软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.007115]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

基于 TEE 和 SE 的移动平台双离线匿名支付方案 *

杨波[1,2], 冯伟[3], 秦宇[3], 张彦超[1,2], 佟冬[1,2]

[1](国家金融科技测评中心, 北京 100041)

[2](银行卡检测中心 研发中心, 北京 100041)

[3](中国科学院 软件研究所 可信计算与信息保障实验室, 北京 100190)

通讯作者: 秦宇, E-mail: qinyu@iscas.ac.cn

摘 要: 近年来, 中央银行数字货币(CBDC)受到全球多个国家和地区的高度关注. 双离线交易作为 CBDC 的可选属性, 在无网络连接的情况下进行支付被认为具有较大的实用价值. 本文面向 CBDC 的双离线匿名支付场景, 基于可信执行环境(TEE)和安全单元(SE)技术, 提出了一种专为移动平台设计的高效双离线匿名支付方案(Dual Offline Anonymous E-Payment for Mobile Devices, OAPM). OAPM 适用于资源受限的移动设备, 允许移动付款者在不联网状态下安全地向收款者支付数字货币, 且不向收款者及商业银行泄露个人隐私信息, 付款者的支付行为也不会被链接, 同时允许收款者设备处于离线状态, 监管机构(如中央银行)在必要情况下能够识别匿名付款者的真实身份. 本方案满足数字货币交易的多项重要属性, 包括正确性、不可链接性、可追踪性、不可陷害性、机密性、真实性、防双花性以及可控匿名性等. 本文实现了原型系统, 并对可能的参数进行了评估. 安全性分析和实验结果表明, 该方案从安全性和效率两方面均能满足移动用户 CBDC 双离线交易的实际需求.

关键词: 央行数字货币 (CBDC); 双离线支付; 可信执行环境 (TEE); 安全单元 (SE); 移动设备; 安全架构

中图法分类号: TPXXX

中文引用格式: 杨波,冯伟,秦宇,张彦超,佟冬. 基于 TEE 和 SE 的移动平台双离线匿名支付方案.软件学报, http://www.jos.org.cn/1000-9825/7115.htm

英文引用格式: Yang B, Feng W, Qin Y, Zhang YC, Tong D. A Dual Offline Anonymous E-Payment Scheme for Mobile Devices Based on TEE and SE. Ruan Jian Xue Bao/Journal of Software. http://www.jos.org.cn/1000-9825/7115.htm

A Dual Offline Anonymous E-Payment Scheme for Mobile Devices Based on TEE and SE

YANG Bo[1,2], FENG Wei[3], QIN Yu[3], ZHANG Yan-Chao[1,2], TONG Dong[1,2]

[1](National Fintech Evaluation Center, Beijing 100041, China)

[2](Research and Development Center, Bank Card Test Center, Beijing 100041 China)

[3]((Trusted Computing and Information Assurance Laboratory, Institute of Software, CAS, Beijing 100190, China)

Abstract: In recent years, many major economies have paid close attention to central bank digital currency (CBDC). As an optional attribute of CBDC, dual offline transaction is considered to have great practical value under the circumstances for payment without network connection. In this paper, for CBDC we propose OAPM, a dual offline anonymous e-payment scheme for mobile devices user as either a payer or a payee based on trusted execution environment (TEE) and secure element (SE). OAPM is suitable for mobile devices with limited resources. It allows payer to safely pay digital currency to payees without networking, without disclosing personal privacy information to payees and commercial banks, and without linking the payment behaviors of payers. Meanwhile, it allows payees' devices to be offline. Regulators, such as central banks, can identify anonymous payers if necessary. The scheme satisfies a number of important attributes of digital currency transactions, including correctness, unlinkability, traceability, non-frame-up, confidentiality, authenticity, anti-double-cross and controllable anonymity, etc. Finally, the prototype system is implemented and the possible parameters are tested. Security analysis and experimental results show that the scheme can meet the actual needs of CBDC offline transaction of mobile users

* 基金项目: 国家重点研发计划资助(2022YFB4501500, 2022YFB4501501).
收稿时间: 2023-09-05; 修改时间: 2023-10-30; 采用时间: 2023-12-15; jos 在线出版时间: 2024-01-05

from both security and efficiency.

Key words: central bank digital currency (CBDC); dual offline payment; trusted execution environment (TEE); secure element (SE); mobile device; secure architecture.

1 引言

当前, 中央银行数字货币(Central Bank Digital Currency, CBDC)已成为全球多个国家和地区的研究热点, 工业界和学术界均看好其在未来的应用价值, 包括中国在内的若干国家或组织已经研发出相应的原型系统进行试点验证. 根据国际清算银行(BIS)发布的 2022 年 CBDC 调查结果, 在 86 个国家或经济体中, 有高达 93%的中央银行正在探索央行数字货币[1]. 在央行数字货币的发展过程中, 移动终端双离线支付需求多次被提及, 其技术目标是在交易双方移动设备均未联网的情况下实现安全有效的数字货币交易, 满足一定情况下的紧急应用场景要求, 如偏远无网地区或灾害发生地区. 由于 CBDC 顶层设计依赖于具有中心权威机构的非分布式架构[2], 实现CBDC框架下的双离线交易是一个公认的难题. 另一方面, 信息泄露的问题正严重威胁着移动支付用户的隐私安全. 当前已部署上线的多数移动支付方案都潜在地将付款者的个人信息源源不断地泄露给银行、第三方支付机构或收款者, 这些个人信息可能包括付款者的真实身份、账单详情或购物记录等[3,4,5,6], 导致付款者的人身与财产安全遭到威胁[7]. 因此, 用户匿名支付与隐私保护需求日益增强. 目前, 尚未有公开方案实现专门针对 CBDC 且满足匿名支付属性的双离线交易技术.

匿名支付方法是解决付款者隐私泄露的一种重要技术手段. 匿名技术如 CL 签名[8]、DAA[9]和 U-Prove[10]等可作为候选协议用于匿名支付方案的设计, 但这些协议很难同时满足匿名支付在匿名性和灵活性两方面的需求. 由 Chaum 提出的 e-cash(Electronic Cash) [11]系统能够允许付款者以匿名的方式从银行提取数字货币并支付给收款者, 整个过程很好地模拟了传统现金交易. 以 e-cash 为基础改进而来的可分(Divisible)e-cash 系统解决了分割大面值数字货币的问题, 然而大多数可分 e-cash 方案的实现效率较差, 不适合部署于资源受限的移动平台. Canard 等人给出了第一个高效的匿名可分 e-cash 协议[12], 该协议中用户可以一次性提取面值为2^n(n为非负整数)的数字货币, 并能够将其分割成不同面额的多个子块分若干次消费, 这种可分 e-cash 方案使数字货币交易更加高效和灵活. 此外, 协议里可信权威(Trusted Authority)是被绝对信任的第三方机构, 在出现双重花费(Double-spending)的现象时, 可以揭开双重花费者(Double-spender)的真实身份. 所谓双花, 指的是同一份数字货币被花费了两次(甚至更多次), 在协议执行安全的假设下, 造成这种情况的原因大多来自于某付款者的恶意欺诈行为, 因此需要可信权威揭露该付款者原本匿名的身份, 用于采取进一步惩罚措施. 这一措施符合央行数字货币的可控匿名需求[13,14,15] , 即付款者信息对收款者或一般机构匿名, 中央银行或可信权威机构在特殊情况下可以识别某个匿名交易的付款者身份, 这样有利于打击匿名带来的非法交易等问题, 符合金融监控的相关要求.

与此同时, 在移动终端设备上构建可信执行环境(Trusted Execution Environment, TEE)成为近年来移动设备应对移动系统及应用安全漏洞的常规手段. 隔离于通用执行环境(Rich Execution Environment, REE), TEE 旨在保护安全敏感的代码执行和相关数据信息免受恶意敌手的攻击和破坏[16,17,18]. 作为 ARM 架构的一种硬件安全扩展技术, TrustZone[19,20,21,22]能够为移动操作系统和上层移动应用提供强大的底层安全支持, 可以用于移动平台的 TEE 构建. TEE 支持自主开发, 设计特定的安全系统. 该技术提供两个执行环境: 安全世界 SW(Secure World)和普通世界 NW(Normal World). SW 通常用于实现 TEE, 执行有定制的安全操作系统以及可信应用, 而 NW 用于实现 REE, 可以运行通用移动操作系统和普通上层应用程序. TrustZone 的应用能够满足传统的在线移动支付场景安全需求, 其安全敏感的操作与计算较高程度地依赖于银行后台交易系统, 风险相对可控. 然而, 在金融级移动离线支付场景下, 移动终端需离线完成所有核心支付流程, 抵御更为苛刻的多种类型离线攻击, 使得支付方案的安全属性更多建立在移动终端本身的安全基础之上. 因此, 可信移动终端的构建应依赖于可靠的信任根和密钥管理机制, 信任根应能抵抗包括底层物理攻击在内的高强度攻击方法, 而 TrustZone 技术本身并不足以满足要求[23,24,25], 安全单元(Secure Element, SE) [26, 27]可作为补充, 实现更加可靠的密钥与敏感数据管理机制, 进而为移动终端提供良好的信任根. SE 是一种安全性极高的硬件模块, 通常是集成于移动设备主电路板上的一颗独立芯片或是直接封装在移动设备主芯片内部的一个独立单元, 能够提供 CC EAL5+级的硬件级安全防护能力, 覆盖全面的入侵检测与响应机制, 即使移动设备上其余的软件或硬件被恶意控制也能够保证驻留在 SE 的核心密钥和数据不被窃取或篡改. TEE+SE 的安全硬件组合目前已较为广泛的应用于移动终端设备, 为相关敏感应用提供高强度的防护能力, 为匿名离线支付提供安全的体系架构.

本文以 TEE+SE 安全技术架构为基础, 设计并实现了高效的移动平台双离线匿名支付方案 OAPM, 解决了移动用户电子支付过程中的隐私保护问题, 并满足数字货币交易的多项重要安全属性, 包括正确性, 不可

链接性, 可追踪性, 不可陷害性, 真实性, 防双花性以及可控匿名性. 本文的主要贡献如下:

1)基于 TEE+SE 安全硬件技术面向 CBDC 移动支付场景定制设计安全支付方案和完整的可信体系架构;

2)改进了 Canard 等人的 e-cash 协议, 使用比特分解技术使付款人可以一次性付出数字货币最大面额 2^n 以内的任意额度 v, 即 $1 \le v \le 2^n$ (其中 v 为整数, n 为非负整数), 让付款者支付数字货币更加灵活;

3)应用 CL 签名技术, 实现了数字货币的匿名分发功能, 更好地保障了付款行为的匿名性;

4)设计专门的预计算步骤, 减少付款者移动终端在支付阶段的时间开销, 提升效率;

5)设计实现了可控匿名属性, 在发现非法交易等特殊情况下, 可信权威机构(中央银行)能够揭露匿名交易的付款者身份.

2 相关工作

当前, 按照技术路线不同, 离线支付方案[28]可以分为 3 种类型. 第一种为离线 POS(Point Of Sales)终端交易方案, 文献[29]提出了一种面向移动终端用户与离线 POS 商户之间的电子支付系统模型, 但是 POS 交易的应用场景有限且不适用于 CBDC 移动终端的双离线支付场景. 第二种是基于区块链的离线支付方案, VolgaPay[30]基于区块链系统设计了安全高效的离线支付移动终端解决方案, 然而分布式的区块链系统无法满足 CBDC 核心系统中心化的设计原则. 第三种是利用密码学协议实现的离线支付方案, 能够同时保护用户隐私, 文献[31]和文献[32]借助不可追踪的盲签名技术(Blind Signature, BS)分别给出了一种安全的离线电子支付方案, 方案并没有考虑 CBDC 实际场景下的移动终端高效适配可行性, 且数字货币支付时的金额灵活度较差. 此外, 专门面向双层架构的 CBDC 体系, OPS[2]给出了一种双离线支付系统的设计方法, 但其缺乏足够的安全考量和具体的实现方式.

在与支付技术相关的移动安全领域, 从软件安全到硬件安全涌现了一系列研究成果. 基于纯软的安全技术方案如代码混淆或白盒加密等, 在面对来自操作系统底层和硬件层面的攻击时较为脆弱[33]. 基于主机端卡模拟(Host-based Card Emulation, HCE)技术方案[34]在近场通信(Near Field Communication, NFC)模块的加持下能够实现较为高效的移动支付功能, 其安全能力依赖于银行后台交易系统或模拟 SE 的本地应用程序, 无法满足双离线支付的高安全要求. 应用 TEE+SE 安全技术的移动支付方案 DOT-M[35], 给出了一种面向 CBDC 安全且高效的移动支付体系架构设计, 但该方案不支持支付者隐私保护, 没有实现匿名支付功能. 目前, 仍然缺乏面向 CBDC 移动支付场景设计的高安全双离线匿名支付方案.

3 方案设计

3.1 基础知识

本节主要介绍了包括常用符合在内的相关基础知识, 包括:本文将要用到的符号及定义;双线性群的定义;本文用到的困难问题假设.

3.1.1 常用符号

本文用到的常用符号及对应定义描述如表 1 所示.

表 1 常用符号与定义

符号	定义
λ	安全参数, 用于描述算法或系统的安全强度
$\mathbb{Z}$	整数集合
$\lvert\mathbb{S}\rvert$	有限集合 $\mathbb{S}$ 中的元素个数
$x \xleftarrow{\$} \mathbb{S}$	从集合$\mathbb{S}$中均匀随机选取元 x
$y := x$	将变量 x 的值或内容赋于变量 y
$x\|\|y$	变量 x 级联变量 y
$(y_1,\ldots,y_j) \leftarrow A(x_1,\ldots,x_i)$	将变量 $x_1,\ldots,x_i$ 输入某一算法 A, 输出变量 $y_1,\ldots,y_j$
$1_{\mathbb{G}}$	群 $\mathbb{G}$ 上的单位元
$\mathbb{G}^*$	去除单位元的群 $\mathbb{G}$, 即 $\mathbb{G}\backslash\{1_{\mathbb{G}}\}$
"abcd"	字符串 abcd
$MAC(k,m)$	使用密钥 k 对数据m计算消息认证码
$Enc_{sym}(k,m)$	使用对称加密算法和密钥 k 对数据 m 进行加密

符号	定义
$\mathrm{Dec}_{\mathrm{sym}}(k,m)$	使用对称解密算法和密钥 k 对数据 m 进行解密
$\mathrm{Enc}_{\mathrm{asym}}(k,m)$	使用非对称加密算法和密钥 k 对数据 m 进行加密
$\mathrm{Dec}_{\mathrm{asym}}(k,m)$	使用非对称解密算法和密钥 k 对数据 m 进行解密
$\mathrm{Sign}(k,m)$	使用签名密钥 k 对数据 m 进行签名
$\mathrm{Veirfy}(k,m)$	使用验证密钥 k 对数据 m 进行验证

3.1.2 双线性群

令群 $\mathbb{G}_1$, $\mathbb{G}_2$ 和 $\mathbb{G}_T$ 是三个具有相同阶 p 的乘法循环群, p 为大素数, $p\approx 2^{\lambda}$, 其中 λ 为安全参数, g_1, g_2 分别是 $\mathbb{G}_1$, $\mathbb{G}_2$ 的生成元, 则定义 $\Lambda=(p,\mathbb{G}_1,\mathbb{G}_2,\mathbb{G}_T,e,g_1,g_2)$ 为双线性群的参数描述, 在 $\mathbb{G}_1$ 和 $\mathbb{G}_2$ 上的一个双线性对(Pairing)即双线性映射 $e:\mathbb{G}_1\times\mathbb{G}_2\to\mathbb{G}_T$ 满足以下性质:

1)双线性:对于所有 $u\in\mathbb{G}_1$, $v\in\mathbb{G}_2$ 以及所有 $a,b\in\mathbb{Z}_p$, 有 $e(u^a,v^b)=e(u,v)^{ab}$;

2)非退化性: $e(g_1,g_2)\neq 1_{\mathbb{G}_T}$;

3)可计算性:对于所有 $u\in\mathbb{G}_1$, $v\in\mathbb{G}_2$, 存在多项式时间的算法计算映射 $e(u,v)$.

Galbraith 等人在文献[36]中依据底层群的不同结构将双线性映射分成三类, 如果 $\mathbb{G}_1=\mathbb{G}_2$, 则称 e 为对称双线性映射(即 Type-1 型映射), 否则称为非对称双线性映射(即 Type-2 型或 Type-3 型映射[37]), 其中 Type-2 型要求满足 $\mathbb{G}_1$ 和 $\mathbb{G}_2$ 之间存在已知的高效可计算同态: $\mathbb{G}_2\to\mathbb{G}_1$, 不满足该要求的非对称双线性映射为 Type-3 型, 本文内容主要使用了 Type-3 型的双线性映射.

