软件学报 ISSN 1000-9825, CODEN RUXUEW
2024,35(12):5621−5635 [doi: 10.13328/j.cnki.jos.007060] [CSTR: 32375.14.jos.007060]

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

面向 Web3 的 SM2 私钥保护方案*

张福泰[1,2], 张 杰[3]

[1](福建师范大学 计算机与网络空间安全学院, 福建 福州 350117)

[2](福建省网络安全与密码技术重点实验室 (福建师范大学), 福建 福州 350117)

[3](西交利物浦大学 智能工程学院, 江苏 苏州 215123)

通信作者: 张杰, E-mail: jie.zhang01@xjtlu.edu.cn

摘 要: 为解决用户私钥安全问题, 将秘密共享方法与边缘计算模式相结合, 提出一种面向用户的、实用的私钥保护框架, 并基于此框架设计针对国密 SM2 公钥密码的私钥保护方案, 将用户的 SM2 私钥通过秘密共享分成两个私钥份额, 分别由用户设备和边缘服务器持有. 当用户使用 Web3 应用服务需要执行公钥密码算法时, 用户设备和边缘服务器利用各自的私钥份额协同执行两方分布式 SM2 公钥密码算法, 在无需恢复原始私钥的情况下完成密码运算. 当用户设备或边缘服务器之一遭到攻击后, 用户通过份额更新算法更新私钥份额, 从而使存在泄漏风险的份额失效. 实验测试结果表明, 新方案的计算时长在现实环境中常用设备 (手机、笔记本电脑) 可接受的范围内.

关键词: 泄漏容忍; 私钥保护; 密钥管理; SM2; 边缘计算

中图法分类号: TP309

中文引用格式: 张福泰, 张杰. 面向Web3的SM2私钥保护方案. 软件学报, 2024, 35(12): 5621−5635. http://www.jos.org.cn/1000-9825/7060.htm

英文引用格式: Zhang FT, Zhang J. Private Key Protected SM2 Scheme for Web3. Ruan Jian Xue Bao/Journal of Software, 2024, 35(12): 5621−5635 (in Chinese). http://www.jos.org.cn/1000-9825/7060.htm

Private Key Protected SM2 Scheme for Web3

ZHANG Fu-Tai[1,2], ZHANG Jie[3]

[1](School of Computer and Cyber Security, Fujian Normal University, Fuzhou 350117, China)

[2](Fujian Provincial Key Lab of Network Security & Cryptology (Fujian Normal University), Fuzhou 350117, China)

[3](School of Advanced Technology, Xi'an Jiaotong Liverpool University, Suzhou 215123, China)

Abstract: To solve the problems of users' private key security, this study proposes a user-oriented and practical private key protection framework by combining secret sharing and edge computing mode. Based on this framework, it designs a private key protection scheme for the SM2 public-key cryptographic system. In this scheme, a user's SM2 private key is divided into two shares via a secret sharing scheme and kept by the user's device and the edge server respectively. The public-key cryptographic task requested by Web3 applications is executed cooperatively by the user's device and the edge server without having to recover the original private key. After the user's device or the edge server is attacked, a key updating protocol will be executed among them to update the private key shares and scrap the one that may have been leaked. Experiment results show that the computing time of the new scheme is acceptable for common devices (smartphones, laptops, etc.) in the real world.

Key words: leakage-resilient; private key protection; key management; SM2; edge computing

下一代互联网 Web3[1]强调用户拥有自己数据的自主权, 其主要是通过公钥密码方案实现. 例如, 在最成熟的 Web3 应用, 即比特币、数字人民币等数字货币领域, 数字货币的所有权是由根据用户公钥生成的地址表示的, 所

* 基金项目: 国家自然科学基金 (62002296, 62172096); 西交利物浦大学研究发展基金 (RDF-21-02-014)
收稿时间: 2023-05-22; 修改时间: 2023-07-19; 采用时间: 2023-09-21; jos 在线出版时间: 2024-01-10
CNKI 网络首发时间: 2024-01-18

有的交易都需要使用用户私钥进行签名. 类似地, 在其他一些 Web3 应用, 如非同质化通证 (non-fungible token, NFT)[2]和去中心化金融 (decentralized finance, DeFi)[3]中, 用户的数据主权也是由公钥密码方案实现的.

因此, 私钥的安全性在 Web3 的去中心化应用 (decentralized application, Dapp) 中至关重要. 自公钥密码学诞生, 对私钥保护的研究就未曾停止过. 其中, 比较经典的方法是将私钥通过秘密共享方案[4]分成若干份额. 基于该方法的方案包括门限秘密分享方案[5−8]和抗泄漏公钥密码方案[9−11]. Web3 中则是采用专用的密钥管理软件, 即加密钱包, 来管理用户私钥. 目前已有大量加密钱包在 Web3 市场上发行.

尽管已经有大量关于私钥保护的研究成果, 现实世界中私钥泄漏问题依然频发. 例如, 2022 年 6 月, Harmony 公链与以太坊间跨链桥 Horizon 遭到黑客攻击, 导致私钥泄漏, 损失加密货币金额超过 1 亿美元. 2022 年 9 月, 世界著名加密做市商 Wintermute 遭到黑客攻击, 因私钥泄漏造成高达 1.6 亿美元天价损失. 此外, 应用程序权限滥用也使私钥的机密性受到威胁. 2022 年 8 月, 总部位于新加坡的信息安全公司 CloudSEK 披露的一份报告中显示, 已发现有 3 207 个移动应用程序泄漏用户密钥和秘密, 其中的 230 个应用程序可在此基础上完全接管用户的推特账号.

此外, 现有的私钥管理方案中, 一旦私钥泄漏或面临泄漏威胁, 公钥私钥需要同时注销. 而在 Web3 场景中, 由于公钥代表用户的钱包地址或数字身份, 经常性的注销和更新公钥会给应用系统造成不便. 例如, 在数字货币系统中, 注销公私钥对之前需要将当前地址中的余额转账到新的地址中, 通常需要支付一定的转账费. 更严重的是, 在用户发现攻击并转移余额前, 攻击者往往已经利用泄漏的私钥将余额转入其所控制的地址中.

为解决上述两个问题, 本文提出具有私钥泄漏容忍性的私钥保护框架, 并基于此框架设计了针对国密 SM2 公钥密码的私钥保护方案, 即私钥泄漏容忍 SM2. 该方案利用基于秘密共享的私钥保护方法, 并结合边缘计算模型, 通过具体 4 个阶段对私钥生成、注册、使用中的安全提供保护: 1) 密钥注册阶段, 用户生成公私钥并向边缘服务器注册公钥, 边缘服务器在验证用户身份后, 存储用户身份和公钥; 2) 安全信道建立阶段, 通过认证密钥协商协议为用户和边缘服务器建立安全信道; 3) 密钥分裂存储阶段, 通过密钥分裂算法将用户 SM2 密钥分成两个份额, 分别由用户设备和边缘服务器持有, 并在其中一个份额面临泄漏风险时, 通过密钥存储更新算法更新两个份额; 4) 分布式算法阶段, 用户设备和边缘服务器使用各自的私钥份额协作完成数字签名、公钥解密. 本文还对所提出的新方案中的两方分布式 SM2 数字签名和公钥解密协议的安全性进行了证明, 并实现了方案的原型. 最后, 本文以智能手机模拟用户设备、笔记本电脑模拟 边缘服务器, 运行方案原型并对其性能进行了理论估算和实验测试. 实验测试结果表明, 虽然本文提出的私钥泄漏容忍 SM2 数字签名和公钥解密的计算时间都在现实环境中常用设备 (手机、笔记本电脑) 可接受范围内.

1 相关工作

本节首先对 Web3 中现有的私钥管理方案进行总结, 然后回顾秘密共享在私钥保护方面的相关工作.

