSA签名的盲签名算法

2. 盲化

① 用户选择随机数$r \in_R Z_n^*$，计算$m' = r^e H(m) \ mod \ n$，其中m是待签名的消息；

② 用户将盲化的消息m'发送给签名者.

3. 签名

签名者计算$\sigma' \equiv m'^d \ mod \ n$，并将$\sigma'$发送给用户.

4. 去盲化

用户计算$\sigma \equiv \sigma' r^{-1} \ mod \ n$.

7.2 几个经典盲签名算法

7.2.1 基于RSA签名的盲签名算法

5. 签名正确性验证

用户通过验证$\sigma^e \equiv H(m)\ mod\ n$.

6. 安全性分析

- **完备性**：如果用户和签名者诚实的执行协议，那么：

  $\sigma^e \equiv (\sigma' r^{-1})^e \equiv (m'^{ed}) r^{-e} \equiv m' r^{-e} \equiv H(m) r^e r^{-e} \equiv H(m)\ mod\ n$,

  有效签名总是能够通过验证.

- **盲性**：使用的随机数——盲化因子r，能够保证每个消息签名对(m, σ)，在统计学上完全独立于签名者在签名过程中可以看到的盲化消息签名对(m', σ').

7.2 几个经典盲签名算法

7.2.1 基于RSA签名的盲签名算法

➢不可追踪性

根据$m' = r^e H(m) \bmod n$和$\sigma' \equiv m'^d \bmod n$，可以发现基于RSA签名的盲签名算法不能提不可追踪性性，具体攻击如下：

① 签名者保存所有盲签名过程中的记录(m', σ');

② 给定(m, σ)，签名者计算$r' = \sigma'/H(m)^d \bmod n$和$m'' = r'^e H(m) \bmod n$；

③ 签名者判定m'和m''是否相等;

④ 如果相等则找到了签名记录，否则遍历所有记录(m', σ').

目　录

7.2 几个经典盲签名算法

7.2.2 基于Schnorr签名的盲签名算法

1. Schnorr签名回顾

Okamoto T. Provable secure and practical identification schemes and corresponding digital signature schemes. *Crypto'92*, pp. 31-52, 1992. (第一个基于离散对数的盲签名算法)

Schnorr签名方案是一个短签名方案，它是ElGamal签名方案的变形，其安全性是基于离散对数困难性和哈希函数的单向性的. 假设p和 q是大素数，是q能被p-1整除，q是大于等于160 bit的整数，p是大于等于512 bit的整数，保证GF(p)中求解离散对数困难；g是GF(p)中元素，且$g^q \equiv 1 \bmod p$.

➢ **密钥生成：**

① Alice选择随机数x为私钥，其中$1<x<q$；

② Alice计算公钥$y \equiv g^x \pmod p$；

7.2 几个经典盲签名算法

7.2.2 基于Schnorr签名的盲签名算法

➢ 签名算法

① Alice首先随机数k，这里$1<k<q$；

② Alice计算$e = h(M, g^k \bmod p)$

③ Alice计算$s = k + x \cdot e \ (\bmod q)$

④ Alice输出签名(e, s)

➢ 验证算法

① Bob计算$g^k \bmod p = g^s \cdot y^e \bmod p$.

② Bob验证$e = h(M, g^k \bmod p)$，如果相等这输出”Accept”，否则输出" Reject"

7.2 几个经典盲签名算法

7.2.2 基于Schnorr签名的盲签名算法

2. 盲签名算法描述

Signer		*User*
随机选择$k' \in Z_q^*$ 计算$w' = g^{k'}$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ 计算$w = g^{\alpha} \cdot y^{\beta} \cdot w'$ 计算e=$h(w, m)$ 计算$e' = e + \beta \bmod q$
	← e'	
计算$s' = k' + x \cdot e' \bmod q$	s' →	
		计算$s = s' + \alpha \bmod q$ 输出(e, s)

7.2 几个经典盲签名算法

7.2.2 基于Schnorr签名的盲签名算法

➢ 完备性

$w = g^{\alpha} \cdot y^{\beta} \cdot w' = g^{\alpha} \cdot (g^{x})^{\beta} \cdot g^{k'}$

$= g^{k'+\alpha+x\cdot\beta} \bmod p$

$s = s' + \alpha = k' + x \cdot e' + \alpha$

$= k' + x \cdot (e + \beta) + \alpha$

$= k' + \alpha + x \cdot \beta + x \cdot e \bmod q$

- $w = g^{k} \bmod q$
- $s = k + x \cdot e \bmod q$

与原始Schnorr签名相比较，这里的$k' + \alpha + x \cdot \beta$即是原始MDSA签名中的$k$，因此生成的签名$(e, s)$可以使用原始MDSA验证算法来验证.

7.2 几个经典盲签名算法

7.2.2 基于Schnorr签名的盲签名算法

➢ 盲性

很显然.

➢ 不可追踪性

根据$w = g^{\alpha} \cdot y^{\beta} \cdot w'$，$e' = e + \beta \bmod q$，$s' = k' + x \cdot e' \bmod q$，$s = s' + \alpha \bmod q$和$e=h(w, m)$，可以发现基于Schnorr签名的盲签名算法不能提供不可追踪性，具体攻击如下：

① 签名者保存所有盲签名过程中的记录(k', w', e', s');

② 给定(m, e, s)，签名者计算$\alpha' = s - \alpha \bmod q$和$\beta' = e' - e \bmod q$；

③ 签名者计算$w'' = g^{\alpha'} \cdot y^{\beta'} \cdot w'$并判定$e$和$h(w'', m)$是否相等;

④ 如果相等则找到了签名记录，否则遍历所有记录(k', e', w', s').

目　录

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

J. Camenisch, J.-M. Piveteau, and M. Stadler, Blind signatures based on the discrete logarithm problem. *Eurocrypt '94*, pp. 428-432, 1994.

DSA的主要参数：

- 全局公开密钥分量，可以为用户公用
  - ✓ p：素数，要求$2^{L-1}<p<2^{L}$,$512<=L<1024$，且L为64的倍数
  - ✓ q: $(p-1)$的素因子，$2^{159}<q<2^{160}$，即比特长度为160位
  - ✓ $g:=h^{(p-1)/q} \bmod p$.其中h是一整数，$1<h<(p-1)$且$h^{(p-1)/q} \bmod p>1$
- 用户私有密钥
  - ✓ x：随机或伪随机整数，要求$0<x<q$
- 用户公开密钥
  - ✓ y：$=g^{x} \bmod p$

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

DSA签名过程：

① 用户随机选取k

② 计算$e=h(M)$;

③ 计算$r=(g^k \bmod p) \bmod q$

④ 计算$s=k^{-1}(e+x \cdot r) \bmod q$

⑤ 输出(r, s)，即为消息M的数字签名.

DSA验证过程：

① 接收者收到M, r, s后，首先验证$0<r<q$, $0<s<q$；

② 计算$e=h(M)$;

③ 计算$u=(s)^{-1} \bmod q$

④ 计算$u_1=e \cdot u \bmod q$

⑤ 计算$u_2=r \cdot u \bmod q$

⑥ 计算$v=[(g^{u1} \cdot y^{u2}) \bmod p] \bmod q$

⑦ 如果$v=r$，则确认签名正确，否则拒绝.

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

MDSA签名过程：

① 用户随机选取k

② 计算$e=h(M)$;

③ 计算$r=(g^k \bmod p) \bmod q$

④ 计算$s=k\,e+x \cdot r \bmod q$

⑤ 输出(r, s)，即为消息M的数字签名.

MDSA验证过程：

① 接收者收到M, r, s后，首先验证$0<r<q$, $0<s<q$；

② 计算$e=h(M)$;

③ 计算$u=e^{-1} \bmod q$

④ 计算$u_1=s \cdot u \bmod q$

⑤ 计算$u_2=-r \cdot u \bmod q$

⑥ 计算$v=[(g^{u1} \cdot y^{u2}) \bmod p] \bmod q$

⑦ 如果$v=r$，则确认签名正确，否则拒绝.

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

Signer		*User*
随机选择$k' \in Z_q^*$ 计算$w' = g^{k'}$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ $e=h(m)$ 计算$w = w'^{\alpha} \cdot g^{\beta} \bmod p$ 计算$e' = \alpha \cdot e \cdot w' \cdot w^{-1} \bmod q$
	← e'	
计算$s' = k' \cdot e' + x \cdot w' \bmod q$	s' →	
		计算$r = w \bmod q$ 计算$s = s' \cdot w \cdot w'^{-1} + \beta \cdot e \bmod q$

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

➢ 完备性

- $r = g^k \bmod q$
- $s = k \cdot e + x \cdot w \bmod q$

$$r = w \bmod q = w'^{\alpha} \cdot g^{\beta} = (g^{k'})^{\alpha} \cdot g^{\beta} = g^{\alpha \cdot k' + \beta} \bmod q$$

$$s = s' \cdot w \cdot w'^{-1} + \beta \cdot e \bmod q \quad (s' = k' \cdot e' + x \cdot w' \bmod q)$$

$$= (k' \cdot e' + x \cdot w') \cdot w \cdot w'^{-1} + \beta \cdot e \bmod q \quad (e' = \alpha \cdot e \cdot w' \cdot w^{-1} \bmod q)$$

$$= (k' \cdot \alpha \cdot e \cdot w' \cdot w^{-1} + x \cdot w') \cdot w \cdot w'^{-1} + \beta \cdot e \bmod q$$

$$= (\alpha \cdot k' + \beta) \cdot e + x \cdot w \bmod q$$

与原始MDSA签名相比较，这里的$k' \cdot \alpha + \beta$即是原始MDSA签名中的k，因此生成的签名(r, s)可以使用原始MDSA验证算法来验证.

