基于 SM2 算法的无证书多重签名及其在区块链交易中的应用

朱炳丞[1,2,3,4]，周凤[2]，田有亮[1,2,3,4]，向阿新[1,2,3,4]，熊伟[1]，彭长根[1,2,3,4]

（1. 贵州大学公共大数据国家重点实验室，贵州 贵阳 550025；2. 贵州大学计算机科学与技术学院，贵州 贵阳 550025；
3. 贵州大学密码学与数据安全研究所，贵州 贵阳 550025；
4. 贵州省密码学与区块链技术特色重点实验室，贵州 贵阳 550025）

摘　要： 多重签名广泛应用于区块链交易方案，随着区块链应用国产化需求的不断增长，安全高效的 SM2 算法日益缺少多重签名方面的研究。此外，现有方案大多依赖公钥基础设施（PKI）体系来实现证书管理，存在效率和可扩展性问题。为此，提出一种基于 SM2 算法的无证书多重签名方案。首先，在 SM2 密钥生成阶段引入无证书密码机制，避免代价高昂的证书管理，设计密钥持有证明，抵御恶意密钥攻击；其次，通过引入树形结构，设计“线上-线下”的 SM2 多重签名算法，实现签名生成的高效性和高可扩展性，并在随机预言机模型（ROM）下证明该方案满足选择消息攻击下的存在性不可伪造性（EUF-CMA）；最后，将所提方案应用于 Hyperledger Fabric 联盟链，优化区块链交易流程。性能分析结果表明，与现有签名方案相比，所提方案在保证安全性的前提下，有效降低了计算开销和通信开销。

关键词： 区块链；多重签名；SM2 算法；无证书密码；随机预言机模型

中图分类号： TP309　　**文献标志码：** A　　**DOI：** 10.19678/j.issn.1000-3428.0070317

Certificateless Multi-Signature Based on SM2 Algorithm and Its Application in Blockchain Transaction

ZHU Bingcheng[1,2,3,4], ZHOU Feng[2], TIAN Youliang[1,2,3,4], XIANG Axin[1,2,3,4], XIONG Wei[1], PENG Changgen[1,2,3,4]

(1. State Key Laboratory of Public Big Data, Guizhou University, Guiyang 550025, Guizhou, China;
2. College of Computer Science and Technology, Guizhou University, Guiyang 550025, Guizhou, China;
3. Institute of Cryptography and Data Security, Guizhou University, Guiyang 550025, Guizhou, China;
4. Guizhou Province Key Laboratory of Cryptography and Blockchain Technology, Guiyang 550025, Guizhou, China)

【Abstract】 Multi-signature is widely used in blockchain transaction schemes. Despite increasing demand for the localization of blockchain applications, research on multi-signature has not sufficiently focused on secure and efficient SM2 algorithms. Additionally, most existing solutions rely on the Public Key Infrastructure (PKI) system to implement certificate management, which poses efficiency and scalability issues. Therefore, this study proposes a certificateless multi-signature scheme based on the SM2 algorithm. First, in the SM2 key generation stage, a certificateless cryptographic mechanism is introduced to avoid expensive certificate management, and a key holding proof is designed to resist malicious key attacks. Second, by introducing a tree structure, an "online-offline" SM2 multi-signature algorithm is designed to achieve efficient and highly scalable signature generation. The scheme is proven to satisfy the Existential UnForgeability under Chosen Message Attacks (EUF-CMA) in a Random Oracle Model (ROM). Finally, the proposed solution is applied to the Hyperledger Fabric consortium chain to optimize the blockchain transaction process. Results of a performance analysis show that, compared with existing signature schemes, the proposed scheme is more effective in reducing computational and communication overhead while ensuring security.