1.1 Web3 中的私钥管理方案

Web3 中专门用来存储和管理用户公私钥对的系统被称为加密钱包. 尽管其名字中有“钱包”, 但并不像普通钱包一样存储货币, 而是存储用户的公私钥并提供用户与 Web3 应用程序交互的接口.

最著名的加密钱包是用于管理用户比特币账户的 Bitcoin Core 钱包软件. 它通过一个随机主密钥和 AES-256-CBC 方案加密用户的私钥, 该主密钥则通过由用户口令生成的密钥和 AES-256-CBC 方案加密. 另一个知名的加密钱包 MetaMask 是用来管理 Ethereum 用户账号、并建立用户与 Ethereum 区块链的连接. 它通过助记词 (secret backup phrase) 来生成用户的私钥, 并通过由用户口令生成的密钥加密助记词和账号数据. 此外, 还有支持多种不同加密货币的加密钱包, 如 Cryptonator、Jaxx, 以及硬件加密钱包 KeepKey、OneKey.

除了上述在客户端管理用户私钥的加密钱包外, Web3 中也有一些使用可信服务器管理用户私钥的方案, 如 Coinbase、GateHub 等, 用户并不完全控制他们的私钥, 服务器负责存储和管理用户私钥, 并通过较强的安全方案来防止对私钥未经授权的访问. 相比于加密钱包, 此类方案具有更好的可用性, 但安全性较低.

1.2 基于秘密共享的私钥保护

秘密共享在私钥保护方面的一个经典应用是门限公钥密码系统. 在 (t, n) 门限公钥密码系统中, 系统的私钥通过

秘密共享方案分成 n 个份额并交给 n 个参与者持有, 通过其中不少于阈值 t 个份额可恢复出私钥. Boneh 等人[5]提出一个用于保护 RSA 私钥的方案, 将 RSA 私钥通过加法秘密共享分成两个份额, 分别由用户和半可信服务器持有. 攻击者若想泄漏 RSA 私钥, 必须同时拿到这两个份额. Lindell 在文献 [6,7] 中提出了两方 ECDSA 方案, 该方案中两个参与者通过密钥生成协议各自生成 ECDSA 私钥的两个份额, 而 ECDSA 私钥无需由任何一方计算出来. 随后, 该方案又被 Lindell 扩展到多方 ECDSA 场景中[8]. 尽管一些门限公钥密码系统中使用秘密共享的出发点是弱化中心的权利而非保护私钥, 但通过秘密共享构造的门限公钥密码方案能够有效解决用户或系统的私钥安全问题.

另一个典型的使用秘密共享保护私钥的例子是抗泄漏公钥密码. 在抗泄漏公钥密码方案中[9−11], 为了抵抗计时攻击、能量分析等类型的侧信道攻击对私钥机密性的威胁, 通过 (2, 2) 门限秘密共享方案将私钥分成两个私钥状态, 分别存储在物理设备的两个独立泄漏的元件中. 抗泄漏公钥密码领域已有大量优秀的理论成果, 如抗泄漏安全模型[12−14]、抗泄漏数字签名[15−18]、抗泄漏公钥加密[19−21]、抗泄漏认证密钥协商[22−25]、抗泄漏零知识证明[26]、抗泄漏秘密共享[27,28]等. 这些方案在具体实现中, 通常假定用户设备中有多个独立泄漏的计算存储元件, 来存储私钥份额和执行计算任务.

1.3 小 结

尽管 Web3 中已经有许多专门用来管理用户公私钥的钱包软件, 并且这些钱包软件也采用了加密、身份认证等方法来防止对私钥未经授权的访问, 但这些方法不能完全阻止对私钥的攻击, 一旦攻击者越过安全防护, 就能拿到完整的私钥. 秘密共享在私钥保护方面的作用已经在门限公钥密码和抗泄漏公钥密码中得到了印证. 本文将基于秘密共享的私钥保护方法与边缘计算模型相结合, 提出适用于 Web3 的私钥保护方案, 能够同时解决前面提到的 Web3 中私钥安全面临的两个问题.

2 基础知识

本节将简要回顾 SM2 中的数字签名和公钥加密方案, 以及本文中方案所基于的困难问题假设和安全模型. 相关符号说明见表 1.

表 1 系统参数

参数	含义
E	定义在素数阶有限域上的椭圆曲线
G	椭圆曲线E的素数阶子群
P	群G的生成元
n	群G中元素的个数
$[k]P$	椭圆曲线上点P的k倍点
$KDF()$	密钥派生函数
$klen$	密钥长度
$H(), H'()$	哈希函数
$Sign(), Verify()$	数字签名/验证算法
$Enc(), Dec()$	公钥加密/解密算法

2.1 SM2 公钥密码方案

令用户的私钥为 $SK \in [1, n-1]$, 公钥为 $PK = [SK]P$, 私钥由用户自己秘密保存, 公钥向系统中其他用户公开. 我们省略 SM2 标准文档中关于数据类型转换的细节, 将数字签名和公钥加密方案回顾如下.

2.1.1 SM2 数字签名方案

• $Sign(M, SK)$: 用户使用私钥生成签名

– 输入消息 M;

– 计算 $e = H(M)$;

– 生成随机数 $k \in [1, n-1]$, 计算 $(x_1, y_1) = [k]P$, $r = (e + x_1) \bmod n$, $s = ((1+SK)^{-1} \cdot (k - r \cdot SK)) \bmod n$;

– 输出用户对消息 M 的签名 $\sigma = (r, s)$.

• *Verify*(M, σ, PK): 验证方使用公钥验证签名

– 输入消息 M 和签名 $\sigma = (r, s)$;

– 计算:

$$\begin{aligned} &e = H(M),\\ &t = (r + s) \bmod n,\\ &(x_1', y_1') = [s]P + [t]PK,\\ &r' = (e + x_1') \bmod n; \end{aligned}$$

– 检验 $r = r'$ 是否成立, 若成立则输出 1 表示签名有效, 否则输出 0 表示签名无效.

2.1.2 SM2 公钥加密方案

• *Enc*(M, PK): 加密方使用公钥加密消息

– 输入消息 M;

– 生成随机数 $k \in [1, n-1]$, 计算:

$$\begin{aligned} &CT_1 = [k]P = (x_1, y_1),\\ &[k]PK = (x_2, y_2),\\ &t = KDF(x_2 \| y_2, klen),\\ &CT_2 = M \oplus t,\\ &CT_3 = H(x_2 \| M \| y_2); \end{aligned}$$

– 输出密文 $CT = CT_1 \| CT_2 \| CT_3$.

• *Dec*(CT, SK): 用户使用私钥解密消息

– 输入密文 $CT = CT_1 \| CT_2 \| CT_3$

– 计算:

$$\begin{aligned} &[SK]CT_1 = (x_2, y_2),\\ &t = KDF(x_2 \| y_2, klen),\\ &M' = CT_2 \oplus t,\\ &u = H(x_2 \| M' \| y_2); \end{aligned}$$

– 验证 $u = CT_3$ 是否成立, 若成立则输出明文 M', 否则报错并退出.

2.2 困难问题假设

令 G 表示椭圆曲线 E 的素数阶子群, P 表示 G 的一个生成元. 本文中的方案假定以下问题在 G 上是困难的.

定义 1. Elliptic curve discrete logarithm (*ECDL*) 问题. 对于任意的 $X \in G$, 找到满足 $[x]P = X$ 的整数 x, 记为 *ECDL*(P, X).

定义 2. Elliptic curve computational Diffie-Hellman (*ECCDH*) 问题. 对于任意的 $X, Y \in G$ 找到满足 *ECDL*(P, Z)=*ECDL*(P, X)·*ECDL*(P, Y) 的元素 Z, 记为 *ECCDH*(P, X, Y).

定义 3. Elliptic curve decisional Diffie-Hellman (*ECDDH*) 问题. 对于任意的 $X, Y, Z \in G$, 若 Z=*ECCDH*(P, X, Y) 则输出 1, 否则输出 0, 记为 *ECDDH*(P, X, Y, Z).