7.2 几个经典盲签名算法

7.2.3 基于MDSA的盲签名算法

➢ 盲性

很显然可以提供盲性.

➢ 不可追踪性

根据$w = w'^{\alpha} \cdot g^{\beta}$， $e' = \alpha \cdot e \cdot w' \cdot w^{-1} \bmod q$，$r = w \bmod q$和$s = s' \cdot w \cdot w'^{-1} + \beta \cdot e$韩亮老师发现基于MDSA的盲签名算法不能提供不可追踪性，具体攻击如下：

① 签名者保存所有盲签名过程中的记录(k', e', w', s')；

② 给定(m, r, s)，签名者计算$e=h(m)$, $\alpha' = e' \cdot e^{-1} \cdot r \cdot w'^{-1}$和$\beta'=(s - s' \cdot w \cdot w'^{-1}) \cdot e^{-1}$；

③ 签名者计算$r' = w'^{\alpha'} \cdot g^{\beta'} \bmod q$并判定$r$和$r'$是否相等；

④ 如果相等则找到了签名记录，否则遍历所有记录(k', e', w', s').

Harn L . Cryptanalysis of the blind signatures based on the discrete logarithm problem, Electronics Letters, 1995, 31(14):1136.

目　录

7.2 几个经典盲签名算法

7.2.4 基于NR签名的盲签名算法

1. NR数字签名算法

K. Nyberg, R.A. Rueppel: A New Signature Scheme Based on the DSA Giving Message Recovery, *The 1st ACM Conference on Computer and Communications Security (CCS'93)*, 1993.

- 全局公开密钥分量，可以为用户公用
  - ✓ p：素数，要求$2^{L-1}<p<2^{L}$,$512<=L<1024$，且L为64的倍数
  - ✓ q: $(p-1)$的素因子，$2^{159}<q<2^{160}$，即比特长度为160位
  - ✓ $g:=h^{(p-1)/q} \bmod p$.其中h是一整数，$1<h<(p-1)$且$h^{(p-1)/q} \bmod p>1$
- 用户私有密钥
  - ✓ x：随机或伪随机整数，要求$0<x<q$
- 用户公开密钥
  - ✓ y：$=g^{x} \bmod p$

7.2 几个经典盲签名算法

7.2.3 基于NR签名的盲签名算法

签名过程：

① 用户随机选取k

② 计算$r=M\cdot g^k \bmod p$

③ 计算$s=k+x\cdot r \bmod q$

④ **输出(r, s)，即为消息M的数字签名.**

验证过程：

① 接收者收到M，r，s后，首先验证$0<r<p$，$0<s<q$；

② $M'=(g^{-s}\cdot y^{r})\cdot r \bmod p$

③ **如果$M'=M$，则确认签名正确，否则拒绝.**

NR签名是基于Schnorr签名算法可恢复消息的签名算法， Camenisch等利用该算法设计了一个盲签名方案.

J. Camenisch, J.-M. Piveteau, and M. Stadler, Blind signatures based on the discrete logarithm problem. *Eurocrypt '94*, pp. 428-432, 1994.

7.2 几个经典盲签名算法

7.2.4 基于NR签名的盲签名算法

2. 基于NR签名的盲签名算法

Signer		*User*
随机选择$k' \in Z_q^*$ 计算$w' = g^{k'} \bmod p$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ 计算$r = M \cdot g^{\alpha} \cdot w'^{\beta} \bmod p$ 计算$M' = r \cdot \beta^{-1} \bmod q$
	← M'	
计算$s' = k' + x \cdot M' \bmod q$	s' →	
		计算$s = \beta \cdot s' + \alpha \bmod q$ 输出(r, s)

7.2 几个经典盲签名算法

7.2.2 基于NR签名的盲签名算法

➢ 完备性

$$r = M \cdot g^{\alpha} \cdot w'^{\beta} = M \cdot g^{\alpha} \cdot (g^{k'})^{\beta}$$

$$= M \cdot g^{\alpha + k' \cdot \beta} \bmod p$$

$$s = \beta \cdot s' + \alpha = \beta \cdot (k' + x \cdot M') + \alpha$$

$$= \beta \cdot (k' + x \cdot r \cdot \beta^{-1}) + \alpha$$

$$= \alpha + k' \cdot \beta + x \cdot r \bmod q$$

- $r = M \cdot g^{k} \bmod p$
- $s = k + x \cdot r \bmod q$

与原始NR签名相比较，这里的$\alpha + k' \cdot \beta$即是原始NR签名中的k，因此生成的签名(e, s)可以使用原始NR验证算法来验证.

7.2 几个经典盲签名算法

7.2.2 基于NR签名的盲签名算法

➢ 盲性

很显然.

➢ 不可追踪性

根据$w' = g^{k'}$，$r = M \cdot g^{\alpha} \cdot w'^{\beta} \bmod p$，$M' = r \cdot \beta^{-1} \bmod q$，$s' = k' + x \cdot M' \bmod q$和$s = \beta \cdot s' + \alpha \bmod q$，我们可以发现基于NR签名的盲签名算法不能提供盲性，具体攻击如下：

① 签名者保存所有盲签名过程中的记录(k', w', M', s');

② 给定(m, r, s)，签名者计算$\beta' = r \cdot M'^{-1} \bmod q$和$\alpha' = s - \beta \cdot s' \bmod q$；

③ 签名者计算$M'' = r \cdot (g^{\alpha'} \cdot w'^{\beta'})^{-1} \bmod p$并判定$M$和$M''$是否相等;

④ 如果相等则找到了签名记录，否则遍历所有记录(k', w', M', s').

目　录

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

解了解决上述方案中中存在在证书管理问题，张方国老师等提出了基于身份的盲签名，该签名算法以下面的基于身份数字签名为基础.

1. 系统参数生成

KGC执行以下步骤生成系统参数和主私钥：

① 选择双线性对$e:G \times G \to G_T$， P 为G 的生成元；

② 生成随机数 $s \in Z_q^*$做为主私钥；

③ 计算系统公钥$P_{pub}=s \cdot P$ 和公开参数$g=e(P, P_{pub})$；

④ 选择两个哈希函数$H:\{0,1\}^* \to Z_q^*$， $H_1:\{0,1\}^* \to G$；

⑤ 保存主私钥 s , 公布系统参数$\{G, q, P, P_{pub}, g, H, H_1\}$.

2. 用户私钥生成

KGC执行以下步骤生成用户私钥：

① 计算$Q_{ID} = H_1(ID)$；

② 计算用户私钥$S_{ID} = s \cdot Q_{ID}$；

Zhang, Fangguo, and Kwangjo Kim. “ID-based blind signature and ring signature from pairings. *ASIACRYPT 2002*. Springer, Berlin, Heidelberg, 2002.

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

➢基于身份的签名算法

给定系统参数，消息m，私钥S_{ID}，签名者执行以下步骤产生签名：

① 生成随机数$r \in Z_q^*$并计算$w = e(r \cdot P, P_{pub})$；

② 计算$h = H(m, w)$ **mod** q；

③ 计算$S = h \cdot S_{ID} + r \cdot P_{pub}$

④ 输出签名(h, S).

➢基于身份的验证算法

给定系统参数，消息m，签名(h, S)，验证者者执行以下步骤验证签名：

① 计算$w' = e(S, P)e(Q_{ID}, P_{pub})^{-h}$

② 检查h和 $H(m, w')$ 是否相等. 如果相等则输出”Accept”，否则输出" Reject“.

容易验证$e(r \cdot P, P_{pub}) = e(S, P)e(Q_{ID}, P_{pub})^{-h}$
正确性成立！

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

3. 盲签名

$Signer$		$User$
随机选择$r \in Z_q^*$ 计算$R = r \cdot P$	R →	
		随机选择$\alpha, \beta \in Z_q^*$ 计算$w = e(\beta \cdot Q_{ID} + R + \alpha \cdot P, P_{pub})$ 计算$h = H(m, w)$ **mod** q 计算$h' = h + \beta$ **mod** q
	← h'	
计算$S' = h' \cdot S_{ID} + r \cdot P_{pub}$	S' →	
		计算$S = S' + \alpha \cdot P_{pub}$ 输出(h, S)

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

4. 验证

给定消息m和签名(h, S)，验证者执行以下步骤验证签名的合法性：

① $Q_{ID} = H_1(ID)$

② 计算$w' = e(S,P)e(Q_{ID}, P_{pub})^{-h}$；

③ 检测h和$H(m, w')$是否相等. 若相等，则输出“Accept”，否则输出“Reject”

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

5. 完备性分析

$w' = e(S,P)e(Q_{ID},P_{pub})^{-h}$

$=e\left(S' + \alpha \cdot P_{pub},P\right)e(Q_{ID},P_{pub})^{-h}$

$=e\left(h' \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)e(Q_{ID},P_{pub})^{-h}$

$=e\left((h+\beta) \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)e(Q_{ID},P_{pub})^{-h}$

$=e\left(\beta \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)e(h \cdot S_{ID},P)e(Q_{ID},P_{pub})^{-h}$

$=e\left(\beta \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)e(h \cdot s \cdot Q_{ID},P)e(Q_{ID},P_{pub})^{-h}$

$=e\left(\beta \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)e(Q_{ID},s \cdot P)^{h}e(Q_{ID},P_{pub})^{-h}$

$= e\left(\beta \cdot S_{ID} + r \cdot P_{pub} + \alpha \cdot P_{pub},P\right)=w$

于是$h = H(m,w)$和$H(m,w')$相等，完备性得证.