【Key words】 blockchain; multi-signature; SM2 algorithm; certificateless cryptography; Random Oracle Model (ROM)

0　引言

区块链[1]是一种分布式账本技术，具备去中心化、数据无法篡改和高透明度的特性。目前，区块链在金融、供应链、电子商务等领域得到广泛应用，区块链数据交易的重要性日益提高，但同时也带来了

基金项目： 国家重点研发计划（2022YFB2701400）；国家自然科学基金（62272123）；贵州省高层次创新型人才项目（黔科合平台人才[2020]6008）；贵州省科技计划项目（黔科合平台人才[2020]5017，黔科合支撑[2022]一般 065，黔科合战略找矿[2022]ZD001）；贵阳市科技计划项目（筑科合[2022]2-4）。

作者简介： 朱炳丞，男，硕士研究生，主研方向为密码学与安全协议、隐私保护技术；周凤（通信作者），副教授；田有亮，教授、博士生导师；向阿新，博士研究生；熊伟，讲师；彭长根，教授、博士生导师。

收稿日期： 2024-09-03　　**修回日期：** 2024-10-10　　**E-mail：** 41544782@qq.com

钓鱼攻击、私钥被盗、交易替代等安全隐患。2022 年,Trust 钱包因网络钓鱼电子邮件活动,大量个人信息被窃取,超 4 000 万美元加密货币被盗。2023 年,加密交易所 CoinEx 因私钥被盗,发生 7 000 万美元盗窃案。

区块链交易系统大多采用标准的椭圆曲线数字签名算法(ECDSA)[2]来保障交易的签名安全,交易双方通常依赖单一的签名来验证和确认交易,由于签名算法计算量繁重且区块存储空间有限,导致吞吐量低下,无法满足目前的应用需求。为解决这些问题,研究人员提出了多重签名[3]的思想,其是一种多个参与方签署一个交易并生成单个紧凑签名的机制,能够有效提升交易的传输和验证效率,提高交易的安全防护水平。目前,在针对区块链的多重签名方案中,文献[4]利用多重签名比特币地址代替单一的比特币地址,实现了区块链交易的安全性,文献[5]采用了文献[6]提出的 CoSi 方案,使区块链交易验证时间缩短,提高了区块链的吞吐量和效率。

然而,文献[4-6]方案依赖于公钥基础设施(PKI)体系,用户需要通过证书颁发机构(CA)颁发的证书进行身份认证,但是证书的分发、吊销、存储和验证需要耗费较高的成本。针对该问题,文献[7]提出了无证书密码机制,密钥生成不需要 CA 介入,而是由密钥生成中心(KGC)与用户本身共同完成,有效地克服了之前 PKI 体系的证书管理问题,因此,无证书多重签名方案得到广泛应用。文献[8]提出一种无证书有序多重签名方案,实现了分布式环境下无须证书管理中心的高效认证。文献[9]针对文献[8]方案进行改进,优化无证书密钥分发,使其能够抵抗伪造攻击。但是,以往无证书多重签名方案中的密钥大小和签名效率都未能被很好地优化。

SM2 算法[10]作为中国首个公开发布的公钥密码算法标准,对于保障信息系统安全和推动国内商用密码产品及信息安全体系建设具有重大意义[11]。SM2 算法轻量的密钥大小和高效的签名速度,能够有效提升区块链应用中多方参与签名的安全性和效率。已有研究者提出基于 SM2 算法的多方签名算法。文献[12]提出了基于 SM2 算法的局部可验证聚合签名,实现了 SM2 签名多方参与的批量验证。文献[13]提出了基于 SM2 算法的可否认环签名,实现了环成员并发进行的 SM2 签名。但是,现有方案的计算和通信开销都会随着签名者数量的增加而线性增长,无法满足区块链应用中多用户、大规模、高并发的需求。同时,SM2 算法在能满足这些需求的无证书多重签名方案中的研究还存在空白。因此,设计基于 SM2 算法的无证书多重签名,能够推动区块链应用发展自主化和可控性,丰富国产密码体系。本文的主要贡献有 3 个方面:

1)提出一种基于 SM2 算法的无证书多重签名方案。该方案通过引入无证书密码机制,避免证书管理和密钥托管问题,并设计持有证明,实现对恶意密钥攻击的抵抗。同时,引入树形结构,设计“线上-线下”的 SM2 多重签名算法,保证签名的紧凑性,且验证效率不随签名者数量的增多而降低。

2)设计一种基于 SM2 无证书多重签名算法的区块链交易方案,用 KGC 与用户共同生成密钥的机制替代传统 Fabric CA 的登记注册流程,并采用背书节点和同步节点注册信息,实现 Fabric 交易流程中背书签名和验证效率的提升。

3)分析本文方案的正确性,并在随机预言机模型(ROM)下证明本文方案的安全性。同时,对本文方案的计算和通信开销进行分析与对比。

1 相关工作

多重签名技术具备签名压缩和快速验证的优势,可在实现认证的同时降低验签开销,使区块链系统吞吐量和存储空间得到优化,是保障区块链交易安全的重要技术手段,因而围绕多重签名的研究具备一定的价值。文献[14]在普通公钥模型下,基于 Schnorr 签名提出了一个不支持公钥聚合的三轮多重签名算法,并在 ROM 下证明了该方案的安全性。文献[15]将缩减多重签名方案的通信轮次作为优化目标,对文献[14]方案进行优化,引入了同态陷门承诺,节约了一轮通信,提出了一个离散对数假设下的两轮 Schnorr 型多重签名方案 BCJ。文献[16]提出了首个基于 Okamoto 的两轮多重签名方案,该方案可兼容现有的 PKI 体系,且具有良好的性能表现。为响应分布式场景下的签名需求,文献[6]引入了 Merkle 树结构,提出了一个高度可扩展的 Schnorr 型多重签名方案 CoSi,该方案将签名的计算开销分摊到树中的各个节点以降低单个节点的负担,从而支持树结构中 8 192 个签名者在 2 s 内协同生成一个多重签名。虽然 CoSi 方案并未提供严格的安全证明,但其作为将树结构引入多重签名方案的典型案例,为后续研究提供了思路,更多的研究利用树结构增强多重签名方案的可扩展性[17-18],其中,文献[18]引入了“线上-线下”模式,支持签名者进行预计算,从而节约签名者收到消息后的签名时间。

针对已有两轮多重签名方案签名尺寸较大(如方案 mBCJ[19])、通信轮次过多(如方案 MuSig[20])、签名生成效率较低(如方案 MuSig-DN[21])等性能问题,文献[22]提出了一个在并发会话下安全的两轮多重签名方案 MuSig2,该方案生成的签名大小与普通 Schnorr 签名一致,且输出签名的复杂度接近普通 Schnorr 签名,有效避免了 mBCJ 面临的效率问题。然而,MuSig2 需要消耗较多资源来协商签名者间的公共参数。文献[23]提出一种签名高效的无证书有序聚合方案,对于多用户进行顺序签名的验证,优化了签名过程的复杂性,但在验证阶段使用了耗时的双线性配对运算。文献[24]随后提出一种无对运算的无证书有序多重签名方案,其不含双线性对操作,但多用户验证的开销会随用户数量线性增长。

SM2 签名算法是中国密码学家设计的一种基于椭圆曲线密码学的数字签名算法,对于 SM2 签名算法的分布式方案已有不少研究。文献[25]提出了一种基于 SM2 的椭圆曲线门限密码算法,提出的门限签名方案可容忍对 t 成员的窃听攻击。文献[26]提出了一种安全的两方协作 SM2 数字签名算法,在通用可组合安全框架下进行证明,与已有 SM2 协作签名方案相比,其交互次数更少,协作签名效率更高。文献[27]提出了基于 SM2 数字签名算法的环签名方案,还提出了一种可连接的 SM2 环签名方案,满足正确性、不可伪造性、无条件匿名性、可链接性和不可诽谤性。文献[28]提出了一种高效的 SM2 数字签名批量验证算法,将最为耗时的点乘运算由 n 次降为 1 次(n 为签名数量),大大缩短了验证时间,增强了 SM2 签名算法的应用部署效率。文献[29]提出了一种高效的 SM2 签名批量验证方案,与一次验证不同,它通过同时验证多个签名来提高计算速度。文献[30]提出的基于 Fabric 的 SM2 签名方案,给出了使用 SM2 算法的 Fabric 链地址的生成过程。