定义 4. Elliptic curve gap Diffie-Hellman (*ECGDH*) 问题. 给定一个 *ECDDH* 预言器, 对于任意的 $X, Y \in G$ 输出 *ECCDH*(P, X, Y), 记为 *ECGDH*(P, X, Y).

2.3 安全模型

2.3.1 认证密钥协商协议的安全模型

eCK 模型是一种广泛认可的用来证明认证密钥协商协议安全性的模型, 它通过 eCK 实验来模拟现实世界中

的针对认证密钥协商协议的攻击. 令 **A** 表示认证密钥协商协议 π 的攻击者, **C** 表示困难问题的挑战者, eCK 安全实验描述如下.

• 初始化: **C** 首先初始化协议参数和参与者的长期公钥私钥, 然后运行协议获得一定数量的会话.

• 询问阶段 I: **A** 针对任意的会话向 **C** 询问该会话中的消息、会话密钥以及两个参与者的长期私钥和临时私钥, 但不能同时询问其中一方的长期私钥和临时私钥.

• 挑战: **A** 选择一个干净会话 (详见定义 5) 作为挑战会话. **C** 生成随机比特 $b\in\{0, 1\}$, 如果 $b = 1$ 则向 **A** 返回挑战会话的会话密钥, 否则向 **A** 返回一个与会话密钥长度相等的随机字符串.

• 询问阶段 II: **A** 继续询问阶段 I 中允许的询问, 但不能询问挑战会话.

• 猜测: **A** 输出 1 个比特 $b'\in\{0, 1\}$. 如果 $b' = b$ 则 **A** 在实验中获胜.

定义 5. 干净会话. 对于一个会话 $\prod_{A,B}^{sid}$, 若以下都不成立, 则该会话被称为干净会话:

• A 或 B 的长期私钥或临时私钥被询问过;

• $\prod_{A,B}^{sid}$ 的会话密钥被询问过;

• 如果 $\prod_{A,B}^{sid}$ 的配对会话 (详见定义 6) 存在, 其配对会话的会话密钥被询问过;

• 如果 $\prod_{A,B}^{sid}$ 的配对会话不存在, B 的长期私钥被询问过.

定义 6. 配对会话. 认证密钥协商协议 π 的两个会话 $\prod_{A,B}^{i}$ 和 $\prod_{B,A}^{j}$ 若满足以下条件, 则这两个会话被称为配对会话:

• $\prod_{A,B}^{i}$ 和 $\prod_{B,A}^{j}$ 都已经计算出了会话密钥;

• $\prod_{A,B}^{i}$ 发送或收到的所有消息与 $\prod_{B,A}^{j}$ 收到或发送的所有消息一致.

定义 7. eCK 安全. 若认证密钥协商协议 π 的任意概率多项式时间的攻击者 **A** 在 eCK 安全实验中获胜的优势, 即:

$$Adv_{\pi}^{\mathrm{eCK}}(\mathbf{A}) = \left|\Pr[\mathbf{A}\text{ wins}] - \frac{1}{2}\right|$$

都是计算可忽略的, 则称该协议是 eCK 安全的.

2.3.2 数字签名方案的安全模型

数字签名方案 (*Sign*, *Verify*) 在选择消息攻击下的存在不可伪造性 (existential unforgeability against chosen-message attacks, EU-CMA) 通过 EU-CMA 安全实验来模拟现实世界中对数字签名方案的攻击. 令 **A** 表示数字签名方案 ς 的攻击者, **C** 表示困难问题的挑战者, EU-CMA 安全实验描述如下.

• 初始化: **C** 初始化系统参数和公钥私钥 (*PK*, *SK*), 并将公钥 *PK* 发送给 **A**;

• 询问: **A** 任意选择消息 *M*, 向 **C** 询问其签名;

• 伪造: **A** 选择一个未询问过的新消息 *M**伪造签名 σ*. 如果 *Verify*(*M**, σ*, *PK*) 输出 1, 则 **A** 在实验中获胜.

定义 8. EU-CMA 安全. 如果数字签名方案 ς 的任意概率多项式时间的攻击者 **A** 在 EU-CMA 安全实验中获胜的优势, 即:

$$Adv_{\varsigma}^{\text{EU-CMA}}(\mathbf{A}) = \left|\Pr[\mathbf{A}\text{ wins}]\right|$$

都是计算可忽略的, 则方案是 EU-CMA 安全的.

2.3.3 公钥加密方案的安全模型

公钥加密方案 (*Enc*, *Dec*) 在适应性选择密文攻击下的不可区分性 (adaptive indistinguishability under chosen-ciphertext attacks, IND-CCA2) 通过 IND-CCA2 安全实验来模拟现实世界中对公钥加密方案的攻击. 令 **A** 表示公钥加密方案 ϵ 的攻击者, **C** 表示困难问题的挑战者, IND-CCA2 安全实验描述如下.

• 初始化: **C** 初始化系统参数和公私钥 (*PK*, *SK*) 并将公钥 *PK* 发送给 **A**;

• 询问阶段 I: **A** 任意选择密文, 向 **C** 询问其解密;

• 挑战: **A** 适应性选择并输出两个等长的不同消息 M_0, M_1 发送给 **C**. **C** 生成随机数 $b\in\{0, 1\}$并计算挑战密文 $CT^* = Enc(M_b, PK)$ 发送给 **A**;

• 询问阶段 II: **A** 选择除 CT^*外的任意密文, 向 **C** 询问其解密;

• 猜测: **A** 输出 $b'\in\{0, 1\}$, 若 $b' = b$ 则 **A** 在实验中获胜.

定义 9. IND-CCA2 安全. 如果公钥加密方案 ϵ 的任意概率多项式时间的攻击者 **A** 在 IND-CCA2 安全实验中获胜的优势, 即:

$$Adv_{\epsilon}^{\text{IND-CCA2}}(\mathbf{A}) = \left|\Pr[\mathbf{A}\text{ wins}] - \frac{1}{2}\right|$$

都是计算可忽略的, 则该方案是 IND-CCA2 安全的.

3 具有私钥泄漏容忍性的私钥保护框架

本节介绍具有私钥泄漏容忍性的私钥保护方案的系统框架、工作流程和具体的算法设计.

3.1 系统架构

如图 1 所示, 私钥保护系统包括两类参与者.

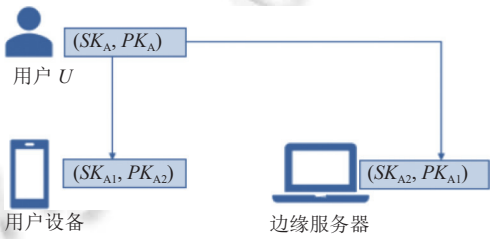

图 1 系统架构

• 用户: 用户拥有一对公私钥 (PK, SK), 其中, 公钥 PK 在系统中公开, 私钥 SK 被分裂成两个私钥份额, 分别由用户设备 U 及其边缘服务器 E 秘密保存. 在私钥分裂后, 用户原始私钥可以离线保存或销毁.

• 边缘服务器: 边缘服务器为用户安全保存私钥的一个份额. 例如, 对于一个组织机构的用户, 其边缘服务器可以是所在组织机构的服务器; 在智能家居场景中, 边缘服务器可以是管理智能家居设备的本地服务器或网关.

3.2 工作流程

私钥保护方案的系统工作流程包括如图 2 所示的 4 个阶段, 其中涉及的主要算法介绍见表 2.

(1) 阶段 I: 密钥注册. 用户通过用户设备生成公私钥 (PK_U, SK_U), 并向通过安全信道 (如线下方式) 向边缘服务器提交公钥和身份信息. 边缘服务器在验证用户身份和公钥后, 存储用户公钥和身份.

(2) 阶段 II: 安全信道建立. 用户设备和边缘服务器通过认证密钥协商协议生成共享认证密钥 K.