3.1.3 困难问题假设

本文设计的 OAPM 方案用到了 3 种困难问题假设, 分别是群 $\mathbb{G}_1$ 上的判定性 Diffie-Hellman(Decisional Diffie-Hellman, $\mathrm{DDH}_{\mathbb{G}_1}$)假设, 弱扩展的交叉 Diffie-Hellman(weak extended version of cross-Diffie-Hellman, Weak-EXDH)假设[12]和盲化的 4 元 LRSW(blind 4-Lysyanskaya-Rivest-Sahai-Wolf, B-4-LRSW)假设[38]. 其中, Weak-EXDH 假设是 XDH 假设的一个变形, B-4-LRSW 假设是 LRSW 假设[39]的一个变形, 以下将给出本文涉及到的 3 种困难问题假设定义.

定义 1 ($\mathrm{DDH}_{\mathbb{G}_1}$ 假设). 给定 $\mathbb{G}_1$ 上的 3 个元素 (g_1,g_1^a,g_1^b), 其中 $a,b\xleftarrow{\$}\mathbb{Z}_p$, 则将 g_1^{ab} 与群 $\mathbb{G}_1$ 上的任意一个元素区分是困难的.

定义 2 (Weak-EXDH 假设). 给定双线性群参数 Λ, $\mathbb{G}_1$ 上的 3 个元素 (g_1^a,g_1^b,g_1^{bc}) 以及 $\mathbb{G}_2$ 上的 2 个元素 (g_2^b,g_2^c), 其中 $a,b,c\xleftarrow{\$}\mathbb{Z}_p$, 则将 g_1^{abc} 与 $\mathbb{G}_1$ 上的任意一个元素区分是困难的.

定义 3 (B-4-LRSW 假设). 给定双线性群参数 Λ 和 $\mathbb{G}_2$ 上的 2 个元素 (g_2^x,g_2^y), 其中 $x,y\xleftarrow{\$}\mathbb{Z}_p$, 存在一个预言机 oracle 使输入 $g_1^m\in\mathbb{G}_1$ 对应输出 (A,A^y,A^{x+mxy},A^{my}), 其中 $A\xleftarrow{\$}\mathbb{G}_1^*$, 则该预言机在从未输入 $g_1^{m^*}$ 的情况下输出 (m^*,A^*,B^*,C^*,D^*) 且满足 $m^*\in\mathbb{Z}_p^*$, $A^*\in\mathbb{G}_1^*$, $B^*:=(A^*)^y$, $C^*:=(A^*)^{x+m^*xy}$ 以及 $D^*:=(A^*)^{m^*y}$ 是困难的.

3.2 设计目标

因为本文 OAPM 方案是面向实际的通用移动设备, 所以在设计时考虑了以下 6 个目标:

1)高安全性:OAPM 能够在移动平台上建立有效的可信执行环境, 该环境能够可靠地保护其中运行的安全敏感程序及相关数据, SE 模块能够安全地提供根密钥种子, 派生适用于不同需要的密钥, 并协助架构建立可靠的信任链, 数字货币支付协议具备必要的安全属性, 包括正确性, 不可链接性, 可追踪性, 不可陷害性, 机密性和真实性, 这些安全属性能够通过安全性分析得到证明;

2)高可控性:OAPM 在数字货币体系层面具备中央银行发行可控, 双花可发现可追溯以及可控匿名的能力, 有利于金融监管机构掌握货币主权, 减少非法交易等恶劣行为的隐秘发生;

3)高效率性:OAPM 对通用移动平台设备引入的额外运算时间开销较小, 因系统模式切换或安全机制触发而造成的时延对用户正常使用设备不构成明显的影响, 此外 TEE 架构对系统计算资源和存储资源的消耗也应较小;

4)高灵活性:OAPM 的可信体系架构从底层硬件, 外接设备, 安全操作系统, 应用程序可信单元到通用操

作系统和应用程序组件, 给予系统及应用程序开发者足够的设计与扩展空间, 使其能够设计并实现适用于多种应用场景的安全解决方案.

4 系统模型

本节将介绍 OAPM 的系统模型, 首先给出方案的交互模型, 解释方案各个参与实体的身份与功能, 随后描述方案的安全假设和敌手模型.

4.1 交互模型

针对双离线匿名支付的应用场景, 本解决方案是以典型的电子支付需求为出发点进行设计的. 在本节给出的移动平台匿名支付方案 OAPM 的系统交互模型中, 共有 5 种参与实体:付款者移动设备 $\mathcal{D}$ (Mobile Device of Payer), 收款者移动设备 $\mathcal{M}$ (Mobile Device of Payee), 在线银行系统 $\mathcal{B}$ (Online Bank System), 可信权威 $\mathcal{T}$ (Trusted Authority)和传统商业银行. 在实际应用场景中, 同一个交互系统可能存在若干个不同的 $\mathcal{D}$ 和 $\mathcal{M}$, 这里为了描述简洁, 只考虑一个实体 $\mathcal{D}$ 和一个实体 $\mathcal{M}$. $\mathcal{D}$ 和 $\mathcal{M}$ 是移动用户直接接触和操作的实体, 配备了支持 TEE 安全扩展技术的处理器芯片, SE 安全单元以及所需的外设, 本方案主要面向以 ARM TrustZone 技术实现的 TEE 进行设计, 在下文描述正常的协议交互时, 为了表述一致性, $\mathcal{D}$ 和 $\mathcal{M}$ 分别指代广义的付款者和收款者行为. $\mathcal{B}$ 负责通过 **Withdraw** 阶段签发数字货币给合法的(或可信的) $\mathcal{D}$, 在现实世界中, $\mathcal{B}$ 可以是支持电子支付的银行卡联合组织(如 VISA, MasterCard 和中国银联)系统或第三方支付机构(如 PayPal, 支付宝和微信支付)平台系统, 也可以是网络支付清算平台系统或数字货币互联互通平台系统. 在 $\mathcal{B}$ 的后台, 连接着若干传统商业银行, 这里是付款者存储物理货币的位置, 收款者也在这里拥有自己的账户, 商业银行将配合 $\mathcal{B}$ 处理所有真实世界物理货币归属权变更的事务. 在交互模型中, 收款者可以是普通的个人, 也可以是专门登记的商户, 收款者在双离线交易情况下使用与付款者相似安全配置的移动终端设备, 他们通过 **Spend** 阶段从 $\mathcal{D}$ 获得数字货币, 并通过 **Deposit** 阶段从 $\mathcal{B}$ 处将这些数字货币兑现成物理货币. 值得注意的一点是, $\mathcal{M}$ 在验证付款者数字货币时, 付款者的身份不会被泄露给包括 $\mathcal{M}$ 在内的任何一个实体. 在双离线的应用场景中, $\mathcal{D}$ 可以通过无线通信技术(如 NFC 或蓝牙)向 $\mathcal{M}$ 支付数字货币. 由政府, 中央银行或行业管理机构运营的 $\mathcal{T}$, 在 **Capture** 阶段捕获试图双重花费数字货币的付款者身份, $\mathcal{T}$ 可以从 $\mathcal{B}$ 处获取关于双花的报告和相关证据, 并识别双花者的身份信息, 以实现防双花的属性. 类似地, 根据 $\mathcal{B}$ 的相关报告, $\mathcal{T}$ 可以在 **Identify** 阶段揭露非法交易的付款者身份, 以实现可控匿名属性. 由于可分 e-cash 协议的属性限制, 交互模型不考虑数字货币二次支付的问题, 只关注支付——提现(Spend-Deposit)这样的一次性货币转移流程, 但这不影响收款者在其他交易中作为一个付款者使用同样安全等级的移动设备花费自己从 $\mathcal{B}$ 取出的数字货币. 图 1 描绘了 OAPM 的系统交互模型. 在实际应用中, 本方案也同样适用于非离线的匿名支付场景, 此时收款者可以作为一个固定商户使用 POS 机或 X86 架构的收银机进行收款.

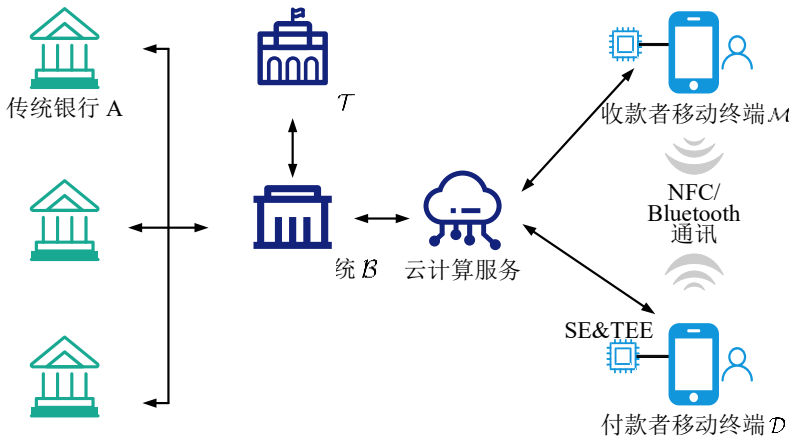

图 1 移动平台匿名支付方案系统交互模型

4.2 安全假设与敌手分析

在系统模型中, 为了简化方案的设计, 假设 $\mathcal{B}$ 与传统商业银行之间以及 $\mathcal{B}$ 与 $\mathcal{T}$ 之间的数据通信建立在诸如 TLS/SSL 的安全传输层协议基础之上, 数据的传输受到机密性, 完整性和认证性的保护, $\mathcal{M}$, $\mathcal{D}$ 和 $\mathcal{B}$ 可以正确获取 $\mathcal{T}$ 发送的公共参数. 此外, 假设 PKI 技术已被应用于认证 $\mathcal{B}$ 和 $\mathcal{M}$ 的身份, 因此存在两点既成事实:

1) $\mathcal{D}$ 和 $\mathcal{M}$ 可以通过验证公钥证书的方式正确地获取$\mathcal{B}$的公钥;

2)类似地, $\mathcal{D}$ 和 $\mathcal{B}$ 可以正确地获取 $\mathcal{M}$ 的公钥.

OAPM 的良性运转需要一些前提信任关系的建立. 首先 $\mathcal{B}$ 被信任不会发行虚假伪造的数字货币. 其次, 移动设备制造商被信任只为支持 TEE 技术且符合一定安全标准的移动设备 $\mathcal{D}$ 颁发设备证书. 最后, 一个被默认的事实是 $\mathcal{T}$ 不会恶意地与其他实体合谋以实施不合规定的行为. 受到市场的信誉监督和政府的法律监管, 上述这些信任关系都可以较为容易地被建立和长期维持.

OAPM 以移动设备安全为出发点, 不考虑 $\mathcal{B}$ 和 $\mathcal{T}$ 自身内部的恶意行为以及可被敌手利用的安全漏洞. 由于 OAPM 构建在移动平台 TEE+SE 架构之上, 根据上述安全假设, OAPM 可以抵御具有以下能力的敌手:

1)敌手试图使用伪造实体, 操纵实体间数据传输和伪造数据的方式攻击 OAPM 方案本身, 以此获取 $\mathcal{D}$ 或付款者的真实身份及相关隐私信息, 或者获取相关用户的数字或物理财产;

2)敌手可以对 REE 实施基于软件的攻击, 也可以访问 SW 中可信服务和 OAPM 相关程序在 NW 中的接口;

3)敌手攻击移动设备的通用非安全存储器, 试图获取用户的数字货币以及 $\mathcal{D}$ 的其它秘密信息, 如密钥等;

4)敌手通过软件或物理方式重启移动设备, 试图访问启动过程中的敏感数据;

5)敌手可以对移动设备的 SE 执行实验室级的攻击或对 SE 进行侧信道攻击, 以获取重要的密钥;

6)敌手试图伪造特定付款者或收款者的身份, 与他人执行离线交易, 以此打破不可伪造性;

7)恶意付款者尝试制造一个不受信任的移动设备, 或操纵其可信的移动设备来执行离线交易, 故意违背本方案流程;

8)恶意付款者试图双花他的数字货币, 以此违背防止双花的属性;

9)恶意付款者试图否认他确实发生过的支付行为, 以此违背不可抵赖的属性;

10)恶意收款者试图否认他确实发生过的收款行为, 以此违背不可抵赖的属性.

本方案同样信任提供 TEE 技术的终端硬件安全性, 不考虑针对 TEE 的高级物理攻击和侧信道攻击.

5 支付方案

本节将具体介绍本文提出的基于 TEE+SE 和可分 e-cash 技术的移动平台双离线匿名支付方案 OAPM, 首先给出方案的移动终端可信体系架构, 随后说明了本方案使用到的密钥与敏感数据管理方法, 之后结合体系架构和管理方法详细阐述基于 TEE+SE 的双离线匿名支付协议, 最后给出其它可选的防御机制.

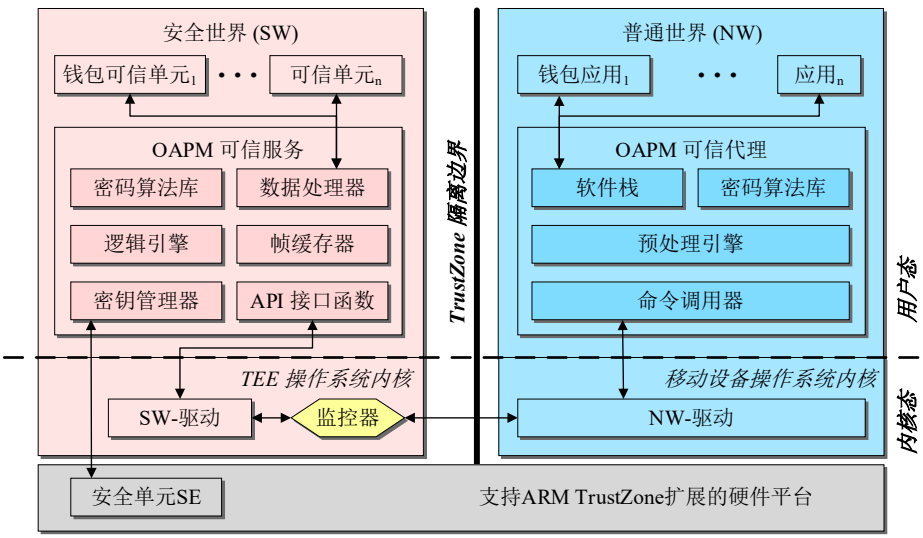

图 2 OAPM 的移动终端可信体系架构

5.1 移动终端可信体系架构

本节面向匿名支付场景提出的移动平台 OAPM 体系架构, 是借助 TEE+SE 安全技术进行设计的, 架构部署于交互模型的 $\mathcal{D}$ 和 $\mathcal{M}$, 因为付款者离线匿名支付行为的安全需求更高, 所以后文的安全设计主要以 $\mathcal{D}$ 的角色需求进行描述, 而仅作为收款方 $\mathcal{M}$ 的设计可以依据 $\mathcal{D}$ 的架构做大规模简化, 本文不再赘述. 在现有的移动平台硬件架构基础上, 本节的 OAPM 构建方法以基于软件的设计和实现为主, 通过 TrustZone 技

术实现 TEE, 图 2 展示了本节给出的 OAPM 体系架构以及各组件之间的交互细节. 通过成熟的信任链构建技术, 可以在 TrustZone 的 SW 中构建安全有效的可信执行环境, 即 TEE. 位于 SW 中的 TEE 隔离于 NW 中实现的通用移动执行环境, 即 REE. 具有安全内核的定制型 TEE 操作系统(Operating System, OS)运行在 SW 中, 为上层执行的安全敏感应用程序或服务代码提供系统服务, 而通用移动 OS 运行在 NW 中, 该 OS 既可以是 Android 系统也可以是 iOS 系统, 它们为上层的常规移动应用程序提供系统服务. 在这里, 架构设计借鉴和吸收了可信计算技术, 在 SW 中加入了可以提供可信服务的模块供上层安全应用调用, 这一模块的可信和安全是广义的, 其功能可以根据具体的需求进行灵活地设计与实现. 下面详细介绍各组件的功能.