7.2 几个经典盲签名算法

7.2.5 基于身份的盲签名算法

6. 盲性

很显然.

7. 不可追踪性

根据$R = r \cdot P$，$w = e(\beta \cdot Q_{ID} + R + \alpha \cdot P, P_{pub})$，$h = H(m, w) \bmod q$，$h' = h + \beta \bmod q$，$S' = h' \cdot S_{ID} + r \cdot P_{pub}$和$S = S' + \alpha \cdot P_{pub}$，我们可以发现这个基于身份的盲签名算法不能提供不可追中性，具体攻击如下：

① 签名者保存所有盲签名过程中的记录(r, R, h', S');

② 给定(m, h, S)，签名者计算$\beta' = h' - h \bmod q$和$T = \alpha \cdot P_{pub} = S - S' \bmod q$；

③ 签名者计算$w' = e(\beta' \cdot Q_{ID} + R, P_{pub})\, e(P, T)$并判定$H(m, w')$和$h$是否相等;

④ 如果相等则找到了签名记录，否则遍历所有记录(r, R, h', S').

目　录

7.3 基于SM2数字签名的盲签名算法

基于Schnorr签名的盲签名算法

Signer		*User*
随机选择$k' \in Z_q^*$ 计算$w' = g^{k'}$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ 计算$w = g^{\alpha} \cdot y^{\beta} \cdot w'$ 计算e=$h(w, m)$ 计算$e' = e + \beta \bmod q$
	← e'	
计算$s' = k' + x \cdot e' \bmod q$	s' →	
		计算$s = s' + \alpha \bmod q$ 输出(e, s)

7.3 基于SM2数字签名的盲签名算法

如何克服上述方案中的缺陷？

以基于**Schnorr**签名的盲签名算法为例：

Signer		*User*
随机选择$k' \in Z_q^*$ 计算$w' = g^{k'}$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ 计算$w = w'^{\alpha^{-1}} \cdot y^{\alpha^{-1} \cdot \beta} \cdot g^{\beta}$ 计算e=$h(w\ ,\ m)$ 计算$e' = \alpha \cdot e + \beta \bmod q$
	← e'	
计算$s' = k' + x \cdot e' \bmod q$	s' →	
		计算$s = \alpha^{-1} \cdot s' + \beta \bmod q$ 输出(e, s)

7.3 基于SM2数字签名的盲签名算法

➢ 完备性

- $w = g^k \bmod q$
- $s = k + x \cdot e \bmod q$

$$w = w'^{\alpha^{-1}} \cdot y^{\alpha^{-1}\cdot\beta} \cdot g^{\beta} = (g^{k'})^{\alpha^{-1}} \cdot (g^{x})^{\alpha^{-1}\cdot\beta} \cdot g^{\beta}$$

$$= g^{\alpha^{-1}\cdot k' + \alpha^{-1}\cdot x\cdot\beta+\beta} \bmod p$$

$$s = \alpha^{-1} \cdot s' + \beta = \alpha^{-1} \cdot (k' + x \cdot e') + \beta$$

$$= \alpha^{-1} \cdot (k' + x \cdot (\alpha \cdot e + \beta)) + \beta$$

$$= \alpha^{-1} \cdot k' + \alpha^{-1} \cdot x \cdot \beta+\beta + x \cdot e \bmod q$$

与原始Schnorr签名相比较，这里的$k' + \alpha + x \cdot \beta$即是原始Schnorr签名中的$k$，因此生成的签名$(e, s)$可以使用原始Schnorr验证算法来验证.

7.3 基于SM2数字签名的盲签名算法

7.3.1 SM2数字签名算法回顾

$$p_{SM2} = 2^{256} - 2^{224} - 2^{96} + 2^{64} - 1$$

1. 曲线参数

SM2标准推荐使用**256位素域F_p上的椭圆曲线$y^2=x^3+ax+b$**，其中：

p=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFF

a=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF FFFFFFFF 00000000 FFFFFFFF FFFFFFFC

b=28E9FA9E 9D9F5E34 4D5A9E4B CF6509A7 F39789F5 15AB8F92 DDBCBD41 4D940E93

n=FFFFFFFE FFFFFFFF FFFFFFFF FFFFFFFF 7203DF6B 21C6052B 53BBF409 39D54123

G_x=32C4AE2C 1F198119 5F990446 6A39C994 8FE30BBF F2660BE1 715A4589 334C74C7

G_y=BC3736A2 F4F6779C 59BDCEE3 6B692153 D0A9877C C62A4740 02DF32E5 2139F0A0

2. 密钥生成算法

① Alice选择随机数d_A做为私钥，其中0< d< n

② Alice计算公钥$P_A = dA \cdot G$

③ 输出密钥对 (sk=d_A, pk= P_A)

7.3 基于SM2数字签名的盲签名算法

7.3.1 SM2数字签名算法回顾

3. 签名算法

- 设Alice发签名消息M给Bob，ID_A是Alice的标识符，$ENTL_A$是ID_A的长度，d_A是A的私钥，基点$G=(x_G, y_G)$，A的公钥$P_A=d_A \cdot G=(x_A, y_A)$.
- $Z_A=H(ENTL_A \| ID_A \| a \| b \| x_G \| y_G \| x_A \| y_A)$， H是SM3算法

① 设置$M^*=Z_A \| M$并计算 $e=H(M^*)$;

② 产生随机数 $k \in [1, n-1]$;

③ 计算椭圆曲线点 $G_1=k \cdot G=(x_1, y_1)$;

④ 计算 $r=(e+x_1) \bmod n$，若$r=0$或$r+k=n$则返回②；

⑤ 计算 $s=(1+d_A)^{-1} \cdot (k-r \cdot d_A) \bmod n = (1+d_A)^{-1} \cdot (k+r) - r \bmod n$，若$s=0$则返回②；

⑥ 以(r, s)作为对消息M的签名.

7.3 基于SM2数字签名的盲签名算法

7.3.1 SM2数字签名算法回顾

4. 验证算法

接收到的消息为M'，签名为(r', s')和发送者Alice的公钥P_A， Bob执行如下步骤验证合法性：

① 检验 $r' \in [1, n-1]$是否成立，若不成立则验证不通过；

② 检验 $s' \in [1, n-1]$是否成立，若不成立则验证不通过；

③ 设置$M^* = Z_A \| M'$；

④ 计算 $e' = H(M^*)$；

⑤ 计算 $t = (r' + s') \bmod n$，若$t = 0$，则验证不通过；

⑥ 计算椭圆曲线点 $(x_1', y_1') = s' \cdot G + t \cdot P_A$；

⑦ 计算 $v = (e' + x_1') \bmod n$，检验$v = r'$是否成立，若成立则验证通过；否则验证不通过.

7.3 基于SM2数字签名的盲签名算法

7.3.2 第一种基于SM2数字签名的盲签名算法

1. Paillier同态加密算法

➢ 密钥生成

① 随机选取两个等长的素数 p 和 q.

② 计算 $g = n + 1, \lambda = \phi(n)$ and $\mu = (\phi(n))^{-1} \, mod \, n$, 其中 $\phi(n) = (p-1)(q-1)$.

③ 公钥$pk = (n, g)$, 私钥$sk = (\lambda, \mu)$.

➢ 加密

① 从Z_n^*中随机选取一个元素r.

② 计算密文 $c = Enc_{pk}(m) = g^m r^n \, mod \, n^2$，其中$0 \leq m < n$.

7.3 基于SM2数字签名的盲签名算法

7.3.2 第一种基于SM2数字签名的盲签名算法

➢ 解密

解密密文c，$m = Dec_{sk}(c) = L\left(c^{\lambda} \bmod n^2\right) \cdot \mu \bmod n$，这里$L(x) = \frac{x-1}{n}$.