(3) 阶段 III: 密钥抗泄漏存储

1) 初始化时, 用户设备通过密钥分裂算法将公私钥分成 (PK_{U1}, SK_{U1}) 和 (PK_{U2}, SK_{U2}) 两对, 并将 (PK_{U1}, SK_{U2}) 通过阶段 II 建立的安全信道发送给边缘服务器. 用户设备安全存储 (PK_{U2}, SK_{U1}), 边缘服务器存储 (PK_{U1}, SK_{U2}), 原始私钥 SK_U 可离线保存或销毁.

2) 当用户设备和边缘服务器之一遭到网络攻击后, 在设备恢复安全运行后, 用户通过用户设备发起密钥存储更新流程, 生成随机数, 并通过阶段 II 建立的安全信道将随机数发送给边缘服务器, 用于更新密钥份额, 从而使可能或已经泄漏的份额失效.

(4) 阶段 IV: 分布式算法. 当用户需要对消息签名或解密时, 通过用户设备和边缘服务器协同执行分布式签名或解密算法, 并在用户设备端将计算结果聚合, 输出签名或明文.

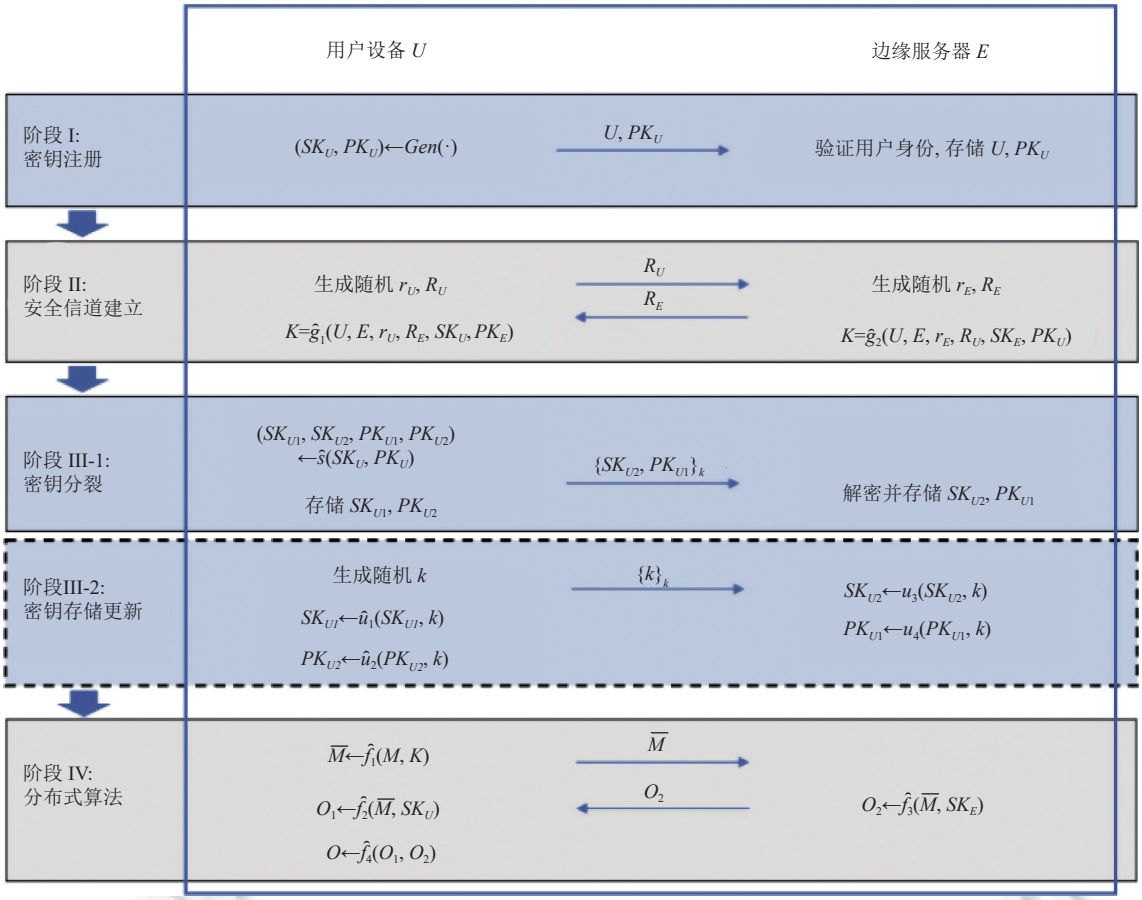

图 2 工作流程

表 2 算法定义

算法	输入	输出	作用
$\hat{g}_1$	r_U, R_E, SK_U, PK_E	K	计算用户设备和其边缘服务器之间的共享密钥
$\hat{g}_2$	r_E, R_U, SK_E, PK_U	K	计算用户设备和其边缘服务器之间的共享密钥
$\hat{s}$	SK_U, PK_U	$SK_{U1}, SK_{U2}, PK_{U1}, PK_{U2}$	将用户密钥进行分裂
$\hat{u}_1$	SK_{U1}, k	新的SK_{U1}	更新SK_{U1}
$\hat{u}_2$	PK_{U2}, k	新的PK_{U2}	更新PK_{U2}
$\hat{u}_3$	SK_{U2}, k	新的SK_{U2}	更新SK_{U2}
$\hat{u}_4$	PK_{U1}, k	新的PK_{U1}	更新PK_{U1}
$\hat{f}_1$	M	$\overline{M}$	对消息进行预处理
$\hat{f}_2$	$\overline{M}, SK_{U1}$	O_1	用户设备端的分布式算法
$\hat{f}_3$	$\overline{M}, SK_{U2}$	O_2	边缘服务器端的分布式算法
$\hat{f}_4$	O_1, O_2	O	将分布式算法的结果聚合

4 私钥泄漏容忍 SM2 公钥密码方案

本节在第 3 节的私钥保护框架基础上, 提出针对 SM2 公钥密码系统的私钥保护方案及其中的分布式数字签

名和分布式公钥解密方案.

4.1 阶段 I: 密钥注册

用户通过调用 SM2 密钥生成算法, 生成一对公私钥 (PK_U, SK_U). 然后通过安全信道 (如线下方式) 向边缘服务器提交公钥 PK_U 和身份 U. 边缘服务器在验证用户身份无误后, 存储 PK_U 和身份 U. 此外, 用户也应以安全方式获得边缘服务器的公钥 PK_E, 该步骤可以在本阶段中实现, 也可以采取一些现有的常用方法 (如公钥基础设施 PKI).

4.2 阶段 II: 安全信道建立

用户设备和边缘服务器之间执行认证密钥协商协议建立安全信道.

• 用户设备生成随机数 $r_U \in [1, n-1]$, 计算 $R_U = [r_U]G$, 并通过公开信道发送 R_U 到边缘服务器;

• 边缘服务器生成随机数 $r_E \in [1, n-1]$, 计算 $R_E = [r_E]G$, 并通过公开信道发送 R_E 到用户设备;

• 用户设备通过以下步骤计算共享密钥:

$$\begin{aligned} K_{U1} &= [r_U]R_E, \\ K_{U2} &= [r_U]PK_E, \\ K_{U3} &= [SK_U]R_E, \\ K_{U4} &= [SK_U]PK_E, \\ K &= H'(U, E, K_{U1}, K_{U2}, K_{U3}, K_{U4}). \end{aligned}$$

• 边缘服务器通过以下步骤计算共享密钥:

$$\begin{aligned} K_{E1} &= [r_E]R_U, \\ K_{E2} &= [SK_E]R_U, \\ K_{E3} &= [r_E]PK_U, \\ K_{E4} &= [SK_E]PK_U, \\ K &= H'(U, E, K_{E1}, K_{E2}, K_{E3}, K_{E4}). \end{aligned}$$

本阶段结束时, 用户设备和边缘服务器建立了共享会话密钥 K.