(1)OAPM 可信服务

可信服务组件是 OAPM 架构 TEE 部分的核心组件, 运行于 SW 的 TEE OS 上, 不仅实现了信任根呈现, 密钥与敏感数据管理和可信环境证明等可信计算相关功能, 还提供 OAPM 核心算法和其它功能. OAPM 可信服务组件实现了基于 TEE 和 SE 的匿名支付协议在移动设备 $\mathcal{D}$ 的关键步骤, 包括密钥与秘密信息初始化设置, 提取数字货币和花费数字货币, 其执行代码是安全敏感的, TrustZone 隔离技术对其进行严格的保护. 以下具体介绍可信服务的 6 个基本子组件:

1)密码算法库:为数据处理器, 密钥管理器和逻辑引擎提供丰富的密码学算法支持, 其根据具体应用需求可以实现对称和非对称加解密与签名验证算法及多种类型的消息摘要算法等;

2)数据处理器:用于接收付款者通过移动应用向可信服务发送的安全敏感指令和数据, 为了防止敌手伪造安全参数(通常是用户名和口令), 数据处理器只接收来自 SW 中移动应用程序可信单元(如钱包应用程序可信单元)的参数输入, 并将参数传递给逻辑引擎, 此外该子组件还负责敏感数据的封装与解封, 封装后的数据可以存储在移动设备平台的通用非易失性存储器中(如机身内部存储器或 SD 卡);

3)逻辑引擎:从其它子组件获取必要的参数输入, 依据所设计的匿名支付协议, 算法或服务逻辑, 执行移动平台各类安全敏感的可信服务运算操作, 输出执行结果, 此外该子组件实现了对 SW 中移动应用程序可信单元的加载度量和启动管控, 必要时可以实现向外界提供进程完整性证明的服务;

4)帧缓存器:用于付款者在支付数字货币时对收款者的身份确认, 事实上 Framebuffer 存储了支付时的待确认信息画面并负责将其安全地显示给付款者观看, 该设计能够防止敌手篡改交易确认信息的显示, 不同于 NW 中的通用画面帧缓存器, Framebuffer 致力于构建可信赖的用户交互界面(Trusted User Interface, TUI);

5)密钥管理器:利用保存在 SE 模块中的根密钥种子生成多种密码学运算所需的密钥, 将这些密钥提供给数据处理器以备使用;

6)应用程序接口函数:接收来自 NW 中可信代理发送的可信服务请求, 解析命令数据, 将运算指令传递给逻辑引擎并等待结果返回可信代理, API 设计参考 GP TEE Internal API 规范[40].

(2)OAPM 可信代理

可信代理运行于 NW 的通用移动 OS 上, 直接与上层移动应用程序交互. 该组件接收移动应用程序的可信服务请求, 根据请求类型及内容组装调用 SW 中可信服务组件的命令, 为 SW 中的实质性安全敏感运算做准备. 该组件亦可根据具体需求增添若干附加功能的子组件, 但应该包含如下 4 个基本子组件:

1)软件栈:为 NW 中的移动应用程序(如钱包应用程序)提供高层可信服务命令接口, 负责解析应用程序的请求数据, 并返回可信服务的响应结果;

2)命令调用器:组装调用可信服务的命令结构, 与 SW 中的可信服务组件交互, 通过规范的 TEE 客户端命令接口(GP TEE Client API 规范[41])实现命令的发送, 借助 NW OS 内核的驱动组件请求 NW 向 SW 的模式切换并等待数据返回;

3)专门用于 OAPM 协议预计算的引擎:该子组件在付款者使用移动设备 $\mathcal{D}$ 成功从 $\mathcal{B}$ 提取一定量数字货币后被触发执行预计算逻辑;

4)密码算法库:提供了用于支持预计算引擎的密码学算法, 实现时这部分算法库只需支持椭圆曲线上的指数运算即可.

(3)钱包应用程序和钱包应用程序可信单元

如果用户想使用数字货币的离线交易, 就需要启动相应的应用程序, 即相关钱包应用. 由 $\mathcal{B}$ 负责发行的数字货币钱包应用程序由两部分组成:一个 NW 的钱包应用程序和一个 SW 的钱包应用程序可信单元. 钱包应用程序即是当前通用的移动应用程序客户端, 它可以向用户提供各种应用衍生功能, 但不涉及安全敏感的代码执行. 而钱包应用程序可信单元在 SW 中通过安全启动被加载执行, 用于处理安全敏感的用户输入(如用户名口令或支付确认)和数据操作. 当钱包应用程序察觉需要执行匿名支付的核心协议时, 将通过调用 OAPM 代理的软件栈触发 SW 中 OAPM 服务组件的协议执行. 同样地, 钱包应用程序使用 TrustZone 提供的域间通信机制通知位于 SW 中的钱包应用程序可信单元准备接收付款者的敏感操作.

(4)内核中的组件

在 SW 的 TEE OS 内核中, 部署有驱动组件 SW-Driver;在 NW 的移动 OS 内核中, 部署有驱动组件 NW-Driver. 以上两个驱动组件用于处理在 TrustZone 两个世界切换时的请求与响应命令, 其中包含了两个世界的通信数据. 作为 TrustZone 定义的安全监视器的实现, 监视器(Monitor)位于 SW 的系统内核中, 其控制底层硬件完成 TrustZone 世界切换的具体动作. 除了这些特殊的组件外, NW 中的 OS 内核实现有各种通用的硬件驱动模块, 这其中包含有网络通信驱动, 移动平台与外界的数据通信均依赖于该驱动.

(5)硬件中的组件

本节提出移动平台 OAPM 架构的底层硬件支持 ARM TrustZone 扩展技术, 受到该技术的保护, 位于硬件中的 SE 模块仅能被 SW 的组件访问. SE 具有能够抵抗实验室级攻击的多种硬件保护机制, 致力于设备根密钥种子的生成和保护, 作为 OAPM 架构的信任根. 此外, TrustZone 技术在 ARM 硬件架构上提供了多种支持隔离的安全硬件模块, 包括安全内存, 安全寄存器和安全非易失性存储器等, 还可以将一些 I/O 外设(如指纹识别模块, LED 指示器或显示屏等)划分或标记到安全域中以实现更多的安全辅助功能. 上述这些安全硬件模块受到 TrustZone 隔离技术的保护, 仅支持来自 SW 中的软硬件和处于安全状态的 CPU 的访问, 且都可以作为 TEE 的可选组件为具体的移动安全解决方案实施提供有力帮助.

5.2 密钥与敏感数据管理

本节为移动平台双离线匿名支付方案 OAPM 设计了移动平台敏感数据管理机制, 方案中基于 TEE 和 SE 实现了根密钥种子的提取, 密钥生成和敏感数据封装方法, 充分利用 TEE 和 SE 技术特性保护密钥与敏感数据管理过程.

(1)根密钥种子的提取

本节利用 SE 内部的真随机数发生器和非易失性存储区, 实现了提取和存储移动平台的根密钥种子 s 的方法. 移动平台设备制造商在设备生产过程中预置根密钥种子生成程序到 SE. 当移动平台初次上电启动 SE 时, 触发根密钥种子生成程序从真随机数发生器读取一随机比特串保存到非易失性存储区, 作为平台的根密钥种子 s.其机密性, 完整性受到 SE 安全隔离技术的严格保护.

(2)密钥生成

密钥体系是构建移动平台安全架构的一个重要环节, 也是可信计算技术的关键组成部分. 本节参考可信计算标准[42], 结合 TEE 和 SE 各自技术特点, 在移动平台上设计和实现了一套密钥体系, 支持多种类型的密钥生成, 可以为上层可信或安全服务及应用提供灵活的密码学功能支撑.

在移动平台 SE 实现了密钥生成函数(Key Derivation Function, KDF), 该函数是一种确定性的映射 $\mathsf{KDF}:\tilde{\mathcal{S}}\times\{0,1\}^*\to\tilde{\mathcal{K}}$,其中 $\tilde{\mathcal{S}}$ 是密钥种子空间, $\{0,1\}^*$ 是用于声明密钥用途和绑定关系的输入参数比特串, $\tilde{\mathcal{K}}$ 是生成目标密钥的空间. KDF 内部包含了多种类型的密钥生成算法, 生成密钥的安全强度由系统安全参数决定, 其对外提供统一的调用接口. KDF 只接受来自于 SW OAPM 可信服务组件的外部调用指令, 根据不同指令和参数组合可以生成多种不同类型的密钥, 以下着重介绍设备密钥和存储根密钥.

设备密钥(Device Key). 设备密钥的实质是一对非对称密钥, 可以用于标识移动设备平台的唯一身份, 在设备身份证明或与外界建立初次连接时具有重要意义. 设备密钥由根密钥种子 s 生成, 其具体种类和生成算法可以根据实际应用需求进行实现, 实现代码被包含在 KDF 中, 而调用 KDF 生成设备密钥公私钥对 $(\mathsf{dpk},\mathsf{dsk})$ 的方式为:

$$(\mathsf{dpk},\mathsf{dsk})\leftarrow\mathsf{KDF}_{\mathsf{s}}(\text{"identity"}) \tag{1}$$

其中: s 为根密钥种子, 是设备密钥的父密钥, "identity"是生成设备密钥的命令标识字符串. 由于设备密钥的唯一属性, 一个移动平台设备通过 KDF 只能生成一个对应的设备密钥对, 其私钥仅能在 SE 中使用, 具体来说该私钥只能由 SE 生成和被 SW OAPM 可信服务组件调用. 设备密钥的公钥一般需要被可信第三方或机构认证才能生效, 在实际应用中, 通常由移动设备制造商对设备公钥进行认证, 厂商使用自己的合法私钥对设备密钥的公钥进行签名并颁发设备证书. 设备证书可以承诺对应的平台设备受到厂商的认证并满足若干规范或功能, 该证书中还包含有厂商对移动平台固件的代码完整性度量值签名, 用于信任链的构建.

存储根密钥(Storage Root Key). 存储根密钥的实质是一个对称密钥, 可以用于进一步生成多种类型的存储密钥. 移动平台设备的存储根密钥也是由根密钥种子 s 生成, 调用 KDF 生成存储根密钥 srk 的方式为:

$$\mathsf{srk}\leftarrow\mathsf{KDF}_{\mathsf{s}}(\text{"storage\_root"}) \tag{2}$$

其中:"storage_root"是生成存储根密钥的命令标识字符串. 同样的, 一个移动平台设备只有一个对应的存储根密钥, 仅能在 SE 组件中生成和被 SW OAPM 可信服务组件调用, 为了防止与其相关的密文多次暴

露于通用执行环境而降低密钥的安全性, 存储根密钥不能直接用于加密数据生成密文, 仅用于生成其它存储密钥, 该套存储密钥的层次结构能够有效地增强密钥使用的隔离性, 这一点同可信计算中的存储密钥保护机制相似.

类似地, 可以使用KDF生成其它用途的密钥, 例如签名认证密钥和通信保护密钥等, 这些都可以根据具体的安全应用解决方案进行定制. 值得强调的一点是, 通过以上方式生成的设备密钥, 存储根密钥的秘密部分都不离开 SE, 且只能每次使用前重新派生;由存储根密钥派生的存储密钥可以基于安全通讯信道传输到TEE并被TEE的可信组件使用, 且只能每次使用前重新派生, 这样可以极大地减小敏感密钥泄露风险.

在方案 OAPM 中, 移动设备依然具有可以唯一标识自己身份的设备密钥公私钥对, 这里记为 (dpk,dsk). 方案使用由存储根密钥 srk 派生的多种存储密钥对 OAPM 的系统公共参数 params, 移动设备 $\mathcal{D}$ 的数字货币 σ 和数字货币密钥 m 连同其它相关变量 CT 和 δ 进行封装存储保护. 其中, 从 $\mathcal{B}$ 提取的未花费的一定量数字货币记为 σ, 一个 $\mathcal{D}$ 可能同时拥有若干不同的 σ, 需要被分别封装保护. 封装操作在 SW OAPM 服务组件的数据处理器中进行, 继续使用数据处理器的数据封装函数 Data_Seal 实现封装操作, 封装生成的数据块可以存储在移动设备非安全的 NW 环境中. 以下给出具体的封装方法:

对于公共参数 params, 封装时只需保护其完整性, 步骤为:

$$mk_{params} \leftarrow KDF_{srk}("storage\_key" \| "MAC" \| params) \tag{3}$$

$$blob_{params} \leftarrow Data\_Seal("MAC", mk_{params}, params) \tag{4}$$

其中:

$$blob_{params} := params \| MAC(mk_{params}, params) \tag{5}$$

KDF 的输入参数 params 标识了所生成的完整性保护密钥是用于保护系统公共参数的.

对于数字货币 σ, 封装时只需保护其完整性, 步骤为:

$$mk_{\sigma} \leftarrow KDF_{srk}("storage\_key" \| "MAC" \| \sigma) \tag{6}$$

$$blob_{\sigma} \leftarrow Data\_Seal("MAC", mk_{\sigma}, \sigma) \tag{7}$$

其中: $blob_{\sigma} := \sigma \| MAC(mk_{\sigma}, \sigma)$, KDF 的输入参数 σ 标识了所生成的完整性保护密钥是用于保护该数字货币 σ 的.

对于数据 m, CT 和 δ, 封装时需要借助变量 U [1]的帮助保护其机密性和完整性, 步骤为:

$$(sk_m, mk_m) \leftarrow KDF_{srk}("storage\_key" \| "Enc+MAC" \| U) \tag{8}$$

$$blob_m \leftarrow Data\_Seal("Enc+MAC", sk_m, mk_m, m \| CT \| \delta, U) \tag{9}$$

其中:

sk_m 为对称密钥, $blob_m := Enc_{sym}(sk_m, m \| CT \| \delta) \| U \| MAC(mk_m, Enc_{sym}(sk_m, m \| CT \| \delta) \| U)$. KDF 的输入参数 U 标识了所生成的机密性和完整性保护密钥是用于保护对应数据 m, CT 和 δ 的.

利用密钥管理器中重构出的存储密钥, 数据处理器可以调用 Data_Unseal ()函数从相应的数据块中恢复并验证敏感数据.

5.3 基于TEE+SE的双离线匿名支付协议

参考原有可分 e-cash 协议[12], 图 3 给出了支付系统的一棵唯一的(unique)公共全局二叉树, 二叉树的深度若为 n 则表明当前系统的每枚数字货币最大面额为 $V = 2^n$, 这棵全局二叉树被同一个系统内的所有数字货币共用, 每一个叶子结点代表了可以被花费的最小面值单位的数字货币. 这里定义 S_n 为比特串的集合, 所含的每个比特串的长度小于或等于 n;定义 $\mathcal{F}_n$ 也为比特串的集合, 所含的每个比特串的长度均等于 n. 因此, 图中二叉树的任何一个结点可以被描述为一个比特串元素 $s \in S_n$, 而根结点为空比特串 ϕ, 每个叶子结点可以被描述为一个比特串元素 $f \in \mathcal{F}_n$. 对于任意结点 $s \in S_n$, 定义集合 $\mathcal{F}_n(s) = \{f \in \mathcal{F}_n \mid s \text{ is a prefix of } f\}$, 该集合包含了以结点 s 为根所构成的子二叉树下面的所有叶子结点比特串.

---

[1] 在 OAPM 方案中, 对于某个固定的群基底 g_1, $U = g_1^m$.

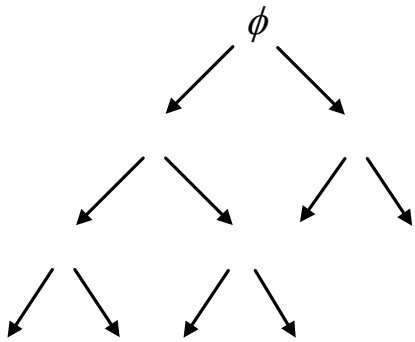

图 3 适用于所有数字货币的公共全局树型结构

假设出厂前，$\mathcal{D}$ 由原始设备厂商(OEM)初始化，在 SW 中生成的唯一的设备密钥对可以唯一标识设备 $\mathcal{D}$ 的身份，那么 OEM 为其公钥 dpk 颁发有设备证书 $\mathsf{cert}_{\mathcal{D}}$，证书表明了 OEM 对 $\mathcal{D}$ 的认可. 此外，证书 $\mathsf{cert}_{\mathcal{D}}$ 还包含有 $\mathcal{D}$ 的其它配置信息(例如 TrustZone 是否可用以及版本号等).