➢ 性质

在Paillier加密中，我们假设$c_1 = Enc_{pk}(m_1)$，$c_2 = Enc_{pk}(m_2)$，则我们可以得到：

1. $Dec_{sk}\left(Enc_{pk}(m_1) \cdot Enc_{pk}(m_2)\right) = m_1 + m_2$
2. $Dec_{sk}\left(Enc_{pk}(m_1)^{m_2}\right) = m_1 m_2$

7.3 基于SM2数字签名的盲签名算法

7.3.2 第一种基于SM2数字签名的盲签名算法

2. 盲签名算法描述

$Signer$		$User$
选择随机数$k_1 \in Z_q^*$ 计算$K_1 = k_1 G$	K_1 →	
		选择随机数$k_2 \in Z_q^*$ 计算$K = k_2 K_1 = (x_K, y_K)$ 计算$r = e + x_K \bmod q$ 选择随机数$\{r_1, r_2\} \in Z_{N^2}^*$ 计算$C_1 = g^r r_1^N \bmod N^2$ 计算$C_2 = g^{k_2} r_2^N \bmod N^2$
	← C_1, C_2, g, N	
选择随机数$r_3 \in Z_{N^2}^*$，$\rho \in Z_q^*$ 计算$C = (C_1 C_2^{k_1})^{(1+x)^{-1}} r_3^N \bmod N^2$	C →	
		解密$D(C) = s' = (1+x)^{-1}(k_1 k_2 + r) \bmod q$ $s = s' - r = (1+x)^{-1}(k_1 k_2 + r) - r \bmod q$

何德彪，张韵茹，谢翔，李升林，孙立林，盲签名的获取方法、装置和服务器，申请公布号：CN109818730A

7.3 基于SM2数字签名的盲签名算法

7.3.2 第一种基于SM2数字签名的盲签名算法

3.完备性

$$D(C)=s'=\frac{L(C^{\lambda}\ mod\ N^2)}{L(g^{\lambda}\ mod\ N^2)}\ mod\ q=\frac{L(g^{\lambda(1+x)^{-1}(k_1k_2+r)}r_3^{\lambda N}mod\ N^2)}{L(g^{\lambda}\ mod\ N^2)}\ mod\ q$$

$$=\frac{L((1+N)^{pt\lambda(1+x)^{-1}(k_1k_2+r)}\ mod\ N^2)}{L(\left((1+N)^{pt\lambda}\right)\ mod\ N^2)}\ mod\ q=\frac{L((1+pt\lambda(1+x)^{-1}(k_1k_2+r)N)\ mod\ N^2)}{L((pt\lambda N)\ mod\ N^2)}mod\ q$$

$$=\frac{pt\lambda(1+x)^{-1}(k_1k_2+r)N}{pt\lambda N}mod\ q$$

$$=(1+x)^{-1}(k_1k_2+r)\ mod\ q$$

$$s=s'-r\ mod\ q=(1+x)^{-1}(k_1k_2+r)-r\ \ mod\ q$$

何德彪，张韵茹，谢翔，李升林，孙立林，盲签名的获取方法、装置和服务器，申请公布号：CN109818730A

7.3 基于SM2数字签名的盲签名算法

7.3.2 第一种基于SM2数字签名的盲签名算法

4. 盲性

很显然

5. 不可追踪

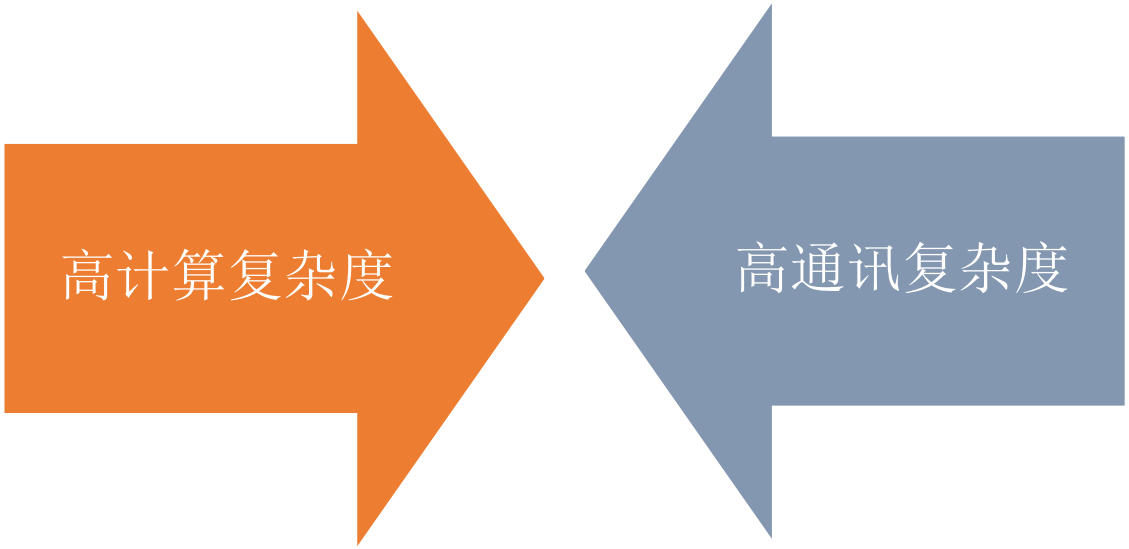
不能提供不可追踪

何德彪，张韵茹，谢翔，李升林，孙立林，盲签名的获取方法、装置和服务器，申请公布号：CN109818730A

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

1. 问题分析

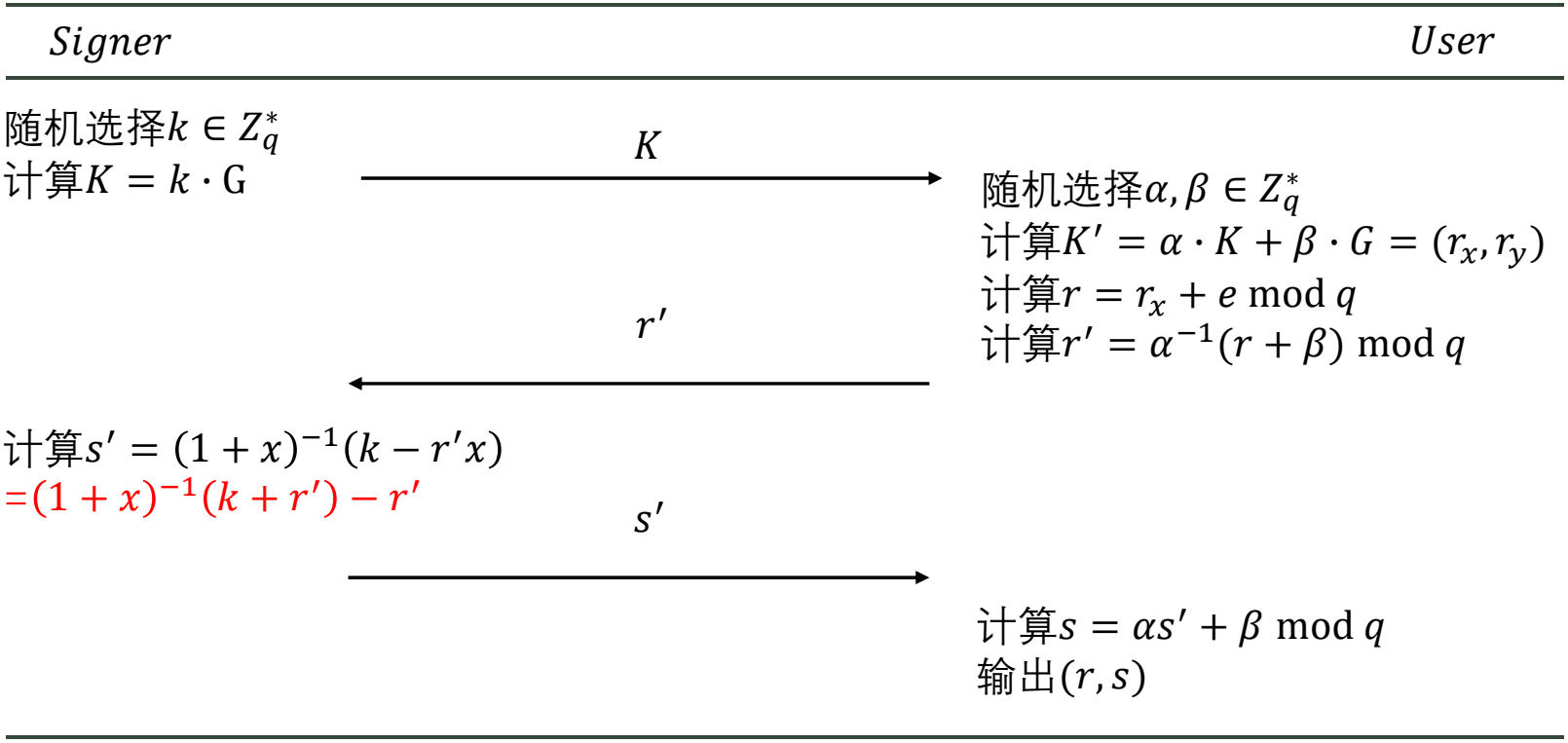


是否存在轻量级盲签名方案？

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

2. 盲签名算法描述



何德彪，张语荻，张方国，冯琦，王婧，一种轻量级SM2盲签名生成方法及系统，申请公布号：CN110213048A

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

3. 正确性分析

$$K' = (\alpha k + \beta)G = (r_x, r_y)$$

$$s = \alpha s' + \beta \bmod q$$

$$= \alpha\left((1+x)^{-1}(k+r') - r'\right) + \beta \bmod q$$

$$= \alpha\left((1+x)^{-1}\left(k + \alpha^{-1}(r+\beta)\right) - \alpha^{-1}(r+\beta)\right) + \beta \bmod q$$

$$= (1+x)^{-1}(\alpha k + \alpha\alpha^{-1}(r+\beta)) - \alpha\alpha^{-1}(r+\beta) + \beta \bmod q$$