4.3 阶段 III: 密钥抗泄漏存储

4.3.1 阶段 III-1: 密钥分裂

用户通过用户设备执行私钥分裂算法 $\hat{s}(SK_U, PK_U)$:

• 生成随机数 $r_s \in [1, n-1]$;

• 令 $SK_{U1} = r_s$, 计算 $SK_{U2} = SK_U \cdot r_s^{-1} \bmod n$;

• 计算 $PK_{U1} = [SK_{U1}]G, PK_{U2} = [SK_{U2}]G$;

• 输出 $(SK_{U1}, SK_{U2}, PK_{U1}, PK_{U2})$.

同时, 为了保证分布式签名方案的顺利执行, 用户还对 $SK'_U = (1 + SK_U)^{-1}$ 进行分裂:

• 生成随机数 $r'_s \in [1, n-1]$;

• 令 $SK'_{U1} = r'_s$, 计算 $SK'_{U2} = SK'_U \cdot r'^{-1}_s \bmod n$;

• 输出 SK'_{U1}, SK'_{U2}.

然后, 用户设备安全存储 $SK'_{U1}, SK'_{U2}, PK_{U2}$ 并通过安全信道将 $SK_{U2}, SK'_{U2}, PK_{U1}$ 发送给边缘服务器存储. 最后, 用户可将原始私钥 SK_U 销毁或离线存储.

4.3.2 阶段 III-2: 密钥存储更新

当用户设备或边缘服务器中的一个遭到攻击, 其存储的私钥份额面临泄漏威胁时, 用户通过用户设备发起份额更新流程:

• 用户设备:

– 生成随机数 $k, k' \in [1, n-1]$;

– 计算:

$$SK_{U1}^{\text{update}} = SK_{U1} \cdot k \bmod n,$$
$$SK_{U1}^{'\text{update}} = SK'_{U1} \cdot k' \bmod n,$$
$$PK_{U2}^{\text{update}} = [k^{-1}]PK_{U2} \bmod n.$$

将 k, k' 通过安全信道发送给边缘服务器.

• 边缘服务器计算:

$$SK_{U2}^{\text{update}} = SK_{U2} \cdot k \bmod n,$$
$$SK_{U2}^{'\text{update}} = SK'_{U2} \cdot k' \bmod n,$$
$$PK_{U1}^{\text{update}} = [k^{-1}]PK_{U1} \bmod n.$$

4.4 阶段 IV: 分布式算法

4.4.1 两方分布式 SM2 (2PD-SM2) 数字签名

当用户需要对消息 M 生成签名时, 通过用户设备与边缘服务器协作执行两方分布式 SM2 (2PD-SM2) 数字签名方案.

• 用户设备:

– 输入消息 M;

– 计算 $e = H(M)$;

– 生成随机数 r_{U1} 并计算 $R_{U1} = [r_{U1}]G$;

– 发送 $msg_1 = (e, R_{U1})$ 给边缘服务器.

• 边缘服务器:

– 生成随机数 r_{U2} 并计算 $(x_1, y_1) = R_{U12} = [r_{U2}]R_{U1}$;

– 计算 $r = (e + x_1) \bmod n$;

– 计算 $s_{21} = SK'_{U2} \cdot r_{U2} \bmod n, s_{22} = SK'_{U2} \cdot r \cdot SK_{U2} \bmod n$;

– 发送 $msg_2 = (r, s_{21}, s_{22})$ 给用户设备.

• 用户设备:

– 计算 $s = (s \cdot SK'_{U1} \cdot r_{U1} - s_{22} \cdot SK'_{U1} \cdot SK_{U1}) \bmod n$;

– 运行 $Verify(M, (r, s), PK_U)$, 若结果为 1 则输出用户对 M 的签名 (r, s), 否则返回错误提示.

签名验证算法与 SM2 签名验证相同.

4.4.2 两方分布式 SM2(2PD-SM2) 公钥解密

公钥加密方案与 SM2 公钥加密方案相同; 当用户需要解密由 PK_U 加密得到的密文时, 通过用户设备与边缘服务器协作执行 2PD-SM2 公钥解密方案.

• 用户设备:

– 输入密文 $CT = CT_1 \| CT_2 \| CT_3$;

– 发送 $msg_1 = (CT_1)$ 给边缘服务器.

• 边缘服务器:

– 计算 $M_2 = [SK_{U2}]CT_1$;

– 发送 $msg_2 = (M_2)$ 给用户设备.

• 用户设备:

– 计算 $[SK_{U1}]M_2 = (x_2, y_2)$;

– 计算 $t = KDF(x_2 \| y_2, klen)$;

– 计算 $M' = CT_2 \oplus t$;

– 计算 $u = H(x_2 \| M' \| y_2)$;

– 验证 $u = CT_3$ 是否成立, 若成立则输出明文 M', 否则报错并退出.

5 安全分析

本节对私钥泄漏容忍 SM2 方案中具体协议的安全性进行分析.

5.1 认证密钥协商协议的安全性

令 π 表示私钥泄漏容忍 SM2 中的认证密钥协商协议, 我们首先通过定理 1 声明其 eCK 安全性, 然后给出该定理的详细证明.

定理 1. 假设 H'是随机谕言器, 群 G 上 *ECGDH* 问题困难, 则协议 π 是 eCK 安全的. 具体地, 令 **A** 表示协议 π 的任意多项式时间的攻击者, **C** 表示 G 上 *ECGDH* 问题的挑战者. 假设 **C** 控制 N_p 个参与者并激活 N_s 个会话, **A** 在 eCK 实验中获胜的优势是:

$$Adv_{\pi}^{\mathrm{eCK}}(\mathbf{A}) \leqslant \max\left(\frac{{N_2}^2}{2}, N_p \cdot N_s\right) \cdot Adv_G^{ECGDH}(\mathbf{C}),$$

其中, $Adv_G^{ECGDH}(\mathbf{C})$ 表示挑战者 **C** 解决 G 中 *ECGDH*(P, X, Y) 的优势.

证明: 首先将会话密钥表示为随机预言器 H' 的输出:

$$\begin{aligned} K &= H'(U, E, K_1, K_2, K_3, K_4) \\ &= H'(U, E, ECCDH(R_U, R_E), ECCDH(R_U, PK_E), ECCDH(PK_U, R_E), ECCDH(PK_U, PK_E)) \\ &= H'(\tau_1, \tau_2, \tau_3, \tau_4, \tau_5, \tau_6). \end{aligned}$$

ECGDH 问题的挑战者 **C** 输入 X, $Y \in G$ 并可以访问 *ECDDH* 预言器, 尝试输出 *ECCDH*(P, X, Y), **A** 和 **C** 执行 eCK 安全实验.

• 初始化: **C** 初始化协议的参数 (G, P, Z_n) 和 N_p 个参与者 A, B, C,…的长期公私钥. 然后 **C** 运行密钥协商协议来获取 $N_s - 3$ 个会话. 最后, **C** 预留 3 个特殊的会话在实验过程中根据 **A** 的询问进行构造.

– $\prod_{A,B}^{i}$: 将其中 4 个公钥 (A 的长期和临时公钥, B 的长期和临时公钥) 中的两个设置为 X 和 Y, 其他参数正常生成;

– $\prod_{B,A}^{j}$: $\prod_{A,B}^{i}$ 的配对会话;

– $\prod_{C,D}^{l}$: 将 C 的长短期公钥中的一个设置为 X, D 的长期公钥设置为 Y, 其他值正常生成.

• 询问阶段 I: **A** 向 **C** 询问任意会话的公开信道上传输的消息、会话密钥、4 个私钥中的任意子集 (不能同时包含一方的长短期私钥). 除了 3 个特殊会话外, **C** 诚实回答 **A** 的询问.