OAPM 方案的双离线匿名支付协议总共包含 7 个执行阶段: 初始化设置(**Setup**), 密钥生成(**KeyGen**), 提取数字货币(**Withdraw**), 花费数字货币(**Spend**), 商户提现数字货币(**Deposit**), 捕获双花者身份(**Capture**)和揭露非法付款者身份(**Identify**). 协议的执行概述如图 4 所示, 7 个执行阶段可以分为三类, 使用不同颜色表示. 其中, 蓝色代表系统初始化类, 包括由 $\mathcal{T}$ 执行以创建系统公共参数的 **Setup**、$\mathcal{B}$ 和 $\mathcal{M}$ 生成各自公私钥对的 **KeyGen**; 绿色代表数字货币流通环节类, 包括 $\mathcal{D}$ 从 $\mathcal{B}$ 提取数字货币所执行的 **Withdraw**、$\mathcal{D}$ 向 $\mathcal{M}$ 匿名支付数字货币所执行的 **Spend**、$\mathcal{M}$ 从 $\mathcal{B}$ 提现数字货币所执行的 **Deposit**; 红色代表交易行为异常处理类, 包括赋予 $\mathcal{T}$ 捕获双花者身份能力的 **Capture** 和赋予 $\mathcal{T}$ 揭露非法付款者身份能力的 **Identify**. 系统初始化类两个阶段执行后, 其它各阶段可以根据实际需求被特定实体触发执行.

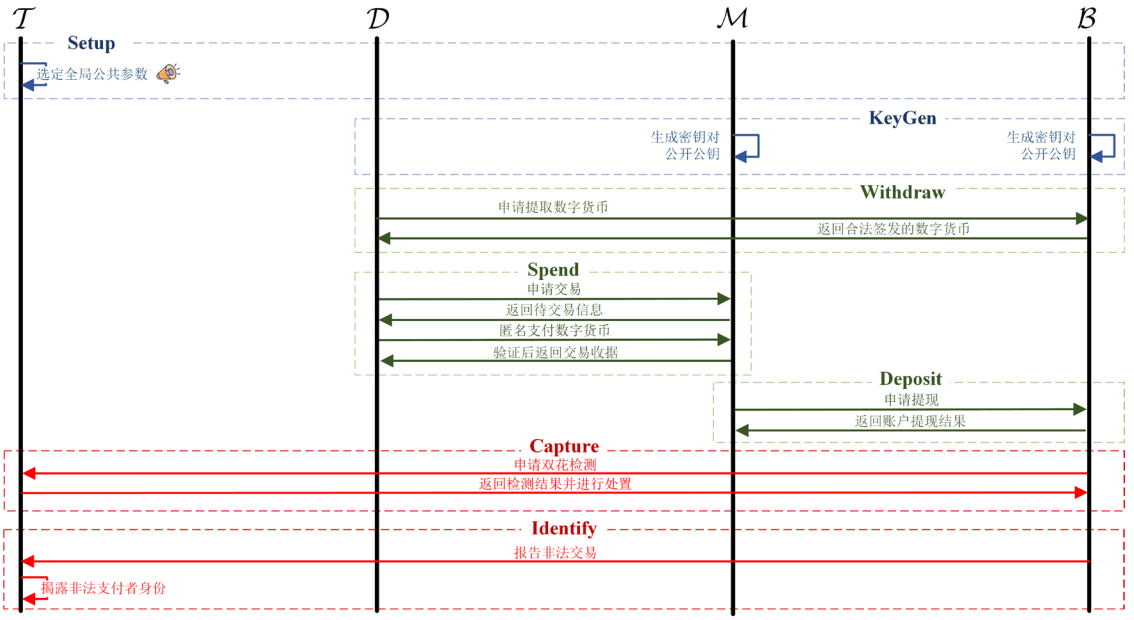

图 4 双离线匿名支付协议的执行概述

结合表 1 的符号定义，表 2 给出了本方案协议使用到的部分算法符号定义. 以下是对协议内容的详细描述.

表 2 匿名支付协议部分密码学算法符号和定义

符号	定义
$\sum_1 = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$	含密钥生成的数字签名算法
$\sum_2 = (\mathsf{MAC})$	消息摘要算法
$\sum_3 = (\mathsf{Enc}_{\mathsf{asym}}, \mathsf{Dec}_{\mathsf{asym}})$	非对称(公钥)加解密算法
$\sum_4 = (\mathsf{Enc}_{\mathsf{sym}}, \mathsf{Dec}_{\mathsf{sym}})$	对称加解密算法

Setup. 在本阶段中, 可信权威 $\mathcal{T}$ 创建公共参数. 给定安全参数 λ, $\mathcal{T}$ 选取合适的双线性群参数 $\Lambda := (\mathsf{p}, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_\mathsf{T}, \mathsf{e}, \mathsf{g}_1, \mathsf{g}_2)$, 要求 p 的比特串长度至少为 2λ. 之后, 根据全局二叉树的定义, $\mathcal{T}$ 执行:1)对于每一个结点 $\mathsf{s} \in \mathcal{S}_\mathsf{n}$, 选取 $\mathsf{r}_\mathsf{s} \xleftarrow{\$} \mathbb{Z}_\mathsf{p}$, 计算 $\mathsf{g}_\mathsf{s} := \mathsf{g}_1^{\mathsf{r}_\mathsf{s}}$;2)对于每一个结点 $\mathsf{s} \in \mathcal{S}_\mathsf{n}$ 以及每一个结点 $\mathsf{f} \in \mathcal{F}_\mathsf{n}(\mathsf{s})$, 选取 $\mathsf{l}_\mathsf{f} \xleftarrow{\$} \mathbb{Z}_\mathsf{p}$, 计算 $\tilde{\mathsf{g}}_{\mathsf{s}\to\mathsf{f}} := \mathsf{g}_2^{\mathsf{l}_\mathsf{f}/\mathsf{r}_\mathsf{s}}$. $\mathcal{T}$ 保存集合 $\mathsf{sck} = \{\mathsf{r}_\mathsf{s} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n}\}$ 作为密钥集供 **Capture** 阶段和 **Identify** 阶段使用. 此外, $\mathcal{T}$ 选定一系列密码学算法 Ψ, 包含算法 $\sum_1$ 至 $\sum_4$ 以及如下 4 个独立的抗碰撞哈希算法:

$$\mathsf{H}_1 : \{0,1\}^* \to \mathbb{Z}_\mathsf{p} \tag{10}$$

$$\mathsf{H}_2 : \{0,1\}^* \to \mathbb{Z}_\mathsf{p} \tag{11}$$

$$\mathsf{H}_3 : \{0,1\}^* \to \{0,1\}^{2\lambda} \tag{12}$$

$$\mathsf{H}_4 : \{0,1\}^* \to \{0,1\}^{2\lambda} \tag{13}$$

最后, $\mathcal{T}$ 设置 $(\Lambda, \mathsf{n}, \Psi, \{\mathsf{r}_\mathsf{s} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n}\}, \{\tilde{\mathsf{g}}_{\mathsf{s}\to\mathsf{f}} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n} \wedge \mathsf{f} \in \mathcal{F}_\mathsf{n}(\mathsf{s})\})$ 为公共参数, 其中: $\mathcal{D}$ 和 $\mathcal{M}$ 只需获取 $\mathsf{params} := (\Lambda, \mathsf{n}, \Psi, \{\mathsf{r}_\mathsf{s} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n}\})$, 而 $\mathcal{B}$ 需要获取 $\mathsf{params}' := (\Lambda, \mathsf{n}, \Psi, \{\mathsf{r}_\mathsf{s} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n}\}, \{\tilde{\mathsf{g}}_{\mathsf{s}\to\mathsf{f}} \mid \mathsf{s} \in \mathcal{S}_\mathsf{n} \wedge \mathsf{f} \in \mathcal{F}_\mathsf{n}(\mathsf{s})\})$. 在获得 params 后, $\mathcal{D}$ 调用 Data_Seal 将其封装并存储封装结果 $\mathsf{blob}_{\mathsf{params}}$.

KeyGen. 本阶段为在线银行系统 $\mathcal{B}$ 和收款者设备 $\mathcal{M}$ 初始化公私钥对.

(1) $\mathcal{B}$ **的密钥生成.** 首先, 给定公共参数 params′ 作为输入, $\mathcal{B}$ 选取 $\mathsf{x}, \mathsf{y} \xleftarrow{\$} \mathbb{Z}_\mathsf{p}^*$, 计算 $\mathsf{X} := \mathsf{g}_2^\mathsf{x}$ 和 $\mathsf{Y} := \mathsf{g}_2^\mathsf{y}$. $\mathcal{B}$ 设置 (x, y) 作为其私钥 $\overline{\mathsf{sk}}_\mathcal{B}$, 并公开 (X, Y) 作为公钥 $\overline{\mathsf{pk}}_\mathcal{B}$. 其次, $\mathcal{B}$ 使用 $\sum_1$ 中的 KeyGen( ) 算法生成用于与 $\mathcal{D}$ 建立会话的密钥对: $(\mathsf{sk}_\mathcal{B}, \mathsf{pk}_\mathcal{B}) \leftarrow \mathsf{KeyGen}(1^\lambda)$, 其中 $\mathsf{sk}_\mathcal{B}$ 为私钥.

(2) $\mathcal{M}$ **的密钥生成.** 类似地, $\mathcal{M}$ 使用 KeyGen( ) 算法生成用于与 $\mathcal{D}$ 建立会话的密钥对: $(\mathsf{sk}_\mathcal{M}, \mathsf{pk}_\mathcal{M}) \leftarrow \mathsf{KeyGen}(1^\lambda)$.

相应地, $\mathcal{D}$ 通过分别验证公钥证书的方式从 $\mathcal{B}$ 和 $\mathcal{M}$ 获取正确的公钥 $\mathsf{sk}_\mathcal{B}$ 和 $\mathsf{pk}_\mathcal{B}$. 同样地, $\mathcal{M}$ 和 $\mathcal{B}$ 可以分别获得正确的公钥 $\overline{\mathsf{pk}}_\mathcal{B}$ 和 $\mathsf{pk}_\mathcal{M}$, $\mathcal{T}$ 获得公钥 $\overline{\mathsf{pk}}_\mathcal{B}$. 应用 PKI 解决方案, CA 分别为 $\mathcal{B}$ 和 $\mathcal{M}$ 的公钥签发公钥证书.

Withdraw. 在本阶段中, 付款者使用移动设备 $\mathcal{D}$ 可以从 $\mathcal{B}$ 提取所需数量的数字货币, 提取的每一枚数字货币都可以被分割消费, 提取步骤如下.

(1) 用户操作 $\mathcal{D}$ 的 NW 钱包应用程序准备进行数字货币提取. $\mathcal{D}$ 切换至 SW, 选取一个随机数 $\mathsf{n} \xleftarrow{\$} \{0,1\}^\lambda$. $\mathsf{n}_\mathcal{D}$ 被保存在安全内存中并被传递给 OAPM 代理, OAPM 代理将 $\mathsf{n}_\mathcal{D}$, $\mathcal{D}$ 的公钥 dpk 和证书 $\mathsf{cert}_\mathcal{D}$ 一同发送给 $\mathcal{B}$.

(2) $\mathcal{B}$ 通过验证 $\mathsf{cert}_\mathcal{D}$ 检查 dpk 是否有效, 并检查 $\mathsf{cert}_\mathcal{D}$ 上的配置信息. 如果检查都通过, $\mathcal{B}$ 选取一个随机数 $\mathsf{n}_\mathcal{B} \xleftarrow{\$} \{0,1\}^\lambda$, 一个用于 MAC 的密钥 $\mathsf{k}_{\mathsf{mac}} \xleftarrow{\$} \{0,1\}^\lambda$ 以及一个用于 $\mathsf{Enc}_{\mathsf{sym}}$ 和 $\mathsf{Des}_{\mathsf{sym}}$ 的密钥 $\mathsf{k}_{\mathsf{enc}} \xleftarrow{\$} \{0,1\}^\lambda$. 之后, $\mathcal{B}$ 使用 dpk 加密 $\mathsf{n}_\mathcal{B}$, $\mathsf{k}_{\mathsf{mac}}$ 和 $\mathsf{k}_{\mathsf{enc}}$, 获得密文: $\mathsf{C}_\mathcal{B} \leftarrow \mathsf{Enc}_{\mathsf{asym}}(\mathsf{dpk}, \mathsf{n}_\mathcal{B} \| \mathsf{k}_{\mathsf{mac}} \| \mathsf{k}_{\mathsf{enc}})$. 随后, $\mathcal{B}$ 使用 $\mathsf{sk}_\mathcal{B}$ 对 dpk, $\mathsf{n}_\mathcal{D}$ 和 $\mathsf{C}_\mathcal{B}$ 进行签名, 输出签名结果: $\alpha \leftarrow \mathsf{Sign}(\mathsf{sk}_\mathcal{B}, \mathsf{dpk} \| \mathsf{n}_\mathcal{D} \| \mathsf{C}_\mathcal{B})$. 最后, $\mathcal{B}$ 将承诺请求 $\mathsf{comm}_{\mathsf{req}} := (\mathsf{C}_\mathsf{B}, \alpha)$ 发送给 $\mathcal{D}$.

(3) $\mathcal{D}$ 的 OAPM 代理将收到的 $\mathsf{comm}_{\mathsf{req}}$ 作为输入参数调用 SW 的 OAPM 可信服务. $\mathcal{D}$ 切换至 SW, 钱包应用可信单元等待付款者输入其银行账户名 $\mathsf{account}_\mathcal{D}$, 相对应的口令 pwd 以及打算提取的数字货币数量. 为了描述的简洁性, 这里只介绍如何提取一枚最大面额的数字货币的过程, **Withdraw** 阶段可以很容易地被

扩展为支持一次性提取多枚数字货币. 需说明的一点是, 目前协议仅支持在一次提取过程中最少提取一枚最大面额的完整数字货币, 可以在系统初始化时将最大面额适当设定小一些, 以此能够实现一次性灵活提取较少价值的数字货币的目的. 当付款者完成输入, 逻辑引擎调用 API OAPM_SW_Withdraw() 生成承诺响应:

$$\mathsf{comm}_{\mathsf{res}} \leftarrow \mathsf{OAPM\_SW\_Withdraw}(\mathsf{blob}_{\mathsf{params}}, \mathsf{n}_{\mathcal{D}}, \mathsf{pk}_{\mathcal{B}}, \mathsf{comm}_{\mathsf{req}}, \mathsf{account}_{\mathcal{D}}, \mathsf{pwd}) \tag{14}$$

此 API 的具体执行如下:

1)调用 Data_Unseal() 解封装数据块 $\mathsf{blob}_{\mathsf{params}}$, 提取 params;

2)使用 $\mathsf{pk}_{\mathcal{B}}$ 验证 α: $\mathsf{res} \leftarrow \mathsf{Verify}(\mathsf{pk}_{\mathcal{B}}, \mathsf{dpk} \| \mathsf{n}_{\mathcal{D}} \| \mathsf{C}_{\mathcal{B}}, \alpha)$, 如果 $\mathsf{res} = \mathsf{false}$, 赋值 $\mathsf{comm}_{\mathsf{res}} := \perp$ 并返回结果;

3)使用 dsk 解密 $\mathsf{C}_{\mathcal{B}}$: $(\mathsf{n}'_{\mathcal{B}}, \mathsf{k}_{\mathsf{mac}}, \mathsf{k}_{\mathsf{enc}}) \leftarrow \mathsf{Dec}_{\mathsf{asym}}(\mathsf{dsk}, \mathsf{C}_{\mathcal{B}})$;

4)选取 $\mathsf{m} \xleftarrow{\$} \mathbb{Z}_{\mathsf{p}}^*$ 作为一枚数字货币的密钥, 计算承诺 $\mathsf{U} := \mathsf{g}_{\mathsf{l}}^{\mathsf{m}}$;

5)设置 $\delta := \mathsf{V}$, δ 表示当前该枚数字货币的余额;

6)设置 CT 为长度是 $2^{n+1}-1$ 比特的比特串, 将所有比特位初始置 1, CT 能够描述当前该枚数字货币的花费状态, 通过 CT 比特串对应位置的 1, 可以重构出由数字货币尚未花费部分组成的树形结构;

7)调用 Data_Seal() 封装数据 m, CT 和 δ, 生成数据块 $\mathsf{blob}_{\mathsf{m}}$ (参见 5.2 小节);

8)选取 $\mathsf{r}_{\mathcal{D}} \xleftarrow{\$} \mathbb{Z}_{\mathsf{p}}^*$, 计算 $\mathsf{R}_{\mathcal{D}} := \mathsf{g}_{\mathsf{l}}^{r_{\mathcal{D}}}$;

9)计算 $\mathsf{c}_{\mathcal{D}} := \mathsf{H}_1(\mathsf{g}_{\mathsf{l}} \| \mathsf{U} \| \mathsf{R}_{\mathcal{D}} \| \mathsf{C}_{\mathcal{B}} \| \alpha \| \mathsf{n}'_{\mathcal{B}})$;

10)计算 $\mathsf{s}_{\mathcal{D}} := \mathsf{r}_{\mathcal{D}} + \mathsf{c}_{\mathcal{D}} \cdot \mathsf{m} \pmod{\mathsf{p}}$;

11)生成密文 $\mathsf{C}_{\mathcal{D}} \leftarrow \mathsf{Enc}_{\mathsf{sym}}(\mathsf{k}_{\mathsf{enc}}, \mathsf{account}_{\mathcal{D}} \| \mathsf{pwd})$;

12)生成 $\tau_{\mathcal{D}} \leftarrow \mathsf{MAC}(\mathsf{k}_{\mathsf{mac}}, \mathsf{U} \| \mathsf{n}'_{\mathcal{B}} \| \mathsf{c}_{\mathcal{D}} \| \mathsf{s}_{\mathcal{D}} \| \mathsf{C}_{\mathcal{D}})$, 输出 $\mathsf{comm}_{\mathsf{res}} := (\tau_{\mathcal{D}}, \mathsf{U}, \mathsf{n}'_{\mathcal{B}}, \mathsf{c}_{\mathcal{D}}, \mathsf{s}_{\mathcal{D}}, \mathsf{C}_{\mathcal{D}})$.