$$= (1+x)^{-1}(\alpha k + \beta + r) - r \bmod q$$

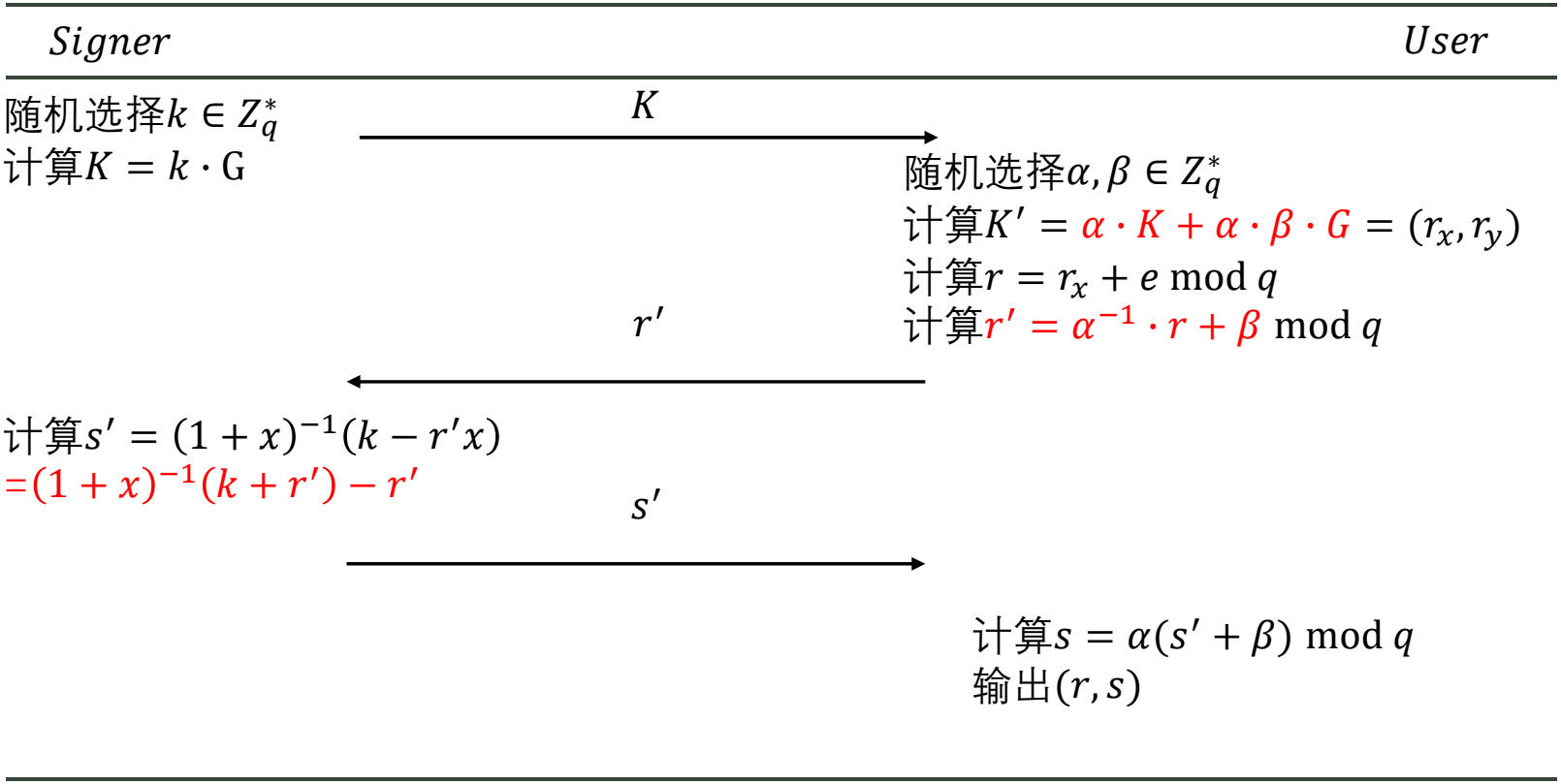
与原始SM2签名相比较，这里的 $\alpha k + \beta$ 即是原始SM2签名中的 k，因此生成的签名 (r, s) 可以使用原始SM2验证算法来验证. **根据这个算法还可以变形盲化过程，比如下面的算法：**

何德彪，张语荻，张方国，冯琦，王婧，一种轻量级SM2盲签名生成方法及系统，申请公布号：CN110213048A

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

4. 变形盲签名算法描述



何德彪，张语荻，张方国，冯琦，王婧，一种轻量级SM2盲签名生成方法及系统，申请公布号：CN110213048A

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

5. 变形盲签名算法的正确性分析

$$s = \alpha(s' + \beta) \bmod q$$

$$= \alpha\left((1+x)^{-1}(k + r') - r'\right) + \alpha\beta \bmod q$$

$$= \alpha\left((1+x)^{-1}(k + \alpha^{-1} \cdot r + \beta) - \alpha^{-1} \cdot r - \beta\right) + \alpha\beta \bmod q$$

$$= (1+x)^{-1}(\alpha k + \alpha\alpha^{-1} r + \alpha\beta) - \alpha\alpha^{-1} \cdot r - \alpha\beta + \alpha\beta \bmod q$$

$$= (1+x)^{-1}(\alpha k + \alpha\beta + r) - r \bmod q$$

$$K' = (\alpha k + \alpha\beta)G$$
$$= \alpha \cdot K + \alpha \cdot \beta \cdot G$$
$$= (r_x, r_y)$$

与原始SM2签名相比较，这里的 $\alpha k + \beta$ 即是原始SM2签名中的k，因此生成的签名(r, s)可以使用原始SM2验证算法来验证.

何德彪，张语荻，张方国，冯琦，王婧，一种轻量级SM2盲签名生成方法及系统，申请公布号：CN110213048A

7.3 基于SM2数字签名的盲签名算法

7.3.3 第二种基于SM2数字签名的盲签名算法

6. 盲性

很显然

7. 不可追踪性

能提供不可追踪性

何德彪，张语荻，张方国，冯琦，王婧，一种轻量级SM2盲签名生成方法及系统，申请公布号：CN110213048A

7.3 基于SM2数字签名的盲签名算法

7.3.4 两种盲签名算法的性能分析

➢ 计算复杂度

	KeyGen	Sign	Verify
同态加密盲签名	2094.147	7323.399	119.426
轻量级盲签名	59.713	179.815	119.426

Sign	Step1	Step2	Step3	Step4
同态加密盲签名	59.713	3150.937	2069.367	2043.382
轻量级盲签名	59.713	119.868	0.2	0.034

➢ 通信复杂度

	同态加密盲签名	轻量级盲签名
Sign过程	8704	1024

测试配置： RASPBERRY Pi 2 MODEL B Linux 4.4.0-1082-raspi2

目　录

7.4 基于SM9数字签名的盲签名算法

7.4.1 SM9数字签名回顾

1. 系统参数生成

密钥生成中心(Key Generation Center, KGC)执行以下步骤生成系统参数和主私钥：

① KGC生成随机数sk做为主私钥，这里$0< sk <q-1$；

② KGC计算系统公钥$P_{pub}=sk \cdot P_2$；

③ KGC保存私钥sk, 公布系统公钥.

- 注意：

① SM9算使用BN曲线，G_1和 G_2分别是椭圆曲线$E(F_p)$和$E(F_{p^2})$的加法群， G_T是乘法群 $F_{p^{12}}$ ，群G_2中元素尺寸是群G_1中元素尺寸的2倍.

② 选择系统公钥为G_2中的元素，那么就可以使得用户私钥和签名中一部分是G_1中元素，降低了用户私钥和签名的尺寸.