– 如果 **A** 询问 $\prod_{A,B}^{i}$ 或 $\prod_{B,A}^{j}$ 中的 A 和 B 的长期私钥, 则 **C** 设置 $R_A = X, R_B = Y$;

– 如果 **A** 询问 $\prod_{A,B}^{i}$ 或 $\prod_{B,A}^{j}$ 中的 A 的长期私钥和 B 的临时私钥, **C** 设置 $R_A = X, PK_B = Y$;

– 如果 **A** 询问 $\prod_{A,B}^{i}$ 或 $\prod_{B,A}^{j}$ 中的 A 的短期私钥和 B 的长期私钥, **C** 设置 $PK_A = X, R_B = Y$;

– 如果 **A** 询问 $\prod_{A,B}^{i}$ 或 $\prod_{B,A}^{j}$ 中的 A 和 B 的短期私钥, 则 **C** 设置 $PK_A = X, PK_B = Y$;

– 如果 **A** 询问 $\prod_{C,D}^{l}$ 中 C 的短期私钥, 则 **C** 设置 $PK_C = X, PK_D = Y$;

– 如果 **A** 询问 $\prod_{C,D}^{l}$ 中 C 的长期私钥, 则 **C** 设置 $R_C = X, PK_D = Y$.

• 挑战: **A** 选择一个干净会话 $\prod_{id}^{*}$ 作为挑战会话. **C** 生成随机比特 $b \in \{0, 1\}$; 如果 $b = 1$ 则向 **A** 返回 $\prod_{id}^{*}$ 的会话密钥, 否则返回一个与会话密钥等长的随机字符串.

• 询问阶段 II: **A** 继续询问阶段 I 中的询问, 但不能询问到 $\prod_{id}^{*}$.

• 猜测: **A** 输出 $b_0 \in \{0, 1\}$. 如果 $b_0 = b_1$ 则 **A** 获胜.

在上述 eCK 安全实验中, 有两种情况.

• 情况 1: $sid^* \in \{i, j\}$ 发生的概率是 $P_1 = 2/N^2$. 在实验结束时, 如果 **A** 获胜, 则 **C** 可以输出 (τ_3, τ_4, τ_5, τ_6) 的一个作为 *ECGDH*(P, X, Y) 的解.

• 情况 2: $sid^* = l$ 发生的概率是 $P_2 = 1/(N_p \cdot N_s)$. 在实验结束时, 如果 **A** 获胜, 则 **C** 同样可以输出 (τ_3, τ_4, τ_5, τ_6)

的一个作为 $ECGDH(P, X, Y)$ 的解.

因此, **C** 解决 G 中 $ECGDH(P, X, Y)$ 的优势:

$$Adv_G^{ECGDH}(\mathbf{C}) \geqslant \max(P_1, P_2) \cdot Adv_\pi^{\mathrm{eCK}}(\mathbf{A}) = \max\left(\frac{2}{N_2^2}, \frac{1}{N_p \cdot N_s}\right) \cdot Adv_\pi^{\mathrm{eCK}}(\mathbf{A}),$$

由于 $ECGDH$ 问题在 G 上是困难的, 因此, 攻击者 **A** 在 eCK 实验中获胜的优势:

$$Adv_\pi^{\mathrm{eCK}}(\mathbf{A}) \leqslant \max\left(\frac{N_2^2}{2}, N_p \cdot N_s\right) \cdot Adv_G^{ECGDH}(\mathbf{C}).$$

是计算可忽略的. 证毕.

5.2 2PD-SM2 数字签名的安全性

我们首先根据第 2.3.2 节中的 EU-CMA 安全实验, 给出 2PD-EU-CMA 安全实验.

• 初始化: **C** 初始化系统参数、公钥私钥 (PK, SK), 并将公钥 PK 发送给 **A**.

• 询问: **A** 任意选择消息 M, 向 **C** 询问签名以及 U 和 E 之间公开信道上的消息.

• 挑战: **A** 选择一个未询问过的新消息 M^*伪造签名 σ^*. 如果 $Verify(M^*, \sigma^*, PK)$ 输出 1, 则 **A** 获胜.

令 ς 表示 2PD-SM2 数字签名协议, 其 2PD-EU-CMA 安全性声明和证明过程如下.

定理 2. 如果 SM2 数字签名方案是 EU-CMA 安全的, 则协议 ς 是 2PD-EU-CMA 安全的.

证明: 令 **A** 为 2PD-SM2 数字签名方案 ς 的攻击者, **C** 为 SM2 数字签名协议的攻击者, **S** 为 EU-CMA 安全实验中的挑战者. **A** 作为攻击者、**C** 作为挑战者执行 2PD-EU-CMA 安全实验, 同时, **C** 作为攻击者和挑战者 **S** 共同执行 EU-CMA 安全实验. 具体过程如下.

• 初始化: **S** 初始化系统参数和公私钥, 并将公钥 PK 发送给 **C**; **C** 收到后将 PK 发送给 **A**.

• 询问: **A** 任意选择消息 M 并向 **C** 询问签名. 当收到询问时, **C** 向 **S** 询问 M 的签名, 并将收到的答复返回给 **A**. 此外, **A** 还可以询问 ς 的会话中公开信道上传输的消息, **C** 将通过以下方式回复 **A**.

– 如果 **A** 询问 msg_1, 则 **C** 按照协议规定生成 e, R_{U1} 并返回给 **A**;

– 如果 **A** 询问 msg_2, 则 **C** 按照协议规定生成 r、生成两个随机字符串作为 s_{21}, s_{22}、并返回给 **A**; 由于 **A** 不知道 $SK_{U1}, SK'_{U1}, r_{U1}$, 因此不能区分收到的 msg_2 中的 s_{21}, s_{22} 是随机生成还是执行协议生成的.

• 挑战: **A** 选择一个未询问过的新消息 M^*并伪造签名 σ^*. 如果 $Verify(M^*, \sigma^*, PK)$ 输出 1, 则 **A** 在实验中获胜.

在上述实验结束时, 如果 **A** 获胜, 则 **C** 可以将 σ^*输出给 **S** 并在 EU-CMA 实验中获胜, 即:

$$\Pr[\mathbf{C} \text{ wins in EU-CMA}] \geqslant \Pr[\mathbf{A} \text{ wins in 2PD-EU-CMA}]$$

由于 SM2 数字签名方案是 EU-CMA 安全的, 因此 Pr[**A** wins in 2PD-EU-CMA] 是计算可忽略的, 即 ς 是 2P-EU-CMA 安全的. 证毕.

5.3 2PD-SM2 公钥解密的安全性

首先根据第 2.3.3 节中描述的 IND-CCA2 安全实验, 给出 2PD-IND-CCA2 安全实验.

• 初始化: **C** 初始化系统参数、公钥私钥 (PK, SK), 并将公钥 PK 发送给 **A**.

• 询问阶段 I: **A** 任意选择密文, 向 **C** 询问其解密以及 U 和 E 之间公开信道上的消息.

• 挑战: **A** 适应性选择并输出两个等长的不同消息 M_0, M_1. **C** 生成随机数 $b \in \{0,1\}$, 计算挑战密文 $CT^* = Enc(M_b, PK)$ 并发送给 **A**.

• 询问阶段 II: **A** 继续与询问阶段 I 相同的方式进行询问, 但不能询问 CT^*的解密.

• 猜测: **A** 输出 $b' \in \{0,1\}$, 若 $b' = b$ 则 **A** 在实验中获胜.

令 ϵ 表示 SM2 两方分布式公钥解密协议, 其 2PD-IND-CCA2 安全性声明和证明过程如下.

定理 3. 如果 SM2 公钥解密方案是 IND-CCA2 安全的, 则协议 ϵ 是 2PD-IND-CCA2 安全的.

证明: 令 **A** 为协议 ϵ 的攻击者, **C** 为 SM2 公钥加密方案的攻击者, **S** 为 IND-CCA2 安全实验中的挑战者. **A** 作为攻击者、**C** 作为挑战者执行 2PD-IND-CCA2 安全实验, 同时, **C** 作为攻击者和挑战者 **S** 共同执行 IND-CCA2 安

全实验. 具体过程如下.