OAPM 服务将 $\mathsf{n}'_{\mathcal{B}}$ 和 $\mathsf{k}_{\mathsf{mac}}$ 保存至安全内存, 将 $\mathsf{blob}_{\mathsf{m}}$ 存储在通用非易失性存储器中. 随后, $\mathcal{D}$ 切换回 NW, 将 $\mathsf{comm}_{\mathsf{res}}$ 发送至 $\mathcal{B}$.

(4)输入 $\mathsf{comm}_{\mathsf{res}}$, $\mathcal{B}$ 运行如下算法, 为 $\mathcal{D}$ 生成一枚只有使用对应密钥 m 才能花费的数字货币 σ:

$$(\sigma, \tau_{\mathcal{B}}) \leftarrow \mathsf{Gen\_DC}(\mathsf{comm}_{\mathsf{res}}, \mathsf{params}', \mathsf{k}_{\mathsf{mac}}, \mathsf{k}_{\mathsf{enc}}, \mathsf{n}_{\mathcal{B}}, \overline{\mathsf{sk}}_{\mathcal{B}}) \tag{15}$$

该算法有如下 7 个步骤:

1)验证 $\tau_{\mathcal{D}} = \mathsf{MAC}(\mathsf{k}_{\mathsf{mac}}, \mathsf{U} \| \mathsf{n}'_{\mathcal{B}} \| \mathsf{c}_{\mathcal{D}} \| \mathsf{s}_{\mathcal{D}} \| \mathsf{C}_{\mathcal{D}})$, 验证是否 $\mathsf{n}'_{\mathcal{B}} = \mathsf{n}_{\mathcal{B}}$;

2)查询数据库, 检查 U 是否之前没有被使用过;

3)计算 $\mathsf{R}'_{\mathcal{D}} := \mathsf{g}_{\mathsf{l}}^{\mathsf{s}_{\mathcal{D}}} \cdot \mathsf{U}^{-\mathsf{c}_{\mathcal{D}}}$ 以及 $\mathsf{c}_{\mathcal{D}'} := \mathsf{H}_{\mathsf{l}}(\mathsf{g}_{\mathsf{l}} \| \mathsf{U} \| \mathsf{R}'_{\mathcal{D}} \| \mathsf{C}_{\mathcal{B}} \| \alpha \| \mathsf{n}_{\mathcal{B}})$;

4)验证是否 $\mathsf{c}'_{\mathcal{D}} = \mathsf{c}_{\mathcal{D}}$;

5)使用 $\mathsf{k}_{\mathsf{enc}}$ 解密 $\mathsf{C}_{\mathcal{D}}$, 即 $\mathsf{account}_{\mathcal{D}} \| \mathsf{pwd} \leftarrow \mathsf{Dec}_{\mathsf{sym}}(\mathsf{k}_{\mathsf{enc}}, \mathsf{C}_{\mathcal{D}})$, 随后与相关商业银行配合验证付款者账户名与口令的正确性, 如果付款者账户中的余额充足, 则将账户中与提取数字货币对应数额的钱财扣除并暂存于 $\mathcal{B}$;

6)选取 $\mathsf{a} \xleftarrow{\$} \mathbb{Z}_{\mathsf{p}}^*$, 计算 $\mathsf{A} := \mathsf{g}_{\mathsf{l}}^{\mathsf{a}}$, $\mathsf{B} := \mathsf{A}_{\mathsf{l}}^{\mathsf{y}}$, $\mathsf{C} := \mathsf{g}_{\mathsf{l}}^{\mathsf{ax}} \cdot \mathsf{U}^{\mathsf{axy}}$ 和 $\mathsf{D} := \mathsf{U}^{\mathsf{ay}}$, 生成 $\sigma := (\mathsf{A}, \mathsf{B}, \mathsf{C}, \mathsf{D})$;

7)生成 $\tau_{\mathcal{B}} \leftarrow \mathsf{MAC}(\mathsf{k}_{\mathsf{mac}}, \sigma \| \mathsf{n}_{\mathcal{D}} \| \mathsf{n}_{\mathcal{B}})$.

在上述算法中, 如果任何验证失败, $\mathcal{B}$ 处的数字货币生成过程将立刻被终止;如果均成功通过, $\mathcal{B}$ 将 $(\sigma, \tau_{\mathcal{B}})$ 发送给 $\mathcal{D}$, 并将 $(\mathsf{U}, \mathsf{dpk}, \mathsf{ID}_{\mathsf{bank}}, \mathsf{ID}_{\mathsf{user}})$ 发送给 $\mathcal{T}$ 做备份, 以用于检测未来可能的双花者和揭露非法付款者身份, 其中: $\mathsf{ID}_{\mathsf{bank}}$ 标识了拥有对应用户账户的商业银行身份, $\mathsf{ID}_{\mathsf{user}}$ 标识了对应公钥 dpk 的付款者身份.

(5)收到 $(\sigma, \tau_{\mathcal{B}})$ 后, $\mathcal{D}$ 切换至SW并使用 MAC, $\mathsf{k}_{\mathsf{mac}}$ 和 $\mathsf{n}'_{\mathcal{B}}$ 验证 $\tau_{\mathcal{B}}$. 之后, 数据处理器调用 Data_Seal() 封装σ并生成 blob_{σ}. 最后, 逻辑引擎从安全内存中删除遗留数据 $\mathsf{n}_{\mathcal{D}}$, $\mathsf{n}'_{\mathcal{B}}$ 和 $\mathsf{k}_{\mathsf{mac}}$.

Pre-Compute.在上述阶段执行完毕后, $\mathcal{D}$ 返回 NW, OAPM 代理在后台(离线)执行预计算步骤, 为接下来的 **Spend** 阶段做准备. 预计算 API OAPM_NW_PreCmpt() 由预计算引擎调用以生成盲化的数字货币:

$$(\mathsf{I}, \mathsf{R}, \mathsf{S}, \mathsf{T}, \mathsf{W}) \leftarrow \mathsf{OAPM\_NW\_PreCmpt}(\mathsf{blob}_{\mathsf{params}}, \mathsf{blob}_{\sigma}) \tag{16}$$

该 API 包含以下 4 个步骤:

1)分别从数据块 $\mathsf{blob}_{\mathsf{params}}$ 和 blob_{σ} 的明文部分直接读取 params 和数字货币 σ;

2)将 σ 解析为 (A,B,C,D);

3)选取 $l \xleftarrow{\$} \mathbb{Z}_p^*$, 计算 $(R,S,T,W) := (A^l,B^l,C^l,D^l)$;

4)输入 (l,R,S,T,W).

预计算引擎将 (l,R,S,T,W) 与 blob_{σ} 一同存储于通用非易失性存储器中.

Spend. 这是一个由付款者移动设备 $\mathcal{D}$ 与收款者移动设备 $\mathcal{M}$ 共同执行的双离线交互阶段, 该阶段 $\mathcal{D}$ 能够匿名地向 $\mathcal{M}$ 支付特定数量的数字货币, 双方可以通过蓝牙或 NFC 方式进行通信.

(1) $\mathcal{D}$ 的钱包应用程序向 $\mathcal{M}$ 发送一个随机数 $\overline{n}_{\mathcal{D}} \xleftarrow{\$} \{0,1\}^{\lambda}$ 以初始化交易.

(2) 收到 $\overline{n}'_{\mathcal{D}}$ 后, $\mathcal{M}$ 也选取一个随机 $n_{\mathcal{M}} \xleftarrow{\$} \{0,1\}^{\lambda}$, 生成签名: $\beta \leftarrow \mathrm{Sign}(sk_{\mathcal{M}},\text{"Spend"} \| \mathsf{info})$ 其中: $\mathsf{info} := (v,\mathsf{date},\mathsf{trans},pk_{\mathcal{M}},\overline{n}'_{\mathcal{D}},n_{\mathcal{M}})$. 在这里, info 是一个字符串集, 包含了交易需要支付的数字货币价值量 v, 交易的日期 date, 其它所需的交易信息以及相关的随机数. $\mathcal{M}$ 将 $(\mathsf{info},\mathsf{cert}_{\mathcal{M}},\beta)$ 发送给 $\mathcal{D}$. 事实上, $\mathsf{cert}_{\mathcal{M}}$ 是认证权威(CA)为 $\mathcal{M}$ 签发的证书, 证书内容包含了 $\mathsf{ID}_{\mathcal{M}}$, $pk_{\mathcal{M}}$ 和数字签名 $\mathrm{Sign}_{CA}(\mathsf{ID}_{\mathcal{M}} \| pk_{\mathcal{M}})$, 其中: $\mathsf{ID}_{\mathcal{M}}$ 标识了 $\mathcal{M}$ 的身份.

(3)当 $\mathcal{D}$ 收到上述数据, OAPM 代理组装命令并请求 SW 的 OAPM 可信服务为匿名支付提供运算. 不失一般性地, 这里假设付款者设备 $\mathcal{D}$ 拥有一枚余额为 δ 的数字货币, 并且 $\delta \geq v$. 对于 $\delta < v$ 的情况, $\mathcal{D}$ 可以使用同样方式花费若干枚其它数字货币, 以此使得用于支付的这些数字货币总价值等于 v. 接收到请求后, $\mathcal{D}$ 将状态切换至 SW. 首先, 逻辑引擎使用 Verify, $pk_{\mathcal{M}}$ 连同 $\mathsf{cert}_{\mathcal{M}}$ 一起通过离线方式验证签名 β. 之后, $\mathcal{D}$ 待付款者输入认证所需的 PIN 码(或使用指纹, 人脸或虹膜等生物识别技术)后进入安全的 TUI 模式, 依赖 Framebuffer, TUI 向付款者显示 $\mathsf{ID}_{\mathcal{M}}$ 以及 v, date 和 trans 的具体内容, 付款者需要亲自确认 $\mathsf{ID}_{\mathcal{M}}$ 和交易信息, 以防止敌手恶意伪造或篡改交易内容, 这一步骤是十分重要的. 当付款者点击屏幕的"OK"按钮后, 逻辑引擎调用 API OAPM_SW_Spend() 使用先前的预计算结果创建价值为 v 的数字货币的主序列号 $\boldsymbol{Z}$, 并生成对其有效性的证明 π:

$$(\boldsymbol{Z},\pi) \leftarrow \mathrm{OAPM\_SW\_Spend}(\mathsf{blob}_{\mathsf{params}},\mathsf{blob}_{\mathsf{m}},\mathsf{blob}_{\sigma} \| (l,R,S,T,W),\ \mathsf{info}) \tag{17}$$

该 API 的具体处理流程如下:

1)调用 Data_Unseal() 解封装数据块, 分别获得所需的 params, (m,CT,δ) 以及 (A,B,C,D);

2)验证是否 $\overline{n}'_{\mathcal{D}} = \overline{n}_{\mathcal{D}}$;

3)将 v 以比特分解的方式重新描述, 即 $v = b_n b_{n-1} \ldots b_0$, 设置集合 $\Phi := \{i \mid 0 \leq i \leq n \wedge b_i = 1\}$;

4)从 n 到0, 对于每一个 $i \in \Phi$, 根据 CT 内容, 在树形结构中的 $n-i$ 层均匀随机选取尚未花费的结点 $s_i \in \mathcal{S}_n$, 并在树中将其标记为已花费(即 CT 相关比特位置0);

5)对于每一个被选取的结点 s_i, 计算 $t_{s_i} := g_{s_i}^m$, 形成3个集合 $\mathbf{s} := \{s_i \mid i \in \Phi\}$, $\mathbf{g_s} := \{g_{s_i} \mid i \in \Phi\}$ 以及 $\mathbf{t_s} := \{t_{s_i} \mid i \in \Phi\}$, 令 $\boldsymbol{Z} := (\mathbf{s},\mathbf{t_s})$;

6)选取 $\overline{r} \xleftarrow{\$} \mathbb{Z}_p^*$, 对于每一个 $i \in \Phi$ 计算 $L_i := g_{s_i}^{\overline{r}}$, 形成集合 $\boldsymbol{L} := \{L_i \mid i \in \Phi\}$, 计算 $\overline{L} := B^{l \cdot \overline{r}}$;

7)计算 $\overline{c} := H_2(\mathbf{g_s} \| \mathbf{t_s} \| R \| S \| T \| W \| \boldsymbol{L} \| \overline{L} \| \mathsf{info})$;

8)计算 $\overline{z} := \overline{r} + \overline{c} \cdot m \pmod p$;

9)赋值 $\pi := (R,S,T,W,\overline{c},\overline{z})$;

10)从非易失性存储器中删除 (l,R,S,T,W);

11)更新 CT 和 $\delta := \delta - v$, 如果 $\delta > 0$, 再次调用 Data_Seal() 使用更新后的 CT 和 δ 值重新生成 $\mathsf{blob}_{\mathsf{m}}$, 否则删除相应的 $\mathsf{blob}_{\mathsf{m}}$ 和 blob_{σ}.

在上述 API 返回后, $\mathcal{D}$ 切换回 NW 并将 $(\boldsymbol{Z},\pi)$ 发送至 $\mathcal{M}$.

(4) $\mathcal{M}$ 设置 $\mathsf{Tr} := (\mathsf{info},\boldsymbol{Z},\pi)$, 通过调用专用的验证算法 Tr_Verify 验证 Tr 内容:

$$\mathsf{res} \leftarrow \mathrm{Tr\_Verify}(\mathsf{params},\overline{pk}_B,\mathsf{Tr}) \tag{18}$$

该算法执行以下步骤:

1)将 Tr 解析为 $(\mathsf{info},\boldsymbol{Z} = (\mathbf{s},\mathbf{t_s}),\pi = (R,S,T,W,\overline{c},\overline{z}))$;

2)对于 $\mathbf{s}$ 中的任意两个结点, 均检查它们互相不为以另一个结点为根的子树下面的结点(即任何一个结点都不是另一个结点的前驱结点, 防止重复花费一枚数字货币的同一个部分);

3)对于每一个结点 $s_i \in \mathbf{s}$, 计算 $\overline{L'} := S^{\overline{z}} \cdot W^{-\overline{c}}$, $L'_i := g_{s_i}^{\overline{z}} \cdot t_{s_i}^{-\overline{c}}$, 设置集合 $L' := \{L'_i \mid i \in \Phi\}$;

4)计算 $\overline{c} := H_2(\mathbf{g_s} \| \mathbf{t_s} \| R \| S \| T \| W \| \boldsymbol{L'} \| \overline{L'} \| \text{info})$;

5)验证关系式 $R \neq 1_{\mathbb{G}_1}$, $W \neq 1_{\mathbb{G}_1}$, $e(R,Y) = e(S,g_2)$, $e(T,g_2) = e(R \cdot W, X)$ 以及 $\overline{c} = \overline{c}$ 是否同时成立;

6)如果上述验证全部通过, 则 $\text{res} := \text{true}$, 否则 $\text{res} := \text{false}$.

根据验证结果 res, $\mathcal{M}$ 决定是否接受来自 $\mathcal{D}$ 的支付并向其付款者提供服务或商品. 如果 $\mathcal{M}$ 接受该交易, 它向 $\mathcal{D}$ 发送一个收据 $\theta_{\mathcal{M}} \leftarrow \text{Sign}(\text{sk}_{\mathcal{M}}, \text{"receipt"} \| \text{Tr})$ 以作为已接收数字货币的证据.

*Pre-Compute.*在完成上面的第 3 个步骤后, $\mathcal{D}$ 的 OAPM 代理在 NW 中再次于后台执行预计算, 生成与某个 blob_σ (如果 $\mathcal{D}$ 已不存在可用的 blob_σ, 则不再执行预计算)相关联的新元组 (I', R', S', T', W'), 以备下一次的 **Spend** 阶段使用.