7.4 基于SM9数字签名的盲签名算法

7.4.1 SM9数字签名回顾

2. 用户私钥生成

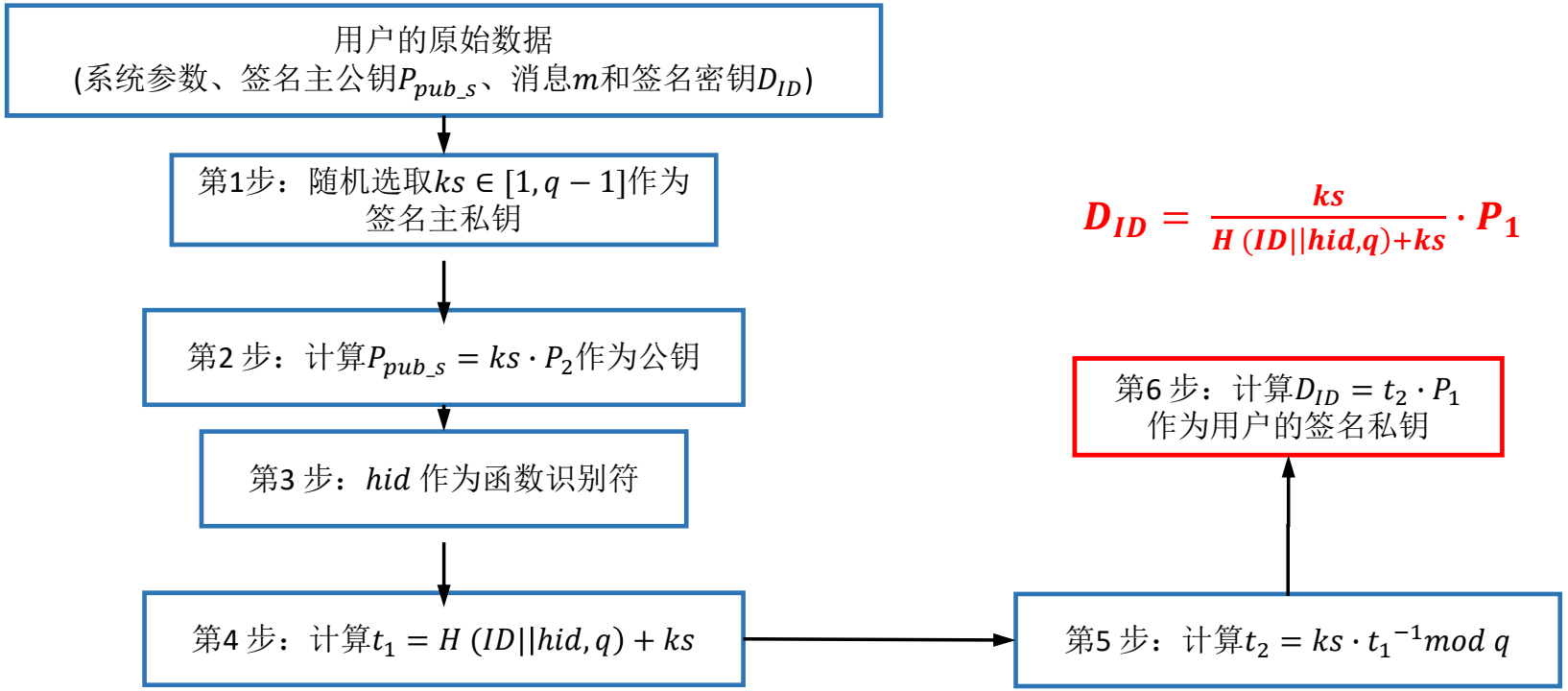


图6.23. SM9数字签名算法用户私钥生成示意图

7.4 基于SM9数字签名的盲签名算法

7.4.1 SM9数字签名回顾

3. 签名

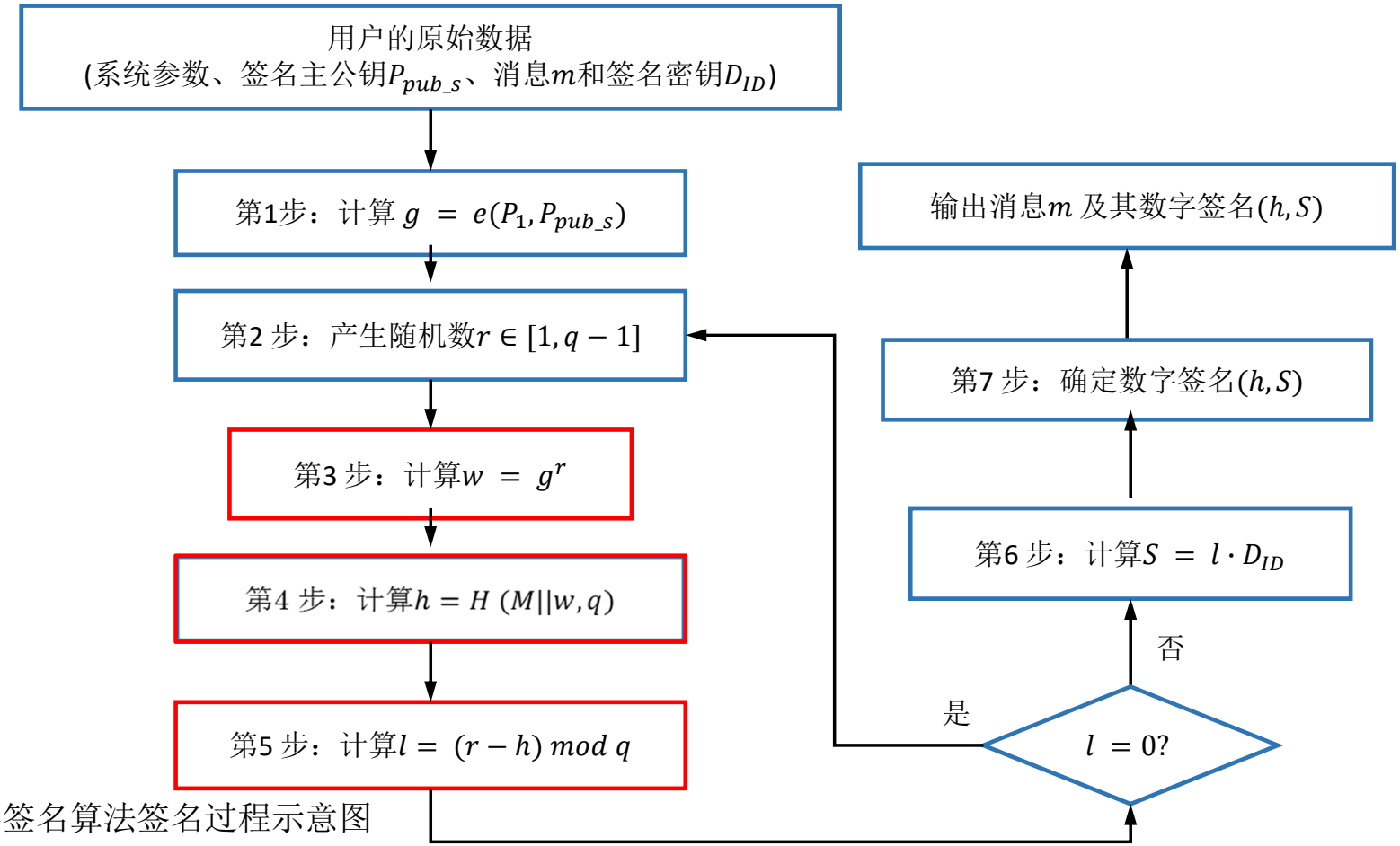


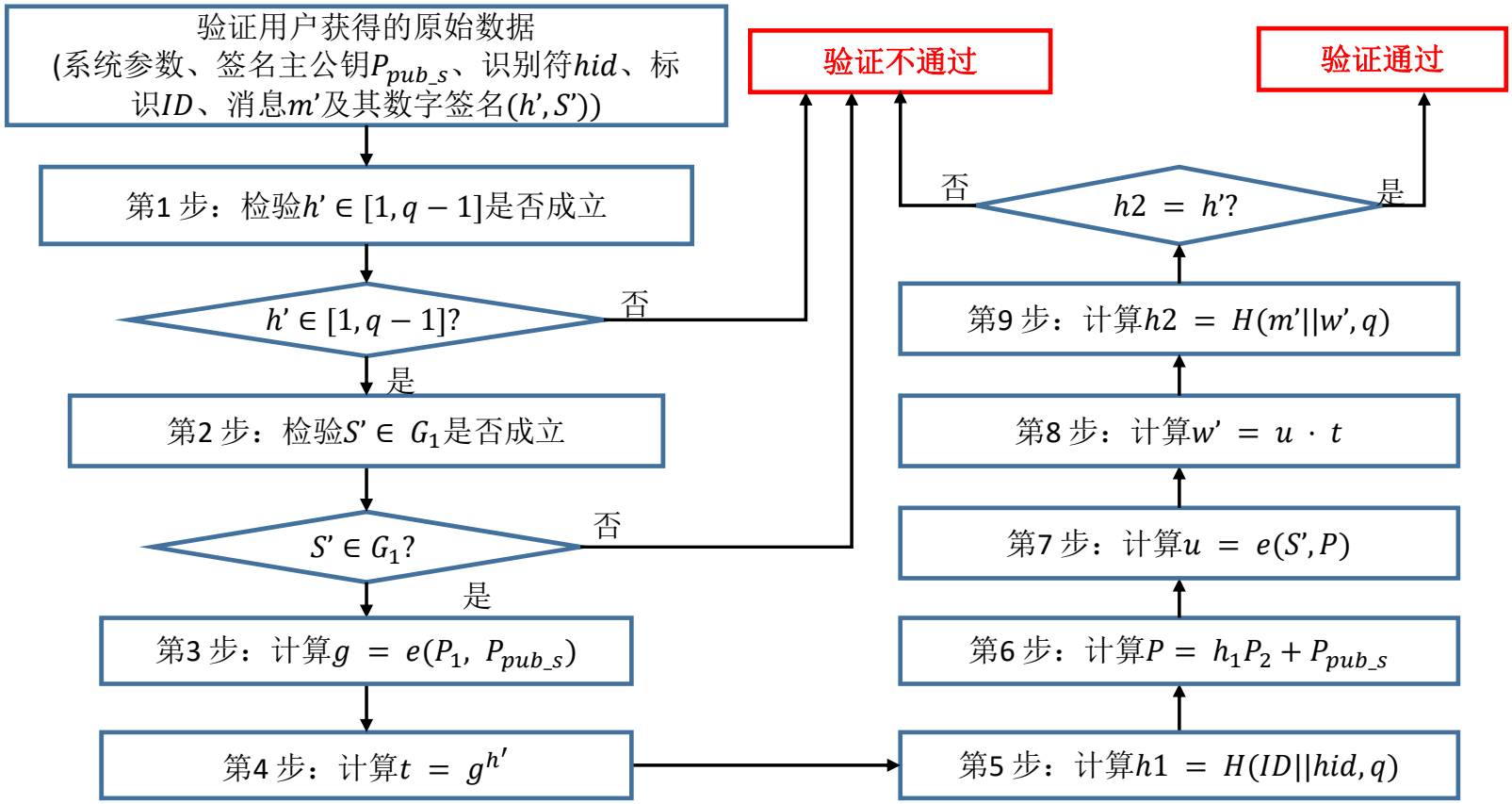
图6.24. SM9数字签名算法签名过程示意图

7.4 基于SM9数字签名的盲签名算法

7.4.1 SM9数字签名回顾

4. 验证

图6.25. SM9数字签名算法验证过程示意图



7.4 基于SM9数字签名的盲签名算法

7.4.2 第一种基于SM9数字签名的盲签名算法

1. 算法描述

$Signer$		$User$
随机选择$r \in Z_q^*$ 计算$R = r \cdot P_1$	R →	随机选择$\alpha, \beta \in Z_q^*$ 计算$w'' = e(\beta \cdot P_1, h_1 P_2 + P_{pub_s})$ 计算$w = e(\alpha \cdot R + \alpha \cdot \beta \cdot P_1, P_{pub}) \cdot w''$ 计算$h = H_2(M \| w, N)$ 计算$h' = \alpha^{-1} \cdot h - \beta \bmod q$
	← h'	
计算$S' = (r - h') \cdot D_{ID}$	S' →	
		计算$S = \alpha \cdot S' + \beta \cdot P_1$ 输出(h, S)