• 初始化: **S** 初始化系统参数和公私钥 (PK, SK), 并将公钥 PK 发送给 **C**; **C** 收到后将 PK 发送给 **A**.

• 询问阶段 I: **A** 任意选择密文 CT 向 **C** 询问其解密; 当 **C** 收到询问后, 向 **S** 询问 CT 的解密并将收到的结果返回给 **A**. 此外, **A** 还可以询问 ϵ 的会话中公开信道上传输的消息, **C** 将通过以下方式回复 **A**.

• 如果 **A** 询问 msg_1, 则 **C** 按照协议规定将 CT_1 返回给 **A**;

• 如果 **A** 询问 msg_2, 则 **C** 生成群 G 中的随机元素并返回给 **A**; 由于 **A** 不知道 SK_{U1}, SK_{U2}, 因此不能区分收到的 msg_2 中的 M_2 是随机生成还是执行协议生成的.

• 挑战: **A** 适应性选择并输出两个等长的不同消息 M_0, M_1; **C** 收到后转发给 **S**; **S** 生成随机数 $b \in \{0,1\}$ 并计算挑战密文 $CT^* = Enc(M_b, PK)$ 发送给 **C**; **C** 收到后转发给 **S**.

• 询问阶段 II: **A** 继续进行与询问阶段 I 相同的询问, 但不能询问 CT^*的解密.

• 猜测: **A** 输出 $b' \in \{0,1\}$, 若 $b' = b$ 则 **A** 在 2PD-IND-CCA2 实验中获胜.

在上述实验结束时, 如果 **A** 获胜, **C** 可以将 b' 输出给 **S** 并在 IND-CCA2 实验中获胜, 即:

$$\Pr[\textbf{C} \text{ wins in IND-CCA2}] \geqslant \Pr[\textbf{A} \text{ wins in 2PD-IND-CCA2}].$$

由于 SM2 公钥加密方案是 IND-CCA2 安全的, 因此 Pr[**A** wins in 2PD-IND-CCA2] 是计算可忽略的, 即 ϵ 是 2PD-IND-CCA2 安全的. 证毕.

6 性　能

本节首先从理论上分析私钥泄漏容忍 SM2 公钥密码方案的计算代价, 然后通过实验测试其在现实世界常用计算设备 (手机和笔记本电脑) 上的具体性能.

6.1 理论分析

在私钥泄漏容忍 SM2 公钥密码方案中用到的所有运算中, 椭圆曲线群 G 上的指数运算 (即求点的倍点运算) 的计算复杂度远远高于其他运算. 因此, 我们通过统计方案中指数运算的个数来评估其计算代价, 具体结果总结在表 3.

表 3　私钥泄漏容忍 SM2 方案中指数运算统计

阶段	用户设备端	边缘服务器端
阶段 I 密钥注册	1	0
阶段 II 安全信道建立	5	5
阶段 III-1 密钥分裂	2	0
阶段 III-2 密钥存储更新	1	1
阶段 IV 2PD-SM2 签名	3	1
阶段 IV 2PD-SM2 解密	1	1

通过表 3 我们可以看出, 私钥泄漏容忍 SM2 中阶段 II 安全信道建立中所用到的群 G 上的指数运算最多, 其计算代价最高. 该阶段中采用的是具有较高的 eCK 可证明安全性的协议, 其攻击模型中假定攻击者有很强的攻击能力, 因而计算代价较高. 在实际场景中, 如果用户设备和边缘服务器所处的网络环境较为安全 (如智能家居场景中家庭场景中的智能设备和路由器之间), 则可以采用较为轻量的方法来建立安全信道.

为了进一步评估私钥泄漏容忍 SM2 对用户设备端带来的计算负担, 我们在表 4 中将阶段 IV 的 2PD-SM2 数字签名和公钥解密与 SM2 数字签名和公钥解密方案中的指数运算个数进行了比较.

通过表 4 我们可以看出, 在用户设备端, 2PD-SM2 数字签名比 SM2 数字签名的计算代价略高, 而 2PD-SM2 公钥解密均与 SM2 公钥密码方案的计算代价相同. 接下来, 我们将通过具体实验来进一步评估 2PD-SM2 对现实场景中常用的用户设备的性能影响.

表 4 用户设备端 2PD-SM2 与 SM2 中指数运算对比

公钥密码算法	SM2	2PD-SM2
数字签名	1	3
公钥解密	1	1

6.2 实 验

6.2.1 实验环境

我们用 Python 3 编程语言在 SM2 标准中推荐的椭圆曲线 SM2-P-256 上实现了私钥泄漏容忍 SM2 方案的原型, 并用一部智能手机和一台笔记本电脑 (详细配置见表 5) 分别模拟用户设备和边缘服务器, 对方案原型进行了实验测试.

表 5 实验硬件环境

参与方	配置
用户设备 (手机)	处理器: 高通骁龙 695 5 GB 8核, 内存: 8 GB, 操作系统: Android 12
边缘服务器 (笔记本电脑)	处理器: 2.3 GHz 双核 Intel Core i5, 内存: 8 GB, 操作系统: macOS 13.1

6.2.2 实验结果

我们运行私钥泄漏容忍 SM2 原型和 SM2 公钥密码方案的原型各 10 次, 统计了具体算法的平均计算时间, 结果总结在表 6 和表 7 中 (在阶段 I 中, 边缘服务器通常需要审核用户的身份, 可能包含人工审核, 因此没有通过计算机来估算计算时间).

表 6 私钥泄漏容忍 SM2 方案计算时间 (s)

阶段	用户设备端	边缘服务器端
阶段 I 密钥注册	0.07526	—
阶段 II 安全信道建立	0.34914	0.16038
阶段 III-1 密钥分裂	0.14186	0.00028
阶段 III-2 密钥存储更新	0.07156	0.03178
阶段 IV 2PD-SM2 签名	0.21337	0.03622
阶段 IV 2PD-SM2 解密	0.06945	0.03341

表 7 用户设备端 2PD-SM2 与 SM2 数字签名和公钥解密计算时间对比 (s)

公钥密码算法	SM2	2PD-SM2
数字签名	0.07653	0.21337
公钥解密	0.08111	0.06945

表 6 中的结果印证了我们在表 3 中理论估算的计算代价. 对于用户手机而言, 最长的计算时间是阶段 II 的 0.35 s, 该结果在普通智能手机设备可接受的范围内.

表 7 中的数据与表 4 中的理论估算结论一致. 在用户设备端, 2PD-SM2 数字签名计算时间约为 SM2 数字签名计算时间的 3 倍, 但具体数据 (0.21 s) 对于普通智能手机而言是可以接受的; 2PD-SM2 公钥解密时间比 SM2 公钥解密时间略低, 这是由于在 2PD-SM2 公钥解密中, 一部分计算任务在边缘服务器执行了.

7 总 结

本文提出了一种面向用户的、实用的私钥保护框架, 并基于此框架提出了针对国密 SM2 公钥密码系统的私钥保护方案, 即私钥泄漏容忍 SM2. 对私钥泄漏容忍 SM2 系统中的认证密钥协商协议、两方分布式 SM2 数字签名和两方分布式 SM2 公钥解密协议的安全性, 本文均通过基于复杂性理论的安全模型进行了详细证明. 此外, 本

文还实现了新方案的原型, 并在现实环境中常用设备手机和笔记本电脑之间运行方案, 测试平均计算时间. 实验结果表明新方案的性能是这些常用设备可以接受的. 随着 Web3 应用的逐步落地和推广, 面向用户的私钥保护方案需求也逐渐增多, 本文的方案及其原型能为 Web3 安全提供理论指导和实施参考. 下一步我们计划研发针对国密 SM9 基于身份的公钥密码系统的私钥保护方案, 为保障我国 Web3 用户的私钥安全提供更多可选方案.