2 预备知识

2.1 多重签名

按照文献[14]的定义,一个多重签名方案描述为系统初始化算法 Setup、密钥生成算法 KeyGen、密钥聚合算法 KeyAgg、签名算法 Sign、验证算法 Verify,具体算法如下:

1)系统初始化算法 Setup。输入系统安全参数 k,生成系统参数 params。

2)密钥生成算法 KeyGen。输入系统参数 params,签名者生成各自的密钥对(sk_i,pk_i)和公钥集合 $P=\{\mathrm{pk}_1,\cdots,\mathrm{pk}_n\}$。

3)密钥聚合算法 KeyAgg。输入公钥集合 P,输出聚合公钥 APK。

4)签名算法 Sign。输入系统参数 params、聚合公钥 APK、各签名者私钥 sk_i 和消息 m,输出一个多重签名 σ。

5)验证算法 Verify。输入系统参数 params、聚合公钥 APK、消息 m 和签名 σ,输出 1 或 0,分别表示多重签名是否有效。

2.2 SM2 签名算法

SM2 签名算法[10]是中国自主研发的公钥密码算法标准之一。定义 F_p 为有限域,p 为大素数,E 为 F_p 上的椭圆曲线,$\mathbb{G}$ 是 E 上的 q 阶群,g 为 $\mathbb{G}$ 的生成元。SM2 签名算法具体描述如下:

1)系统初始化算法 Setup。输入系统安全参数 k,生成系统参数 params。

2)密钥生成算法 KeyGen。输入系统参数 params,签名者选择随机数 $d\in\mathbb{Z}_q^*$ 作为私钥,计算公钥 $P=d\cdot g$。

3)签名算法 Sign。输入安全参数 params、私钥 d、公钥 P 和消息 m,签名者选择一个随机数 $k\in\mathbb{Z}_q^*$,计算椭圆曲线上的点 $k\cdot g=(x_1,y_1)$,计算 $e=H(m)$,其中 H 是国密 SM3 哈希算法[31];计算 $r=(e+x_1)\bmod q$,若 $r=0$ 或者 $r+k=q$,则重新选择 k,计算 $s=(1+d)^{-1}\cdot(k-rd)\bmod q$,若 $s=0$,则重新选择 k;输出签名 $\sigma=(r,s)$。

4)验证算法 Verify。输入系统参数 params、公钥 P、消息 m 和签名 σ。验证 $r',s'\in\mathbb{Z}_q^*$,$r'+s'\neq q$ 是否成立,若不成立则验证失败,否则计算 $t=(r'+s')\bmod q$,若 $t=0$ 则验证失败,否则计算椭圆曲线上的点 $(x_1',y_1')=s\cdot g+t\cdot P$,计算 $R=(e'+x_1')$,验证 $R=r$ 是否成立,若成立则验证成功,否则验证失败。

2.3 离散对数困难问题

椭圆曲线离散对数问题(ECDLP):给定 2 个椭圆曲线 $E(F_p)$ 上的点 P、$Q\in E(F_p)$。如果存在一个正整数 $k\in\mathbb{Z}_q^*$,使得 $Q=kP$ 成立,计算 k 的值。

ECDLP 的困难性为随机选择算法求解 k 的值。设概率多项式时间算法 A 可用于解决 ECDLP,则成功解决 ECDLP 的概率为:

$$\Pr[\mathbb{A}(P,kP)=a\mid a\in\mathbb{Z}_q^*] \tag{1}$$

若任意概率多项式时间算法 A 解决 ECDLP 的概率可忽略不计,则 ECDLP 被视为数学上的困难问题。

2.4 一般分叉引理

证明数字签名方案的安全性时，常用的模型之一是 ROM[32]。在 ROM 中，数字签名方案所使用的哈希函数中至少有一个被视为随机预言机，证明数字签名方案安全性的重要技术之一是随机预言机重放技术，这一技术的理论基础源于著名的一般分叉引理[14]。本文将利用一般分叉引理来证明方案的安全性。