注意：也可以随机生成一个G_1中元素A来替代$\beta \cdot P_1$

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.2 第一种基于SM9数字签名的盲签名算法

2. 正确性分析

$$w = e(\alpha \cdot R + \alpha \cdot \beta \cdot P_1, P_{pub}) \cdot w''$$

$$= e(\alpha \cdot r \cdot P_1 + \alpha \cdot \beta \cdot P_1, P_{pub}) \cdot w''$$

$$= e((\alpha \cdot r + \alpha \cdot \beta) \cdot P_1, P_{pub}) \cdot w'' = g^{\alpha \cdot r + \alpha \cdot \beta} \cdot w''$$

$$e(S, h_1 \cdot P_2 + P_{pub\_s})\, g^h$$

$$= e(\alpha \cdot S' + \beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s})\, g^h$$

$$= e(\alpha \cdot S', h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$$

$$= e((\alpha \cdot r + \alpha \cdot \beta - h) \cdot D_{ID}, h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$$

$$= g^{\alpha \cdot r + \alpha \cdot \beta} \cdot g^{-h} \cdot w'' \cdot g^h = g^{\alpha \cdot r + \alpha \cdot \beta} \cdot w'' = w$$

因此生成的签名 (h, S) 可以通过SM9验证算法来验证. **根据这个算法还可以变形盲化过程，比如下面的算法：**

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.2 第一种基于SM9数字签名的盲签名算法

3. 变形算法描述

$Signer$		$User$
随机选择$r \in Z_q^*$ 计算$R = r \cdot P_1$	R →	随机选择$\alpha, \beta \in Z_q^*$ 计算$w'' = e(\beta \cdot P_1, h_1 P_2 + P_{pub_s})$ 计算$w = e(\alpha \cdot R + \beta \cdot P_1, P_{pub}) \cdot w''$ 计算$h = H_2(M \| w, N)$ 计算$h' = \alpha^{-1} \cdot (h - \beta) \bmod q$
	← h'	
计算$S' = (r - h') \cdot D_{ID}$	S' →	
		计算$S = \alpha \cdot S' + \beta \cdot P_1$ 输出(h, S)

注意：也可以随机生成一个G_1中元素A来替代$\beta \cdot P_1$

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.2 第一种基于SM9数字签名的盲签名算法

4. 完备性分析

$$w = e(\alpha \cdot R + \beta \cdot P_1, P_{pub}) \cdot w''$$

$$= e(\alpha \cdot r \cdot P_1 + \beta \cdot P_1, P_{pub}) \cdot w''$$

$$= e((\alpha \cdot r + \beta) \cdot P_1, P_{pub}) \cdot w''$$

$$= g^{\alpha \cdot r + \beta} \cdot w''$$

$$e(S, h_1 \cdot P_2 + P_{pub\_s})\, g^h$$

$$= = e(\alpha \cdot S' + \beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s})\, g^h$$

$$= e(\alpha \cdot S', h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$$

$$= e((\alpha \cdot r + \beta - h) \cdot D_{ID}, h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$$

$$= g^{\alpha \cdot r + \beta} \cdot g^{-h} \cdot w'' \cdot g^h = g^{\alpha \cdot r + \beta} \cdot w'' = w$$

因此生成的签名(h, S)可以通过SM9验证算法来验证.

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.2 第一种基于SM9数字签名的盲签名算法

5. 盲性

很显然

6. 不可追踪性

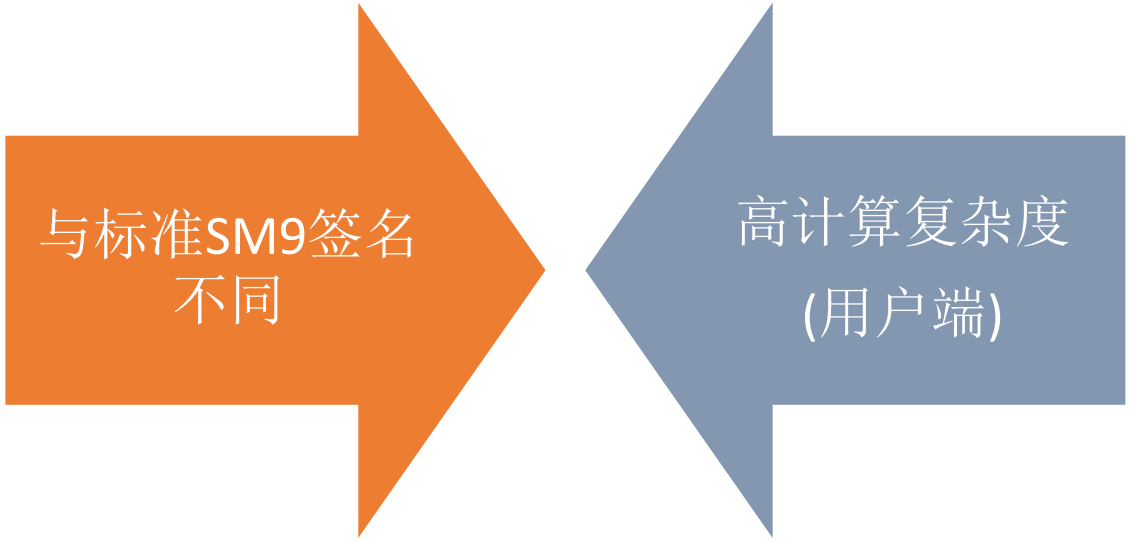
能提供不可追踪性

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

1. 问题分析



是否存在高效盲签名方案？

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

2. 算法描述

$Signer$		$User$
随机选择$r \in Z_q^*$ 计算$w' = g^r$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ $w'' = e(\beta \cdot P_1, h_1 P_2 + P_{pub\_s})$ 计算$w = w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot w''$ 计算$h = H_2(M \| w, N)$ 计算$h' = \alpha^{-1} \cdot h - \beta \bmod q$
	← h'	
计算$S' = (r - h') \cdot D_{ID}$	S' →	
		计算$S = \alpha \cdot S' + \beta \cdot P_1$ 输出(h, S)

注意：也可以随机生成一个G_1中元素A来替代$\beta \cdot P_1$

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

3. 正确性分析

$w'' = e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s})$

$w = w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot w''$

$e(S, h_1 \cdot P_2 + P_{pub\_s})\, g^h$

$= e(\alpha \cdot S' + \beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s})\, g^h$

$= e(\alpha \cdot S', h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$

$= e((\alpha \cdot r + \alpha \cdot \beta - h) \cdot D_{ID}, h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^h$

$= w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot g^{-h} \cdot w'' \cdot g^h$

$= w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot w'' = w$

因此生成的签名(h, S)可以通过SM9验证算法来验证.根据这个算法还可以变形盲化过程，比如下面的算法：

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

4. 变形盲签名算法算法描述

$Signer$		$User$
随机选择$r \in Z_q^*$ 计算$w' = g^r$	w' →	
		随机选择$\alpha, \beta \in Z_q^*$ $w'' = e(\beta \cdot P_1, h_1 P_2 + P_{pub\_s})$ 计算$w = w'^{\alpha} \cdot g^{\beta} \cdot w''$ 计算$h = H_2(M\|\|w, N)$ 计算$h' = \alpha^{-1}(h - \beta) \bmod q$
	← h'	
计算$S' = (r - h') \cdot D_{ID}$	S' →	
		计算$S = \alpha \cdot S' + \beta \cdot P_1$ 输出(h, S)

注意：也可以随机生成一个G_1中元素A来替代$\beta \cdot P_1$

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

5. 变形盲签名算法正确性分析

$$w = w'^{\alpha} \cdot g^{\beta} \cdot w'' = (g^{r})^{\alpha} \cdot g^{\beta} \cdot w''$$

$$= g^{\alpha \cdot r + \beta} \cdot w''$$

$$e(S, h_1 \cdot P_2 + P_{pub\_s})\, g^{h}$$

$$= e(\alpha \cdot S' + \beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s})\, g^{h}$$

$$= e(\alpha \cdot S', h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^{h}$$

$$= e((\alpha \cdot r + \alpha \cdot \beta - h) \cdot D_{ID}, h_1 \cdot P_2 + P_{pub\_s}) \cdot e(\beta \cdot P_1, h_1 \cdot P_2 + P_{pub\_s}) \cdot g^{h}$$

$$= w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot g^{-h} \cdot w'' \cdot g^{h}$$

$$= w'^{\alpha} \cdot g^{\alpha \cdot \beta} \cdot w'' = w$$

因此生成的签名(h, S)可以通过SM9验证算法来验证.