References:

[1] Liu ZT, Xiang YX, Shi J, Gao P, Wang HY, Xiao XS, Wen BH, Li Q, Hu YC. Make Web3 connected. IEEE Trans. on Dependable and Secure Computing, 2022, 19(5): 2965–2981. [doi: 10.1109/TDSC.2021.3079315]

[2] Wang Q, Li RJ, Wang Q, Chen SP. Non-fungible token (NFT): Overview, evaluation, opportunities and challenges. arXiv:2105.07447, 2021.

[3] Li WK, Bu JY, Li XQ, Chen XY. Security analysis of DeFi: Vulnerabilities, attacks and advances. In: Proc. of the 2022 IEEE Int'l Conf. on Blockchain. Espoo: IEEE, 2022. 488–493. [doi: 10.1109/Blockchain55522.2022.00075]

[4] Shamir A. How to share a secret. Communications of the ACM, 1979, 22(11): 612–613. [doi: 10.1145/359168.359176]

[5] Boneh D, Ding XH, Tsudik G, Wong CM. A method for fast revocation of public key certificates and security capabilities. In: Proc. of the 10th Conf. on USENIX Security Symp. Washington: USENIX Association, 2001. 22. [doi: 10.5555/1251327.1251349]

[6] Lindell Y. Fast secure two-party ECDSA signing. In: Proc. of the 37th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2017. 613–644. [doi: 10.1007/978-3-319-63715-0_21]

[7] Lindell Y. Fast secure two-party ECDSA signing. Journal of Cryptology, 2021, 34(4): 44. [doi: 10.1007/s00145-021-09409-9]

[8] Indell Y, Nof A. Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody. In: Proc. of the 2018 ACM SIGSAC Conf. on Computer and Communications Security. Toronto: ACM, 2018. 1837–1854. [doi: 10.1145/3243734.3243788]

[9] Chari S, Jutla CS, Rao JR, Rohatgi P. Towards sound approaches to counteract power-analysis attacks. In: Proc. of the 19th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 1999. 398–412. [doi: 10.1007/3-540-48405-1_26]

[10] Trichina E, Bellezza A. Implementation of elliptic curve cryptography with built-in counter measures against side channel attacks. In: Proc. of the 4th Int'l Conf. on Cryptographic Hardware and Embedded Systems. Redwood Shores: Springer, 2002. 98–113. [doi: 10.1007/3-540-36400-5_9]

[11] Kiltz E, Pietrzak K. Leakage resilient ElGamal encryption. In: Proc. of the 16th Int'l Conf. on the Theory and Application of Cryptology and Information Security. Singapore: Springer, 2010. 595–612. [doi: 10.1007/978-3-642-17373-8_34]

[12] Micali S, Reyzin L. Physically observable cryptography. In: Proc. of the 1st Theory of Cryptography Conf. Cambridge: Springer, 2004. 278–296. [doi: 10.1007/978-3-540-24638-1_16]

[13] Agrawal S, Dodis Y, Vaikuntanathan V, Wichs D. On continual leakage of discrete log representations. In: Proc. of the 19th Int'l Conf. on the Theory and Application of Cryptology and Information Security. Bengaluru: Springer, 2013. 401–420. [doi: 10.1007/978-3-642-42045-0_21]

[14] Alwen J, Dodis Y, Wichs D. Survey: Leakage resilience and the bounded retrieval model. In: Proc. of the 4th Int'l Conf. on Information-theoretic Cryptography. Shizuoka: Springer, 2010. 1–18. [doi: 10.1007/978-3-642-14496-7_1]

[15] Boyle E, Segev G, Wichs D. Fully leakage-resilient signatures. Journal of Cryptology, 2013, 26(3): 513–558. [doi: 10.1007/s00145-012-9136-3]

[16] Faust S, Kiltz E, Pietrzak K, Rothblum GN. Leakage-resilient signatures. In: Proc. of the 7th Theory of Cryptography Conf. Zurich: Springer, 2010. 343–360. [doi: 10.1007/978-3-642-11799-2_21]

[17] Katz J, Vaikuntanathan V. Signature schemes with bounded leakage resilience. In: Proc. of the 15th Int'l Conf. on the Theory and Application of Cryptology and Information Security. Tokyo: Springer, 2009. 703–720. [doi: 10.1007/978-3-642-10366-7_41]

[18] Yu Y, Ding YJ, Zhao YQ, Li YN, Zhao Y, Du XJ, Guizani M. LRCoin: Leakage-resilient cryptocurrency based on bitcoin for data trading in IoT. IEEE Internet of Things Journal, 2019, 6(3): 4702–4710. [doi: 10.1109/JIOT.2018.2878406]

[19] Tang JH, Zhu YQ, Luo XZ. Identity-based encryption scheme against adaptive leakage. Journal on Communications, 2012, 33(7): 90–95 (in Chinese with English abstract). [doi: 10.3969/j.issn.1000-436X.2012.07.011]

[20] Chow SSM, Dodis Y, Rouselakis Y, Waters B. Practical leakage-resilient identity-based encryption from simple assumptions. In: Proc. of the 17th ACM Conf. on Computer and Communications Security. Chicago: ACM, 2010. 152–161. [doi: 10.1145/1866307.1866325]

[21] Li JG, Yu QH, Zhang YC. Identity-based broadcast encryption with continuous leakage resilience. Information Sciences, 2018, 429: 177–193. [doi: 10.1016/j.ins.2017.11.008]

[22] Zhang J, Zhang FT. Identity-based key agreement for blockchain-powered intelligent edge. IEEE Internet of Things Journal, 2022,

9(9): 6688–6702. [doi: 10.1109/JIOT.2021.3111552]

[23] Zhang J, Zhang FT, Huang X, Liu X. Leakage-resilient authenticated key exchange for edge artificial intelligence. IEEE Trans. on Dependable and Secure Computing, 2021, 18(6): 2835–2847. [doi: 10.1109/TDSC.2020.2967703]

[24] Hou DK, Zhang J, Man KL. Enhancing the security of numeric comparison secure simple pairing in bluetooth 5.0. In: Proc. of the 19th IEEE Int'l Conf. on Trust, Security and Privacy in Computing and Communications. Guangzhou: IEEE, 2021. 1622–1629. [doi: 10.1109/TrustCom50675.2020.00224]

[25] Tseng YM, Chen JL, Huang SS. A lightweight leakage-resilient identity-based mutual authentication and key exchange protocol for resource-limited devices. Computer Networks, 2021, 196: 108246. [doi: 10.1016/j.comnet.2021.108246]

[26] Garg S, Jain A, Sahai A. Leakage-resilient zero knowledge. In: Proc. of the 31st Annual Cryptology Conf. Santa Barbara: Springer, 2011. 297–315. [doi: 10.1007/978-3-642-22792-9_17]

[27] Aggarwal D, Damgård I, Nielsen JB, Obremski M, Purwanto E, Ribeiro J, Simkin M. Stronger leakage-resilient and non-malleable secret sharing schemes for general access structures. In: Proc. of the 39th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2019. 510–539. [doi: 10.1007/978-3-030-26951-7_18]

[28] Srinivasan A, Vasudevan PN. Leakage resilient secret sharing and applications. In: Proc. of the 39th Annual Int'l Cryptology Conf. Santa Barbara: Springer, 2019. 480–509. [doi: 10.1007/978-3-030-26951-7_17]

附中文参考文献:

[19] 汤佳惠, 朱艳琴, 罗喜召. 抗自适应泄漏的基于身份加密方案. 通信学报, 2012, 33(7): 90–95. [doi: 10.3969/j.issn.1000-436X.2012.07.011]

张福泰(1965－), 男, 博士, 教授, 博士生导师, CCF 专业会员, 主要研究领域为密码学及其应用.

张杰(1986－), 女, 博士, 副教授, 博士生导师, 主要研究领域为公钥密码学, 密钥管理.