Deposit.在本阶段, $\mathcal{M}$ 通过在线银行系统 $\mathcal{B}$ 将数字货币从 Tr 中提现到收款者的银行账户 $\text{account}_{\mathcal{M}}$ 中.

(1) $\mathcal{M}$ 生成一个数字签名 $\gamma \leftarrow \text{Sign}(\text{sk}_{\mathcal{M}}, \text{"Deposit"} \| \text{Tr} \| \text{account}_{\mathcal{M}})$, 之后将 Tr, $\text{account}_{\mathcal{M}}$ 和 γ 与 $\text{cert}_{\mathcal{M}}$ 一起发送给 $\mathcal{B}$;

(2)首先, $\mathcal{B}$ 使用 $\text{cert}_{\mathcal{M}}$ 和 Verify 验证 γ. 然后, $\mathcal{B}$ 从 info 中提取 $\text{pk}_{\mathcal{M}}$, 并验证该公钥是否与 $\text{cert}_{\mathcal{M}}$ 中指明的公钥相同. 如果相同且无误, $\mathcal{B}$ 计算 $H_3(\text{Tr})$, 通过查询数据库 DB_{Tr} 检查 Tr 之前是否被使用过. 如果没有被使用过, $\mathcal{B}$ 执行验证算法 Tr_Verify() 验证 Tr 的有效性. 如果验证通过, $\mathcal{B}$ 立刻与某商业银行联网协调将价值为 v 的实体货币转入账户 $\text{account}_{\mathcal{M}}$ 中.

(3)在上述步骤完成后, $\mathcal{B}$ 以离线的方式检测可能的双花行为. 具体的检测过程如下:

1)从 Tr 中提取 s 和 $\mathbf{t_s}$, 加载 params';

2)对于每一个 $t_{s_i} \in \mathbf{t_s}$ 和每一个 $f \in \mathcal{F}_n(s_i)$, 计算 $d_{s_i \mapsto f} := e(t_{s_i}, \tilde{g}_{s_i \mapsto f})$ 和 $d_{s_i,f} := H_4(d_{s_i \mapsto f})$;

3)设置集合 $\boldsymbol{d} := \{d_{s_i,f} \mid s_i \in \mathbf{s} \wedge f \in \mathcal{F}_n(s_i)\}$;

4)向数据库 DB_{Tr} 插入表项 $(H_3(\text{Tr}), \text{Tr}, \boldsymbol{d})$;

5)对于每一个 $d_{s_i,f}$, 查询数据库 DB_{Tr} 以检测是否存在一个具有相同 $d_{s_i,f}$ 的交易 Tr', 如果存在则说明可能已出现双花行为, $\mathcal{B}$ 将 Tr 和 Tr' 通过安全信道一同发送给 $\mathcal{T}$ 以捕获双花者的身份.

Capture.本阶段赋予 $\mathcal{T}$ 捕获双花者身份的能力.

(1)在 $\mathcal{T}$ 接收到来自 $\mathcal{B}$ 的双花报告 (Tr, Tr') 后, 其执行验证算法 Tr_Verify() 验证 Tr 和 Tr' 的有效性. 如果它们均有效, $\mathcal{T}$ 从数据 Tr 中选取一个结点 $s_i \in \mathbf{s}$, 并且从自己的密钥集 **sck** 中找到对应的 r_{s_i} 用于恢复 U, 即计算 $U := t_{s_i}^{1/r_{s_i}}$ (也就是 g_i^m). 同样地, $\mathcal{T}$ 从 Tr' 中恢复出 U'.

(2)如果 $U = U'$, 则表明 Tr 和 Tr' 确实造成了一次双花行为, 则 $\mathcal{T}$ 将公开花费者的信息 $(\text{Tr}, \text{Tr}', U, \text{dpk}, \text{ID}_{\text{bank}}, \text{ID}_{\text{user}})$.这之后, 针对付款者 ID_{user} 的一些可能的惩罚措施(例如扣除付款者的部分押金, 暂时停止付款者使用 e-payment 系统或将付款者列入黑名单等)将被触发执行.

Identify.本阶段赋予 $\mathcal{T}$ 揭露非法支付者身份的能力.

$\mathcal{T}$ 从 $\mathcal{B}$ 获取交易报告 Tr 后, 其执行验证算法 Tr_Verify() 验证 Tr 的有效性. 如果有效, $\mathcal{T}$ 从数据 Tr 中选取一个结点 $s_i \in \mathbf{s}$, 并且从自己的密钥集 **sck** 中找到对应的 r_{s_i} 用于恢复 U, 即计算 $U := t_{s_i}^{1/r_{s_i}}$ (也就是 g_i^m). 据此, $\mathcal{T}$ 可揭露非法付款者的信息 $(\text{Tr}, U, \text{dpk}, \text{ID}_{\text{bank}}, \text{ID}_{\text{user}})$.

6 安全性分析

本节将首先给出移动平台双离线匿名支付方案所需要满足的安全属性, 随后分析 OAPM 对于这些安全属性的满足情况.

6.1 安全属性描述

根据文献[14]中的安全定义和证明, 这里给出所提出的移动平台双离线匿名支付方案需要满足的安全属性, 以下使用非形式化的方式以本方案架构的视角对这些属性进行定义.

定义 4 (正确性). 当一个诚实的付款者使用设备 $\mathcal{D}$ 与一个诚实的 $\mathcal{B}$ 共同执行完 **Withdraw** 阶段后, 付款者提取的可分数字货币可以被任何诚实的收款者设备 $\mathcal{M}$ 认可和接受.

定义 5 (不可链接性). 即使 $\mathcal{B}$ 与恶意付款者和收款者合谋, 且 $\mathcal{D}$ 在 **Spend** 阶段生成的支付副本(即 $\mathcal{D}$ 付给 $\mathcal{M}$ 的匿名化数字货币)对应的花费部分(即花费的节点)在一个数字货币树形结构中的位置被泄露, $\mathcal{B}$ 与这些恶意付款者和收款者也无法判断某两个支付副本是否来自同一个付款者(或 $\mathcal{D}$), 本方案的不可链接性涵盖了匿名性.

定义 6 (可追踪性). 如果有恶意的(合谋)付款者花费多于他们提取的数字货币(且试图通过 $\mathcal{B}$ 在 **Deposit** 阶段对货币的验证), 或者他们试图双重花费某个数字货币, 或者他们实施了非法的交易付款行为, 他们的身份将被揭露.

定义 7 (不可陷害性). 即使与恶意付款者和收款者合谋, 且攻破付款者移动设备 $\mathcal{D}$ 的 NW, $\mathcal{B}$ 也无法虚假指控(即陷害)一个诚实的用户双重花费(double-spend)了某个数字货币.

定义 8 (机密性). 即使一个付款者的移动设备 $\mathcal{D}$ 的 NW 被攻破, 其他合谋的恶意付款者和收款者也不能从该用户执行 **Withdraw** 阶段的数据中获取机密信息(如付款者的口令).

定义 9 (真实性 1). 只有某个付款者拥有一台配备 TrustZone 和 SE 的移动设备和一对有效的银行账户名口令, 他才可以向 $\mathcal{B}$ 证实他的合法性.

定义 10 (真实性 2). 只有收款者的身份被某个付款者确认和接受, 收款者才能够将与该付款者的交易所得提现并存储在自己的银行账户中.

6.2 OAPM安全性分析

因为给出形式化的安全证明超出了本文的范围, 这里只给出对OAPM的非形式化安全证明. 总体上讲, 如果 $\mathrm{DDH}_{\mathbb{G}_1}$, Weak-EXDH和B-4-LRSW假设成立, 算法 Σ_1 是EUF-CMA安全的[44], 算法 Σ_2 是uf-cmva安全的[45], 算法 Σ_3 是IND-CPA安全的以及算法 Σ_4 是IND-CPA安全的, 则可以证明OAPM满足上一小节定义的安全属性. 由于密码学算法和方案本身使数据块 $\mathrm{blob}_{\mathrm{params}}$, blob_{σ} 和 $\mathrm{blob}_{\mathrm{m}}$ 能够为敏感数据提供完整性, 并且数据块 $\mathrm{blob}_{\mathrm{m}}$ 能够为 m 提供机密性, 为了分析的简洁, 下文不再涉及对这些数据块的安全分析. 事实上, 这里可以基于文献[12, 38]的安全证明来证明OAPM满足正确性, 不可链接性, 可追踪性和防陷害性的安全属性, 而OAPM的机密性和真实性可以通过标准的方式(Standard Way)给出证明, 本文均不再赘述. 以下通过非形式化的直观描给出OAPM满足上述安全属性的证明.

1)**正确性**

该属性可以通过方案协议所有阶段的成功执行给以检验和证明.

2)**不可链接性**

在**Spend**阶段的协议副本中, 只有 $(\boldsymbol{Z}=(\mathbf{s},\mathbf{t_s}),\pi=(\mathrm{R},\mathrm{S},\mathrm{T},\mathrm{W},\bar{c},\bar{z}))$ 与所花费的数字货币信息 (m,σ) 相关联. 首先, $(\bar{c},\bar{z})$ 是对 m 的非交互零知识证明, 即对于每一个点 $\mathbf{s}_i\in\mathbf{s}$ 及 $\mathrm{W}=\mathrm{S}^{\mathrm{m}}$, 有 $\mathrm{t}_{\mathrm{s}_i}=\mathrm{g}_{\mathrm{s}_i}^{\mathrm{m}}$, 那么 $(\bar{c},\bar{z})$ 不会泄露 m 的任何信息. 其次, $(\mathrm{R},\mathrm{S},\mathrm{T},\mathrm{W})$ 是经过随机化因子 l 随机化处理过的, 且 $\mathrm{S}=\mathrm{R}^{\mathrm{y}}$, $\mathrm{T}=\mathrm{R}^{\mathrm{x}+\mathrm{mxy}}=\mathrm{R}^{\mathrm{x}}\cdot(\mathrm{R}^{\mathrm{m}})^{\mathrm{xy}}$ 以及 $\mathrm{W}=\mathrm{S}^{\mathrm{m}}=(\mathrm{R}^{\mathrm{m}})^{\mathrm{y}}$, 在 $\mathrm{DDH}_{\mathbb{G}_1}$ 假设成立的前提下, 给定 g_1, $\mathrm{U}=\mathrm{g}^{\mathrm{m}}$ 和 R, 判断 $\mathbb{G}_1$ 的一个元素是随机的还是与 R^{m} 相等的是困难的, 因此元组 $(\mathrm{R},\mathrm{S},\mathrm{T},\mathrm{W})$ 不会泄露任何关于 (m,σ) 的有效信息, 实现了匿名化. 此外, 根据文献[12]中的安全证明, 在Weak-EXDH假设下, 将 $\mathrm{g}_{\mathrm{s}}^{\mathrm{m}}$ 与 $\mathbb{G}_1$ 上的一个随机元素区分是困难的, 因此可以进一步得出 $\mathbf{t_s}$ 与随机元素的计算不可区分性. 值得说明的是, σ 必须有 $\mathcal{B}$ 生成, 且 $\mathrm{k}_{\mathrm{mac}}$ 仅有 $\mathcal{B}$ 和 $\mathcal{D}$ 的SW知晓. 综上所述, $\mathbf{s}$ 是仅有的可以泄露所花费数字货币信息的数据, 而这一数据无法帮助敌手链接花费数字货币的付款者(或 $\mathcal{D}$).

3)**可追踪性**

在B-4-LRSW假设下, 没有概率多项式时间内的敌手可以伪造一个数字货币并且不被追溯到**Withdraw** 阶段的协议执行. 由于零知识证明 $(\bar{c},\bar{z})$ 是可靠的, 对于每一个 $\mathbf{s}_i\in\mathbf{s}$, $\mathrm{t}_{\mathrm{s}_i}$ 和 W 必须使用同一个密钥 m 生成. 根据验证关系式 $e(\mathrm{R},\mathrm{Y})=e(\mathrm{S},\mathrm{g}_2)$ 和 $e(\mathrm{T},\mathrm{g}_2)=e(\mathrm{R}\cdot\mathrm{W},\mathrm{X})$, 可知 T 包含了同一个 m, 即 m 是被 $\mathcal{B}$ 所认证的(certified). 因此, 恶意付款者无法花费多于他所提取的数字货币, 并且任何的双花行为都将被 $\mathcal{B}$ 检测出来, 双花者的身份将被 $\mathcal{T}$ 所识别. 类似地, 非法交易的付款者身份也能够被 $\mathcal{T}$ 所揭露.

4)**不可陷害性**

事实上, $(\bar{c},\bar{z})$ 是一个对消息 info 的知识证明的签名, 因此, 如果一个敌手试图想要花费一个诚实付款

者的数字货币, 他必须知道数字货币的密钥 m. 然而, 给定 $\mathsf{U}=\mathsf{g}_1^{\mathsf{m}}$, 相关联的零知识证明 $(\mathsf{c}_{\mathcal{D}},\mathsf{s}_{\mathcal{D}})$ 和 $(\boldsymbol{Z},\pi)$, 在离散对数(DL)假设成立的情况下, 敌手获得 m 是困难的. 所以, 方案的不可陷害性是可以被保证的.

5)**机密性**

因为算法 Σ_1 是EUF-CMA安全的, 并且 $\mathsf{n}_{\mathcal{D}}$ 与其它nonce碰撞的概率可以被忽略, 所以被移动设备 $\mathcal{D}$ 所接受的数据对 $(\mathsf{C}_{\mathcal{B}},\alpha)$ 必须由 $\mathcal{B}$ 所生成. 由于算法 Σ_3 是IND-CPA安全的, 所以密钥 $\mathsf{k}_{\mathsf{mac}}$ 和 $\mathsf{k}_{\mathsf{enc}}$ 仅有 $\mathcal{D}$ 的SW和 $\mathcal{B}$ 知道. 此外, 算法 Σ_2 是uf-cmva安全的, 算法 Σ_4 是IND-CPA安全的, 则付款者的口令 pwd 只能被 $\mathcal{B}$ 所获得. 综上, OAPM满足机密性属性.

6)真实性 1

证书 $\mathsf{cert}_{\mathcal{D}}$ 可以证明一个移动设备拥有合法的设备公钥 dpk, 并配备了ARM TrustZone和SE. 通过挑战响应(Challenge-and-Response)的方式, 移动设备 $\mathcal{D}$ 可以证明拥有私钥 dsk, 并且可以通过使用 $\mathsf{k}_{\mathsf{mac}}$ 生成一个标签 $\tau_{\mathcal{D}}$ 的方式为 U 和 $\mathsf{C}_{\mathcal{D}}=\mathsf{Enc}_{\mathsf{sym}}(\mathsf{k}_{\mathsf{enc}},\mathsf{account}_{\mathcal{D}}\,||\,\mathsf{pwd})$ 提供真实性和完整性保障. 经过上述步骤, 在线银行系统 $\mathcal{B}$ 相信 $\mathcal{D}$ 支持TrustZone并且 U 和 $\mathsf{C}_{\mathcal{D}}$ 是由 $\mathcal{D}$ 所生成的, 通过验证 $(\mathsf{account}_{\mathcal{D}},\mathsf{pwd})$ 的有效性, $\mathcal{B}$ 相信 $\mathcal{D}$ 的付款者确实是账户 $\mathsf{account}_{\mathcal{D}}$ 的所有者. 因此, OAPM满足真实性1.

7)**真实性 2**

因为算法 Σ_1 是 EUF-CMA 安全的, 数字签名 β 必须是由拥有身份 $\mathsf{ID}_{\mathcal{M}}$ 的收款者设备 $\mathcal{M}$ 所生成. 通过 TUI 安全显示 $(\mathsf{ID}_{\mathcal{M}},\mathsf{v},\mathsf{date},\mathsf{trans})$, 付款者可以判断 $\mathcal{M}$ 是否是所期望的交易对象, 而被包含在 info 的公钥 $\mathsf{pk}_{\mathcal{M}}$ 通过 $(\overline{\mathsf{c}},\overline{\mathsf{z}})$ 被签名. 在 **Deposit** 阶段, 数字签名 γ 被生成, 只有知道私钥 $\mathsf{sk}_{\mathcal{M}}$ 的 $\mathcal{M}$ 才可以将 $\mathsf{Tr}=(\mathsf{info},\boldsymbol{Z},\pi)$ 提现. 因此, OAPM 满足真实性 2.