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.3 第二种基于SM9数字签名的盲签名算法

6. 盲性

很显然

7. 不可追踪性

能提供不可追踪性

何德彪，张语荻，张宇波，陈泌文，一种基于SM9数字签名的盲签名生成方法及系统，申请公布号：CN108551392A

7.4 基于SM9数字签名的盲签名算法

7.4.4 两种盲签名算法的性能分析

➢ 计算复杂度(ms)

签名	Step1	Step2	Step3	Step4	总计
传统盲签名	176.057	1723.868	176.058	176.057	2252.039
改进盲签名	579.632	1159.501	176.058	176.057	2091.248

➢ 通信复杂度(bit)

	第一种盲签名	第二种盲签名
签名过程	256x2+256+256x2= 1280	256x12+256+256x2= 3584

➢ 测试环境：

- ✓ Intel I5-4210 CPU @1.7GHz
- ✓ 4G RAM
- ✓ Win 7
- ✓ VS2017
- ✓ MIRACL 5.5.4

目　录

7.5 盲签名算法在区块链中的应用

在比特币中，交易是在用户之间直接进行的，没有中间人. 用户通过向通信节点的对等网络广播数字签名的消息来发送支付.

区块链技术的快速发展，越来越多的在线商家提供比特币支付功能. 比特币的匿名性：交易被记录并公开，但它们只与地址而不是真实身份联系在一起.

区块链中的“地址”类似于银行卡账号，是用户参与区块链业务时使用的假名. 因此，无论你用你的比特币购买什么，都无法具体追踪到你的真实地址.

一般来说，匿名化的目的是防止攻击者通过比特币网络和区块链发现比特币钱包地址与真实用户身份信息之间的关系. 但是，比特币交易的匿名性特征绝非完美：如果客户在线向提供商支付比特币，例如通过银行转账、信用卡，甚至阿里支付，提供商可以将客户的支付信息链接到交易比特币的电子地址.

7.5 盲签名算法在区块链中的应用

身份隐私：用户身份信息与区块链地址之间的关联关系

区块链系统中地址是由用户自行生成，与用户的身份信息无关，用户创建和使用地址不需要第三方参与. 因此，相对于传统的账号(例如银行卡号)，区块链地址具有较好的匿名性.

但是区块链交易之间的关联性可以被用于推测敏感信息. 区块链所有数据都存储在公开的全局账本中，用户在使用区块链地址参与区块链业务时，有可能泄露一些敏感信息，例如区块链交易在网络层传播轨迹.

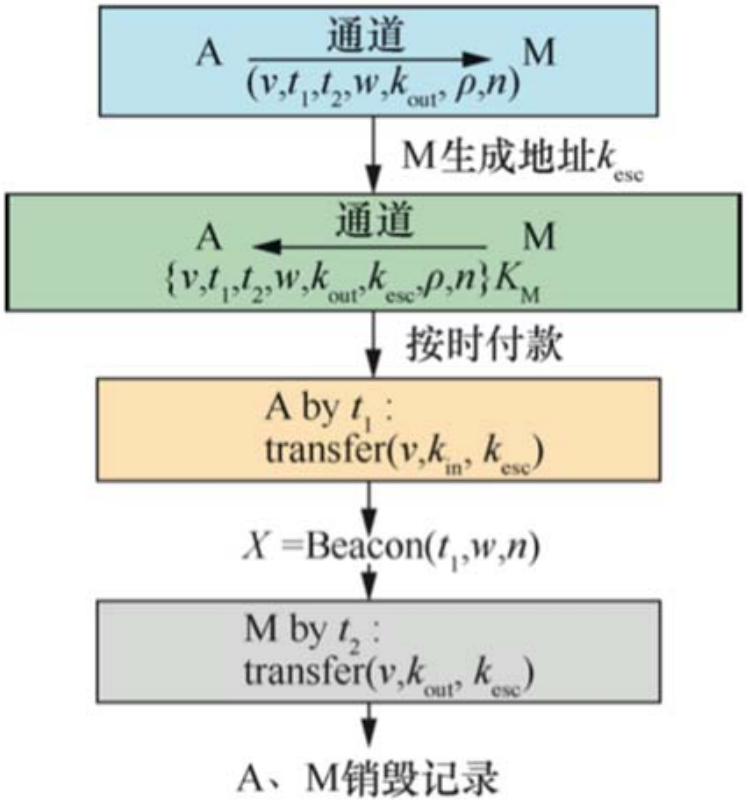
通过分析这些交易之间的关联关系(比如：同一交易的所有输入地址属于同一用户集合、找零地址和输入地址属于同一用户等等)，再结合一些背景知识，能够逐步降低区块链地址的匿名性，甚至发现匿名地址对应用户的真实身份.

7.5 盲签名算法在区块链中的应用

- 区块链网络中，一笔交易的输入肯定是某一笔交易的输出，找到交易间的关联关系.
- 为了断开k_{in}和k_{out}之间的连接，保护区块链隐私：混币技术
- **Mixcoin协议方案**

$$A\begin{pmatrix} k_{in} \xrightarrow{v} k_{esc} \\ k_{out} \xleftarrow[v]{} k_{esc} \end{pmatrix}M$$

✓ 混币服务器掌握A的k_{in}和k_{out}之间的连接，存在泄漏用户隐私的问题



Bonneau J, Narayanan A, Miller A, et al. Mixcoin: Anonymity for Bitcoin with accountable mixes[C]//International Conference on Financial Cryptography and Data Security. Springer, Berlin, Heidelberg, 2014: 486-504.

7.5 盲签名算法在区块链中的应用

混币技术——盲签名：

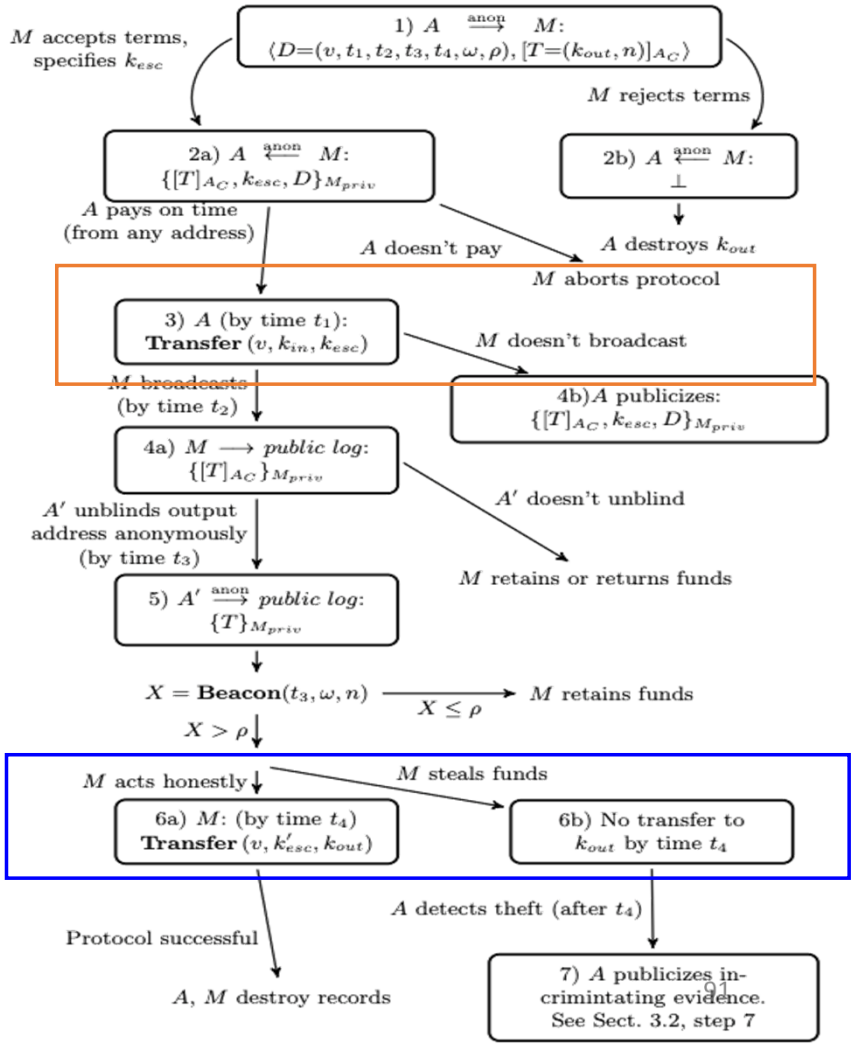
➢ **Blindcoin协议方案**

$$A\left(\begin{array}{c} k_{in} \overset{v}{\rightarrow} k_{esc} \\ k_{out} \overset{v}{\leftarrow} k'_{esc} \end{array}\right) M$$

✓ M公布盲签名$\{[T]_{A_c}\}_{sign}$，作为追责凭证

✓ A公布去盲签名$[T]_{sign}$，作为M向指定输出地址发送交易的凭证

✓ 用户对输出地址采用盲签名， M不知道A的k_{in}和k_{out}之间的连接， 不会将用户的输入地址与输出地址相连接，从而保护用户交易信息



7.5 盲签名算法在区块链中的应用

混币技术——盲签名：

- **Blindcoin协议方案**

Blindcoin保证了没有被动对手可以在特定的混合中链接输入/输出地址对. 因此，Blindcoin在同时参与混合的所有非恶意用户集合内实现k-匿名性. 除了混币服务器的资源外，对可以同时参与混币的用户数没有任何限制，因此参与者越多，匿名集就越大.

通过使用盲签名方案隐藏参与者的输入/输出地址映射，实现了对混合服务的匿名性.

Valenta, Lukeand Rowan, Brendan. Blindcoin: Blinded, Accountable Mixes for Bitcoin. Financial Cryptography and Data Security, Springer Berlin Heidelberg, 2015. Berlin: Heidelberg, 112—126.

谢谢！