引理 1（一般分叉引理[14]） 设 q 是一个正整数，H 是一个包含 $h>2$ 个元素的集合，Λ 是一个随机概率算法，当给定可访问大小为 h 的随机预言机 $\mathcal{O}$ 的输入$\{x,(h_1,h_2,\cdots,h_q,\rho)\}$，其中 x 通过参数生成器 $\mathcal{IG}$ 生成，ρ 为指向 $\mathcal{A}$ 的随机磁带，则 Λ 输出(J,σ)，其中 $J\in\{1,\cdots,q\}$。设 P_{acc} 为成功输出 J 为非空集合的概率。令与 Λ 相关的分叉算法 $\mathcal{F}_\Lambda$ 描述如下：

输入 x，为 Λ 随机选取 $\rho\in\{0,1\}$，从集合 H 中随机选择 $h_1,h_2,\cdots,h_q$，算法输出(J,σ)，若 $J=0$，则返回$(0,\perp,\perp)$。从集合 H 中随机选择 $h'_1,h'_2,\cdots,h'_q$，算法输出(J',σ')，若 $J=J'$ 且 $h_J\neq h'_J$，则返回$(1,\sigma,\sigma')$，否则返回$(0,\perp,\perp)$。设 $P_{\text{frk}}=\Pr[b=1:x\leftarrow\mathcal{IG};(b,\sigma,\sigma')\leftarrow\mathcal{F}_\Lambda(x)]$为 $\mathcal{F}_\Lambda$ 成功输出$(1,\sigma,\sigma')$的概率，则有：

$$P_{\text{frk}}\geqslant P_{\text{acc}}\left(\frac{P_{\text{acc}}}{q}-\frac{1}{h}\right) \tag{2}$$

2.5 恶意密钥攻击

由于多重签名引入了多个用户的公钥参与，因而容易受到恶意密钥攻击(Rogue-key Attack)[33]，目前存在于包括文献[6]等多重签名方案中。一个恶意签名者 S_1 的私钥为 sk_1，将自身的公钥设置为 $\text{pk}_1=g\cdot\text{sk}_1-\sum_{i=2}^{n}\text{pk}_i$，其中 $\text{pk}_i(2\leqslant i\leqslant n)$为所有签名者 $S_i(1\leqslant i\leqslant n)$的公钥，该公钥会破坏公钥聚合算法，以 pk_1 取代 APK，恶意签名者 S_1 就可以独立地伪造消息 m 上的多重签名。为了抵御恶意密钥攻击，在产生多重签名之前，每个签名者需要对自己的密钥进行证明。

持有证明(Proof of Possession)[33]广泛用于抵御多重签名方案的恶意密钥攻击，要求签名者在生成公钥时证明拥有相应的私钥。这通常是通过非交互式零知识证明来实现的，该证明在不泄露私钥的情况下验证签名者对私钥的知识。

3 无证书多重签名方案及其安全模型

3.1 无证书多重签名方案

一个无证书多重签名方案定义为系统初始化算法 Setup、部分私钥提取算法 PartialKeyExtract、秘密值生成算法 SecretValue、密钥生成算法 KeyGen、密钥聚合算法 KeyAgg、签名算法 Sign、验证算法 Verify，算法具体描述如下：