7 实现方法

本节介绍移动平台双离线匿名支付方案 OAPM 的实现方法. 由于方案 OAPM 是基于移动平台 TEE+SE 架构设计的, 方案原型系统的付款者移动设备 $\mathcal{D}$ 和收款者移动设备 $\mathcal{M}$ 搭建在支持 TEE 安全扩展和具备 SE 的移动平台上. 此外, 方案原型系统还在一台 PC 上模拟实现了在线银行系统 $\mathcal{B}$ 和可信权威 $\mathcal{T}$.

在硬件实现方面, 支持 TEE 安全扩展的移动平台使用符合 Linaro96 板标准的 Hikey-960 开发板及其配套系统. 该开发板主芯片是载有 4 个 2.4 GHz ARM Cortex-A73 核心和 4 个 1.8 GHz Cortex-A53 核心的麒麟 960 处理器, 支持 TrustZone 安全扩展技术. 该开发板还配备了 3GB 的 LPDDR4 SDRAM 内存和 32GB 的 UFS 2.0 闪存. 由于 HiKey-960 缺少 NFC 硬件资源, 为了模拟符合 ISO14443 协议的最大传输速率, 原型系统选择使用与 NFC 相同的串行通信接口在移动设备之间传输数据. 此外, 移动平台采用 STM32F103 模块作为安全芯片(也就是 SE). 原型系统在移动设备上使用文献[43]中的方法为临时数据存储实现了安全内存, 为安全显示实现了 Framebuffer. 对于在线银行系统 $\mathcal{B}$ 和可信权威 $\mathcal{T}$ 的仿真实现, 原型系统使用了一台 Dell OptiPlex 5060 台式计算机, 配备有 3.0GHz Intel i5-8500 六核处理器和 16GB 内存, 运行有内核版本为 Linux 4.4.0 的 Ubuntu16.04 操作系统.

在软件实现方面, 原型系统安全世界 SW 内部署了符合 GP TEE 技术标准规范的 OP-TEE 操作系统, 即安全操作系统, 在普通世界 NW 中采用了 Android9.0 操作系统. 为了模拟实现移动平台 OAPM 安全结构, 安全芯片中实现了密钥生成、密钥派生、安全通道建立等函数, OP-TEE 内实现了全部可信服务, 支持 NW 中的应用程序调用离线匿名交易相关接口. 此外, 为了模拟双离线匿名支付协议的全部流程, 原型系统分别在 Android 上实现了钱包应用程序 Wallet, 在 OP-TEE 内实现了钱包应用可信单元 Wallet Trustlet. 使用 GmSSL 3.0 和支持椭圆曲线的 PBC 0.5.14 算法库, 原型系统实现了 OAPM 安全架构中使用的全部密码学算法, 具体算法实现参见表 3. 对于 $\mathsf{H}_1\sim\mathsf{H}_4$ 这 4 个哈希函数(消息摘要算法), 由于安全性的缘故, 需要区分算法实现的不同, 在实现时 4 个函数的基本算法均为 SM3, 它们的输入参数被附加了不同的填充块(Padding \#1～Padding \#4), 以此可以区分它们的不同. 移动设备安全世界 SW 的组件和辅助功能使用了 5268 行 C 语言代码完成实现, 普通世界 NW 的 OAPM 代理组件由 2879 行 C 语言代码组成. 此外, 原型系统还实现了一个测试应用程序, 用于请求调用 OAPM 方案的执行, 该程序包含实现 Wallet 的 1268 行 C 语言代码和 785 行 JAVA 语言代码, 以及实现 Wallet Trustlet 的 661 行 C 语言代码.

为验证方案的通用性和有效性, 上述原型系统中的移动终端部分被实验性地移植到了另外 3 个具有 TEE 和串口通讯能力的移动平台, 分别为配备 2.0GHz ARM Cortex-A72 核心的 Rockchip RK3399 开发板, 配备频率达 2.8 GHz 高通 CPU 的 Robotics RB3 开发板, 以及配备 2.7GHz ARM Cortex-A76 核心的迅海智能

T820 国产化移动终端开发平台. 实验证明, 本文方案能够良好地运行在各类移动平台端, 具有较好的普适性. 原型系统所采用的 Hikey-960 综合性能高于 Rockchip RK3399, 但弱于 Robotics RB3 和迅海智能 T820, 后文方案评估将基于原型系统的 Hikey-960 开展.

表 3 OAPM 密码算法实现

算法函数	密码学实现	安全强度(bits)
H_1	SM3 with Padding \#1	128
H_2	SM3 with Padding \#2	128
H_3	SM3 with Padding \#3	128
H_4	SM3 with Padding \#4	128
$\Sigma_1=(\mathsf{KeyGen},\mathsf{Sign},\mathsf{Verify})$	256-bit SM2	128
$\Sigma_2=(\mathsf{MAC})$	HMAC-SM3	128
$\Sigma_3=(\mathsf{Enc}_{\mathsf{asym}},\mathsf{Des}_{\mathsf{asym}})$	256-bit SM2	128
$\Sigma_4=(\mathsf{Enc}_{\mathsf{sym}},\mathsf{Des}_{\mathsf{sym}})$	128-bit SM4-CBC	128

8 方案评估

本节从理论上分析 OAPM 的运行效率, 并基于原型系统对 OAPM 的性能进行评估和分析.

8.1 效率分析

由于专为匹配 TrustZone 安全扩展技术和 SE 技术而设计, OAPM 实现了一些具有实用价值和实际意义的目标属性. 然而, 方案的定制化构造特点使其很难与其它匿名电子支付解决方案进行对比, 现有解决方案基本都是为 PC 平台而设计的. 因此, 这里只列出 OAPM 的主要效率分析, 理论数据来源于假设方案使用最大面值为 2^n 的数字货币, 各实体效率分析数据见表 4. 表中的分析数据主要关注了相信 $\mathcal{D}$, $\mathcal{B}$ 和 $\mathcal{M}$ 三个实体关键计算步骤的空间与时间复杂度, 这些步骤中的常量级数据规模和低复杂度计算(如 MAC 和 $\mathsf{ENC}_{\mathsf{sym}}$)被忽略. 表中符号含义为: $\mathsf{E}_{\mathbb{G}_1}$ 表示群 $\mathbb{G}_1$ 上的指数运算(椭圆曲线倍点运算); $\mathsf{E}^2_{\mathbb{G}_1}$ 表示群 $\mathbb{G}_1$ 上的双指数运算, 例如计算 $a_1^{b_1}\cdot a_2^{b_2}$; P 表示双线性对运算;密码学函数符号(例如 Verify)表示它们各自对应的计算量; $|\mathbb{G}_1|$ 和 $|\mathbb{G}_2|$ 表示群 $\mathbb{G}_1$ 和 $\mathbb{G}_2$ 上元素所占空间大小; $|\mathsf{p}|$ 表示 $\mathbb{Z}_{\mathsf{p}}^*$ 上随机数所占空间大小. 由于一些计算量与实际花费的数字货币量 n 相关, 表中使用 $|\Phi|(1\le|\Phi|\le\mathsf{n})$ 表示集合 Φ 中的元素个数.

表 4 数字货币最大面值为 2^n 情况下 OAPM 的主要效率数据

数据项	$\mathcal{D}$	$\mathcal{B}$	$\mathcal{M}$
公共参数所占空间	$(2^{n+1}-1)\lvert\mathbb{G}_1\rvert$	$(2^{n+1}-1)\lvert\mathbb{G}_1\rvert+(n+1)2^n\lvert\mathbb{G}_2\rvert$	$(2^{n+1}-1)\lvert\mathbb{G}_1\rvert$
数字货币所占空间	$\lvert\mathsf{p}\rvert+4\lvert\mathbb{G}_1\rvert$	–	–
Withdraw 计算量	$1\mathsf{Verify}+1\mathsf{Dec}_{\mathsf{asym}}+2\mathsf{E}_{\mathbb{G}_1}$	$1\mathsf{Enc}_{\mathsf{asym}}+1\mathsf{Sign}+2\mathsf{E}^2_{\mathbb{G}_1}+3\mathsf{E}_{\mathbb{G}_1}$	–
Pre-Compute 计算量	$4\mathsf{E}_{\mathbb{G}_1}$	–	–
Spend 计算量	$2\mathsf{Verify}+\lvert\Phi\rvert\mathsf{E}_{\mathbb{G}_1}$	–	$2\mathsf{Sign}+(\lvert\Phi\rvert+1)\mathsf{E}^2_{\mathbb{G}_1}+4\mathsf{P}$
双花检测计算量	–	$(\lvert\Phi\rvert+1)\mathsf{E}^2_{\mathbb{G}_1}+(\mathsf{v}+4)\mathsf{P}$	–

从表中 $\mathcal{D}$ 的视角观察, 如果使用 BN 曲线[36]来实现方案, 采用 $\lambda=128$ 和 $\mathsf{n}=10$ 的参数设置, 则存储公共参数的空间需要 66KBytes, 存储一个最大面值数字货币的空间仅需要 160Bytes, 这样的空间占用大小对于现代移动设备来说是毫无压力的. 当 $\mathsf{n}=10$ 时, 允许付款者将一个最大面额的数字货币分割成最多 1024

个子块进行花费. 在相同情况下, 商户 $\mathcal{M}$ 需要等量的空间存储公共参数, 而在线银行系统 $\mathcal{B}$ 需要 721KBytes 的额外空间存储公共参数. 此外, 每一百万个交易数据, $\mathcal{B}$ 需要约 320MBytes～17GBytes 的空间存储 DB_{Tr}, 这对于使用大型服务器集群或数据中心实现的 $\mathcal{B}$ 来说也是完全可以接受的.

在计算复杂度方面, 除了一些验证和解密操作, 付款者移动设备 $\mathcal{D}$ 在 **Withdraw** 和 **Spend** 阶段只需要 $\mathbb{G}_1$ 上的若干指数运算, 尤其在执行最为频繁的在线 **Spend** 阶段, $\mathcal{D}$ 只需要 $|\Phi|$ 个 $\mathbb{G}_1$ 上的指数运算. 收款者移动设备 $\mathcal{M}$ 在 **Spend** 阶段需要执行 $\mathbb{G}_1$ 上的 $(|\Phi|+1)$ 个的双指数运算和 4 个双线性对运算. 虽然这对于移动设备来说运算量过大, 但由于近年移动设备上主芯片(CPU)计算性能大幅提升以及图形计算芯片(GPU)的大规模应用, 复杂运算过程的计算耗时逐渐进入到可接受范围之内. 对于在线银行系统 $\mathcal{B}$ 来说, 复杂度最高的计算出自检测双花的步骤, 也是因为该步骤需要 $v+4$ 个双线性对运算, 然而该步骤是完全离线执行的, 它将不会阻塞正常的在线交易, 而且这些计算对于 $\mathcal{B}$ 的 PC 端计算能力来说是毫无压力的. 综合来看, 在高安全强度下, 对于当前主流的存储和计算能力, 即使这些能力来源于资源受限的移动设备, OAPM 方案在理论上都是可行的.

8.2 性能评估

本小节使用原型系统对 OAPM 的性能进行评估. 因为资源受限的移动设备是方案性能的主要瓶颈, 且方案主要关注的是移动设备端的设计, 所以 OAPM 的评估实验主要围绕着移动设备展开, 实验过程中应用多种不同的参数组合完成了一系列的测试. 参考 ISO/IEC 15946-5 标准[37], 评估实验选取了 1 种能够实现 Type-3 型双线性映射的椭圆曲线:嵌入度为 12 的 BN 曲线[46]. 为了实现 128-bit 的安全强度, 实验采用了 BN256 曲线. 评估实验仿真了整个 OAPM 的执行和处理流程, 涵盖了实例化的方案协议, TrustZone 状态切换和敏感数据管理, 每一项实验结果均为 50 次测试运行结果的平均值.

对于最大面值为 2^{10} 和 2^{20} 的数字货币, 分别花费其中的 287 单位, 图 5 描述了 OAPM 关键步骤的平均时间开销, 这些步骤包括付款者移动设备 $\mathcal{D}$ 的 **Withdraw**, 预计算(*Pre-Compute*)和 **Spend** 步骤, 以及收款者移动设备 $\mathcal{M}$ 的 **Spend** 步骤. 从图中的实验结果可以看出, 受益于预计算的启用, 付款者移动设备 $\mathcal{D}$ 执行最为频繁的 **Spend** 的运算时间开销均小于 275ms. 因为要执行双指数运算和双线性对运算, 收款者移动设备 $\mathcal{M}$ 在 **Spend** 阶段耗时相对较大, 其平均运算时间开销小于 1600ms. 付款者移动设备 $\mathcal{D}$ 不频繁执行的 **Withdraw** 时间开销小于 400ms. 参数 n 的取值对于关键步骤的时间开销并没有明显的影响.

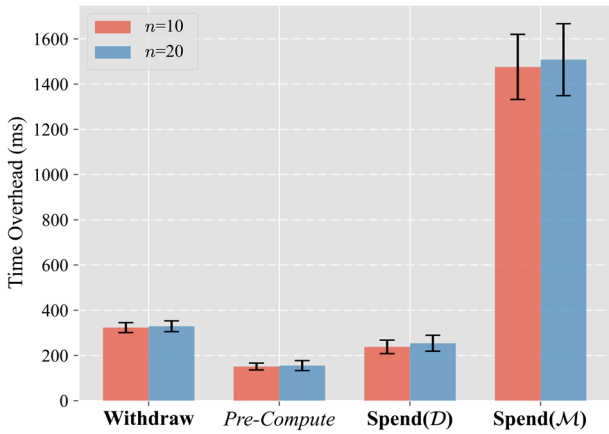

图 5 对于 2^n 的数字 $v=287$ 时关键步骤的时间开销

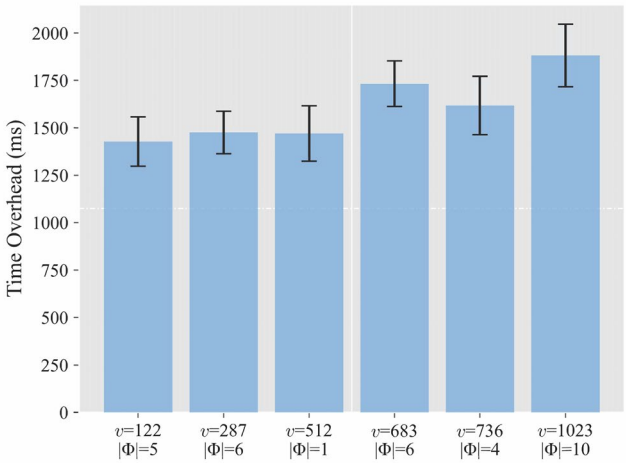

图 6 $n=10$时不同v的取值情况下移动设备

图 6 显示了在 $\mathsf{n}=10$ 并且 $\mathsf{v}\in\{122,287,512,683,736,1023\}$ 时，付款者和收款者移动设备协作执行 **Spend** 阶段的总时间开销. 根据 v 的不同取值，$|\Phi|$ 相应地取值于 v 比特分解后二进制串中 1 的个数. 从图中可以看出，随着 $|\Phi|$ 的增大，**Spend** 阶段的时间开销有着明显的升高，但时间开销与 v 本身取值的大小无关. 值得庆幸的是，在最坏情况下，即当 $|\Phi|=10$ 时，时间开销依然少于 1900ms，这一时延是移动平台交易用户可以接受的.

根据效率分析和实验结果，可以认为 OAPM 对于移动设备平台来说是一个相当高效的方案. 此外，局限于实现时间和设备限制，上述评估的时间效率结果可根据硬件实现平台不同，算法库使用不同，椭圆曲线选择不同以及方案的编程实现优化程度不同取得进一步提升. 一方面，有文献指出使用 BLS12 椭圆曲线可以在实现 128bit 安全强度的情况下达到比 BN256 曲线更快的运算速度;另一方面，本方案评估基于资源受限的嵌入式开发板完成，如果使用目前市场主流的移动终端设备，其具备更高的运算性能，则 OAPM 方案的时间开销将进一步下降.

8.3 对比分析

为解决不同应用场景的电子支付问题，各类方案的设计差异较大，选取前文提及过的与本文目标较为相近的三种双离线支付方案 DOT-M[35]、BS E-Payment[31]和 Divisible E-Cash[12]作为参考对象，与本文方案 OAPM 进行对比.