1）系统初始化算法 Setup。输入安全参数 k，KGC 生成系统参数 params 和系统主密钥 msk。

2）部分私钥提取算法 PartialKeyExtract。签名者向 KGC 输入 params 和用户身份 ID_i，KGC 输出部分私钥 u_i。

3）秘密值生成算法 SecretValue。输入 params 和 ID_i，签名者随机取秘密值 v_i。

4）密钥生成算法 KeyGen。输入 params、u_i、v_i，输出用户完整公私钥对$(\text{sk}_i,\text{pk}_i)$。

5）密钥聚合算法 KeyAgg。输入公钥集合$\{\text{pk}_1,\text{pk}_2,\cdots,\text{pk}_n\}$，输出聚合公钥 APK。

6）签名算法 Sign。输入 params、APK、各签名者私钥 sk_i 和消息 m，输出一个多重签名 σ。

7）验证算法 Verify。输入 params、APK、m 和 σ，输出 1 或 0 表示多重签名是否有效。

在无证书机制下的多重签名系统中，恶意密钥攻击仍然是一个潜在的问题。无证书多重签名系统中有一部分公钥由用户个人生成和保管，无法确认用户是否拥有相应的私钥，这可能使得多重签名更容易受到恶意密钥攻击。因此，本文在密钥生成算法中引入持有证明，并在公钥聚合之前设计相应的密钥验证算法。

3.2 安全模型

针对无证书多重签名方案，本文考虑以下两种敌手类型：第一类敌手 $\mathcal{A}_\text{I}$ 模拟恶意签名者，被允许替换任意用户的公钥，而无法获取系统主密钥；第二类敌手 $\mathcal{A}_\text{II}$ 则模拟恶意 KGC，允许获取系统主密钥，但无法替换任意用户的公钥。方案的安全性验证通过挑战者 $\mathcal{C}$ 与敌手 $\mathcal{A}_\text{I}$、$\mathcal{A}_\text{II}$ 之间的攻击游戏进行模拟。若敌手 $\mathcal{A}_\text{I}$、$\mathcal{A}_\text{II}$ 能够成功对方案发动攻击，那么可以借助敌手 $\mathcal{A}_\text{I}$、$\mathcal{A}_\text{II}$ 来破解椭圆曲线离散对数困难问题。

定理 1 如果两类敌手 $\mathcal{A}_\text{I}$、$\mathcal{A}_\text{II}$ 赢得以下两个游戏的概率可以忽略不计，那么无证书多重签名方案被认为满足选择消息攻击下的存在性不可伪造性(EUF-CMA)。

1）游戏Ⅰ。此游戏涉及敌手 $\mathcal{A}_1$ 和挑战者 $\mathcal{C}_1$ 的互动。在游戏进行过程中，$\mathcal{C}_1$ 会使用一系列列表来记录所有的查询和它们所收到的回答。最终，$\mathcal{C}_1$ 将利用这些记录的信息来解决 ECDLP。以下是游戏的详细信息。

初始化阶段：挑战者 $\mathcal{C}_{\mathrm{I}}$ 通过执行 Setup 算法输出系统参数发送给 $\mathcal{A}_{\mathrm{I}}$，将系统主密钥 msk 保密。

查询阶段：$\mathcal{A}_{\mathrm{I}}$ 试图获取其他未被攻破的签名者的部分签名，将被允许自适应地查询以下预言机：

(1)哈希预言机：$\mathcal{A}_{\mathrm{I}}$ 具有权利查询任意输入的哈希值。

(2)部分私钥预言机：当 $\mathcal{A}_{\mathrm{I}}$ 想要查询身份为 ID_i 的部分私钥 u_i 时，$\mathcal{C}_{\mathrm{I}}$ 生成 u_i 发送给 $\mathcal{A}_{\mathrm{I}}$。

(3)秘密值预言机：收到查询后，$\mathcal{C}_{\mathrm{I}}$ 执行 SecretValue 算法生成相应秘密值 v_i 发送给 $\mathcal{A}_{\mathrm{I}}$。

(4)公钥预言机：收到查询后，$\mathcal{C}_{\mathrm{I}}$ 执行 KeyGen 算法计算相应的公钥 pk_i 发送给 $\mathcal{A}_{\mathrm{I}}$。