表 5 OAPM 与相关方案的能力属性比较

能力属性	DOT-M[35]	BS E-Payment[31]	Divisible E-Cash[12]	OAPM
单次支付数币灵活性	**任意数额**	**单次提取数额**	2^i	**任意数额**
预计算	×	×	×	✓
可信移动平台架构	✓	×	×	✓
不可链接性	×	✓	✓	✓
可追踪性	✓	✓	✓	✓
不可陷害性	✓	**依赖平台安全**	**依赖平台安全**	✓
机密性	✓	**依赖平台安全**	**依赖平台安全**	✓

表 5 展示了不同方案的能力属性对比，其中不可链接性、可追踪性、不可陷害性与机密性定义参见 6.1 节. 可以看出，在单次支付数字货币灵活性方面，DOT-M 和 OAPM 均能够根据付款者意愿花费已提取货币的任意数额，而 BS E-Payment 仅能花费之前从银行单次提取的具体数额，这意味着支付行为较为紧密地绑定了数币提取步骤，交易效率较低，对于 Divisible E-Cash 方案，由于其设计的数字货币数据结构形式，支付时数额必须是 2^i (其中 $0\le i\le n$, i 为整数, n 为非负整数)，灵活性受限. 与 OAPM 相比，同样基于 TEE+SE 可信移

动平台设计的 DOT-M 具备可追踪性、不可陷害性和机密性, 但其不支持不可链接性, 没有实现匿名支付的功能. 借助密码学协议专门面向匿名支付设计的 BS E-Payment 和 Divisible E-Cash, 具备不可链接属性, 但两个方案没有深入考量移动应用场景, 其不可陷害性和机密性能否实现取决于部署平台的安全能力. 通过对比, 在双离线支付场景的能力属性方面, 本文方案 OAPM 具有综合优势.

在方案效率方面, 鉴于离线支付数字货币动作是支付场景中最为频繁执行的步骤, 也是用户体验最为敏感的步骤, 表6对比了不同方案在类似**Spend**步骤的主要计算量. 由于DOT-M没有提供匿名支付功能, 仅使用传统签名验证算法, 其计算量主要取决于方案中 n_b 和 n_d 大小, 其中 n_b 表示支付交易中货币数据块的数量, n_d 表示所有数据块过往离线交易的总次数. BS E-Payment的盲签名算法构建在RSA体系之上, 其实际满足有效功能的**Spend**步骤共计13个Galois域上的指数运算, 即 E_{GF}. Divisible E-Cash与本文方案OAPM的协议均构建在双线性群上, Divisible E-Cash未考虑实际应用协议中的交互安全, 没有使用传统签名验证算法. OAPM支持任意数额的货币支付, 其计算量增加了与所支付数字货币数量及结构相关的 $\mathbb{G}_1$ 群上指数和双指数运算. 由于上述各方案算法的实现方式不同, 且计算量与实际花费的数字货币情况有关, 无法直接比较方案的效率高低. 根据8.2节对OAPM的实验测算, 在实现多种能力属性条件下, OAPM的运行效率仍然能够满足实际场景需求.

表 6 OAPM 与相关方案支付步骤的主要计算量比较

协议步骤	DOT-M[35]	BS E-Payment[31]	Divisible E-Cash[12]	OAPM
Spend	$(4+n_b)\mathrm{Sign}+(4+n_b+n_d)\mathrm{Verify}$	$13E_{GF}$	$2E_{\mathbb{G}_1}^2+5E_{\mathbb{G}_1}+4P$	$2\mathrm{Sign}+2\mathrm{Verify}+(\lvert\Phi\rvert+1)E_{\mathbb{G}_1}^2+\lvert\Phi\rvert E_{\mathbb{G}_1}+4P$

9 总结

本文基于 TEE+SE 和可分 e-cash 技术提出了一种完整且高效的移动平台双离线匿名支付方案——OAPM, 方案专注于解决移动支付用户在非联网状态付款时的安全和隐私保护问题. OAPM 允许移动付款者从在线银行系统提取最大面值为 2^n 的数字货币, 并在花费时根据需求将该货币分割成若干块分次付出. 预计算步骤和数字货币比特分解技术被精心地设计和添加入 OAPM, 以提升方案的运行效率和灵活性. 由 SE 提供的信任根以及密钥派生和敏感数据管理机制配合 TrustZone 隔离技术共同增强了方案的安全性, 对于方案的安全性分析进一步确认了 OAPM 所具备的安全属性. 此外, 面向监管需求, OAPM 还具备可控匿名的属性, 可信权威机构能够揭露非法交易付款者的匿名身份. 从 OAPM 的实现, 效率分析和实验评估中可以清楚地看出, OAPM 对于使用资源受限的移动设备的支付用户来说是一套较为实用和高效的双离线匿名交易方案.

本方案的设计遵循了现有金融支付体系架构, 可应用于中央银行数字货币体系, 尤其面向应用需求日益提升的移动支付领域, 方案的支付交易环节在交易双方移动终端完全没有网络连接情况下可正常开展, 实现真实的双离线交易能力, 同时可控匿名能够在保障付款人隐私的情况下满足金融监管的需求, 减少非法交易的隐秘发生.

References:

[1] Kosse A, Mattei I. Making headway: results of the 2022 BIS survey on central bank digital currencies and crypto. No.136: Bank for International Settlements, 2023.

[2] Christodorescu M, Gu W C, Kumaresan R, Minaei M, Ozdayi M, Price B, Raghuraman S, Saad M, Sheffield C, Xu MH, Zamani M. Towards a two-tier hierarchical infrastructure: an offline payment system for central bank digital currencies. arXiv:2012.08003, 2020.

[3] Rial A. Privacy-preserving e-commerce protocols [PhD thesis]. Leuven: University of Leuven, 2013.

[4] Blenner S R, KÖllmer M, Rouse A J, Daneshvar N, Williams C, Andrews L B. Privacy policies of android diabetes apps and sharing of health information. Jama, 2016, 315(2):1051-1052.

[5] Chang K C, Zaeem R N, Barber K S. Is your phone you? how privacy policies of mobile apps allow the use of your personally identifiable information. In: Proc. of the 2020 Second IEEE International Conf. on Trust, Privacy and Security in Intelligent Systems and Applications. Atlanta: IEEE, 2020. 256-262. [doi: 10.1109/TPS-ISA50397.2020.00041]

[6] Yang TX. International comparative analysis of the right to web privacy protection. Journal of Shanghai Business School, 2007, 8(4):41-44. (in Chinese)

[7] Reaves B, Scaife N, Bates A, Traynor P, Butler K R B. Mo(bile) money, mo(bile) problems: analysis of branchless banking applications in the developing world. In: Proc. of the 24th USENIX Security Symposium (USENIX Security 15). Washington DC: USENIX Association, 2015. 17-32.

[8] Prajapat S, Kumar P, Sharma V. An Efficient CL-Signature scheme over NTRU Lattices. In: Proc. of 2022 4th International Conf. on Advances in Computing, Communication Control and Networking (ICAC3N), Greater Noida: IEEE, 2022. 1220-1224. [doi: 10.1109/ICAC3N56670.2022.10074591]

[9] Brickell E, Camenisch J, Chen L. Direct anonymous attestation. In: Proc. of the 11th ACM Conf. on Computer and Communications Security. Washington DC: ACM, 2004. 132-145. [doi: 10.1145/1030083.1030103]

[10] Paquin C. U-prove technology overview V1.1 (Revision 2). Microsoft Corporation, 2013.

[11] Chaum D. Blind signatures for untraceable payments. In: Proc. of Crypto 82. Santa Barbara: Springer, 1983. 199-203. [doi: 10.1007/978-1-4757-0602-4_18]

[12] Canard S, Pointcheval D, Sanders O, Traoré J. Divisible e-cash made practical. In: Proc. of the 18th IACR International Conf. on Practice and Theory in Public-Key Cryptography. Gaithersburg: Springer, 2015. 77-100. [doi: 10.1007/978-3-662-46447-2_4]

[13] Yang B, Feng DG, Qin Y, Zhang QY, Xi L, Zheng CW. Research on direct anonymous attestation scheme based on trusted mobile platform. Journal of Computer Research and Development, 2014, 51(7):1436-1445. (in Chinese)

[14] Wu ZQ, Zhou YW, Qiao ZR. A controllable and trusted anonymous communication scheme. Chinese Journal of Computers, 2010, 33(9):1686-1702. (in Chinese)

[15] Yue XH, Zhou FC, Lin MQ, Li FX. Anonymous attestation scheme with user-controlled-linkability for trusted mobile platform. Chinese Journal of Computers, 2013, 36(7):1434-1447. (in Chinese)

[16] Arfaoui G, Gharout S, Traoré J. Trusted execution environments: A look under the hood. In: Proc. of the 2014 2nd IEEE International Conf. on Mobile Cloud Computing, Services, and Engineering. Oxford: IEEE, 2014. 259-266. [doi: 10.1109/MobileCloud.2014.47.]

[17] Feng DG, Qin Y, Wang D, Chu XB. Research on trusted computing technology. Journal of Computer Research and Development, 2011, 48(8):1332-1349. (in Chinese)

[18] Feng DG, Liu JB, Qin Y, Feng W. Trusted computing theory and technology in innovation-driven development. Scientia Sinica Informationis, 2020, 50(8):1127-1147. (in Chinese)

[19] Vahidi A, Ekdahl P. VETE: virtualizing the trusted execution environment. Swedish Institute of Computer Science, 2016.

[20] Yang B, Feng DG, Qin Y, Zhang YJ: Secure access scheme of cloud services for trusted mobile terminals using trustzone. Ruan Jian Xue Bao/Journal of Software, 2016,27(6):1366-1383 (in Chinese)

[21] Zheng XY, Li W, Meng D. Analysis and research on trustzone technology. Chinese Journal of Computers, 2016, 39(9):1912-1928. (in Chinese)

[22] Zhang YJ, Feng DG, Qin Y, Yang B. A trustzone-based trusted code execution with strong security requirements. Journal of Computer Research and Development, 2015, 52(10):2224-2238. (in Chinese)

[23] Yang F, Zhang QY, Shi ZP, Guan Y. Survey on Software Side-channel Attacks in Trusted Execution Environment. Ruan Jian Xue Bao/Journal of Software, 2023, 34(1): 381-403 (in Chinese)

[24] Miao XL, Chang R, Pan SP, Zhao YW, Jiang LH. Modeling and Security Analysis of Access Control in Trusted Execution Environment. Ruan Jian Xue Bao/Journal of Software, 2023, 34(8): 3637-3658 (in Chinese)

[25] Zhao SJ, Zhang QY, Qin Y, Feng W, Feng DG. SecTEE: A Software-based Approach to Secure Enclave Architecture Using TEE. In: Proc. the 2019 ACM SIGSAC Conference on Computer and Communications Security. London: ACM, 2019. 1723-1740. [doi: 10.1145/3319535.3363205]

[26] Schläpfer T, Rüst, A. Security on IOT devices with secure elements. In: Proc. of the Embedded World Conference. 2019:26-28. [doi: 10.21256/zhaw-3350]

[27] Xu N, Wei W. Design and implementation of trusted platform based on secure chip. Application Research of Computers, 2006, 23(8):117-119. (in Chinese)

[28] Gupta Y K, Jeswani G, Pinto O. M-Commerce offline payment. SN Computer Science, 2022, 3(1): 100.

[29] Li S N, Hu X R, Ling F, Zhang Y, Dong W, Ye J, Sun H L. Research on offline transaction model in mobile payment system. In: Proc. of the 2018 International Conference on Frontier Computing. Kuala Lumpur: Springer, 2018. 1815-1820. [doi: 10.1007/978-981-13-3648-5_235]

[30] Ivanov N, Yan QB. System-wide security for offline payment terminals. In: Proc. of 17th International Conference on Security and Privacy in Communication Networks. Virtual Event: Springer, 2021. 99-119. [doi: 10.1007/978-3-030-90022-9_6]

[31] Kutubi M A A R, Alam K M R, Morimoto Y. A simplified scheme for secure offline electronic payment systems. High-Confidence Computing, 2021, 1(2): 100031.

[32] Batten L, Yi X. Off-line digital cash schemes providing untraceability, anonymity and change. Electronic Commerce Research, 2019, 19(1): 81-110.

[33] EMVCo. EMV mobile payment software-based mobile payment security requirements v1.4. EMVCo, 2020.

[34] Micallef S, Markantonakis K. Mobile payments using Host Card Emulation with NFC: security aspects and limitations. Royal Holloway Information Security thesis series, 2018.

[35] Yang B, Zhang Y, Tong D. DOT-M: a dual offline transaction scheme of central bank digital currency for trusted mobile devices. In: Proc. of 16th International Conference on Network and System Security. Fiji: Springer, 2022. 233-248. [doi: 10.1007/978-3-031-23020-2_13]

[36] Galbraith S D, Paterson K G, Smart N P. Pairings for cryptographers. Discrete Applied Mathematics, 2008, 156(16):3113-3121.

[37] ISO/IEC. 15946-5: 2017 Information technology--security techniques--cryptographic techniques based on elliptic curves: part 5: elliptic curve Generation. ISO/IEC, 2017.

[38] Bernhard D, Fuchsbauer G, Ghadafi E, Smart N P, Warinschi B. Anonymous attestation with user-controlled linkability. International Journal of Information Security, 2013, 12(3):219-249.

[39] Lysyanskaya A, Rivest R L, Sahai A, Wolf S. Pseudonym systems. In: Proc. of the 6th Annual International Workshop on Selected Areas in Cryptography. Kingston: Springer, 1999. 184-199. [doi: 10.1007/3-540-46513-8_14]

[40] GlobalPlatform. TEE internal core api specification v1.2.1. GlobalPlatform, 2019.

[41] GlobalPlatform. TEE client api specification v1.0. GlobalPlatform, 2010.

[42] Trusted Computing Group. TPM main specification version 1.2. Trusted Computing Group, 2011.

[43] Sun H, Sun K, Wang YW, Jing JW. TrustOTP: transforming smartphones into secure one-time password tokens. In: Proc. of the 22nd ACM SIGSAC Conf. on Computer and Communications Security. Denver: ACM, 2015. 976-988. [doi: 10.1145/2810103.2813692]

[44] Goldwasser S, Micali S, Rivest R L. A digital signature scheme secure against adaptive chosen-message attacks. SIAM Journal on Computing, 1988, 17(2):281-308.

[45] Dodis Y, Kiltz E, Pietrzak K, Wichs D. Message authentication, revisited. In: Proc. of the 31st International Conf. on the Theory and Applications of Cryptographic Techniques. Cambridge: Springer, 2012. 355-374. [doi: 10.1007/978-3-642-29011-4_22]

[46] Barreto P S L M, Naehrig M. Pairing-friendly elliptic curves of prime order. In: Proc. of the 12th International Workshop on Selected Areas in Cryptography. Kingston: Springer, 2006. 319-331. [doi: 10.1007/11693383_22]

附中文参考文献:

[6] 杨天翔. 网络隐私权保护:国际比较分析与借鉴. 上海商学院学报, 2007, 8(4):41–44.

[13] 杨波, 冯登国, 秦宇, 张倩颖, 奚璨, 郑昌文. 基于可信移动平台的直接匿名证明方案研究. 计算机研究与发展, 2014, 51(7):1436–1445.

[14] 吴振强, 周彦伟, 乔子芮. 一种可控可信的匿名通信方案. 计算机学报, 2010, 33(9):1686–1702.

[15] 岳笑含, 周福才, 林慕清, 李福祥. 面向可信移动平台具有用户可控关联性的匿名证明方案. 计算机学报, 2013, 36(7):1434–1447.

[17] 冯登国, 秦宇, 汪丹, 初晓博. 可信计算技术研究. 计算机研究与发展, 2011, 48(8):1332–1349.

[18] 冯登国, 刘敬彬, 秦宇, 冯伟. 创新发展中的可信计算理论与技术. 中国科学: 信息科学, 2020, 50(8):1127–1147.

[20] 杨波, 冯登国, 秦宇, 张英骏. 基于 Trustzone 的可信移动终端云服务安全接入方案. 软件学报, 2016, 27(6):1366–1383.
[21] 郑显义, 李文, 孟丹. Trustzone 技术的分析与研究. 计算机学报, 2016, 39(9):1912–1928.
[22] 张英骏, 冯登国, 秦宇, 杨波. 基于 Trustzone 的强安全需求环境下可信代码执行方案. 计算机研究与发展, 2015, 52(10):2224–2238.
[23] 杨帆, 张倩颖, 施智平, 关永. 可信执行环境软件侧信道攻击研究综述. 软件学报, 2023, 34(1):381–403.
[24] 苗新亮, 常瑞, 潘少平, 赵永望, 蒋烈辉. 可信执行环境访问控制建模与安全性分析. 软件学报, 2023, 34(8):3637–3658.
[27] 徐娜, 韦卫. 基于安全芯片的可信平台设计与实现. 计算机应用研究, 2006, 23(8):117–119.