(5)公钥替换预言机：对于用户 ID_i，$\mathcal{A}_{\mathrm{I}}$ 用新的公开公钥 pk_i' 发送给 $\mathcal{C}_{\mathrm{I}}$，$\mathcal{C}_{\mathrm{I}}$ 查询列表并替换原来的公钥 pk_i。

(6)签名预言机：$\mathcal{A}_{\mathrm{I}}$ 在消息 m_i 上查询身份 ID_i 的签名时，$\mathcal{C}_{\mathrm{I}}$ 查询相关列表记录，找到必要的参数后，执行 Sign 算法生成签名 σ_i 发送给 $\mathcal{A}_{\mathrm{I}}$。

伪造阶段：敌手 $\mathcal{A}_{\mathrm{I}}$ 在进行多项式预言机查询后，输出签名者集合对消息 m^* 的多重签名 σ^*。$\mathcal{A}_{\mathrm{I}}$ 在以下条件取得游戏Ⅰ的胜利：

(1)多重签名 σ^* 必须有效，满足验证算法。

(2)$\mathcal{A}_{\mathrm{I}}$ 没有对秘密值进行查询。

引理 2 若敌手 $\mathcal{A}_{\mathrm{I}}$ 在 ROM 下概率多项式时间 t 内对游戏Ⅰ获胜的概率 ε 可以忽略不计，则无证书多重签名对于敌手 $\mathcal{A}_{\mathrm{I}}$ 满足 EUF-CMA。

2)游戏Ⅱ。此游戏涉及敌手 $\mathcal{A}_{\mathrm{II}}$ 和挑战者 $\mathcal{C}_{\mathrm{II}}$ 的互动。在游戏进行过程中，$\mathcal{C}_{\mathrm{II}}$ 会使用一系列列表来记录所有的查询和它们所收到的回答。最终，$\mathcal{C}_{\mathrm{II}}$ 将利用这些记录的信息来解决 ECDLP。以下是游戏的详细信息。

初始化阶段：挑战者 $\mathcal{C}_{\mathrm{II}}$ 通过执行 Setup 算法输出系统参数和系统主密钥 msk，发送给 $\mathcal{A}_{\mathrm{II}}$。

查询阶段：敌手 $\mathcal{A}_{\mathrm{II}}$ 被允许自适应地发起“哈希预言机查询”、“部分私钥预言机查询”、“公钥预言机查询”、“秘密值预言机查询”和“签名预言机查询”，具体过程类似于游戏Ⅰ中的预言机查询。

伪造阶段：敌手 $\mathcal{A}_{\mathrm{II}}$ 在进行多项式预言机查询后，输出签名者集合对消息 m^* 的多重签名 σ^*。$\mathcal{A}_{\mathrm{II}}$ 在以下条件取得游戏Ⅱ的胜利：

(1)多重签名 σ^* 必须有效，满足验证算法。

(2)$\mathcal{A}_{\mathrm{II}}$ 没有对秘密值进行查询。

引理 3 若敌手 $\mathcal{A}_{\mathrm{II}}$ 在 ROM 下概率多项式时间 t 内在游戏Ⅱ中获胜的概率 ε 可以忽略不计，则无证书多重签名对于敌手 $\mathcal{A}_{\mathrm{II}}$ 满足 EUF-CMA。

4 具体方案

4.1 基于 SM2 算法的无证书多重签名方案

1)系统初始化算法 Setup。输入系统安全参数 k，生成大素数 q 与定义在有限域 F_p 上的椭圆曲线 $E(F_p)$。$\mathbb{G}$ 是由 F_p 中的点组成的 q 阶的加法群，g 是该群的生成元。KGC 选择 $\mathrm{msk}\in\mathbb{Z}_q^*$ 作为系统主密钥，msk 由 KGC 秘密保存，并计算系统主公钥 $P_{\mathrm{pub}}=\mathrm{msk}\cdot g$，公开系统主公钥。选取安全抗碰撞哈希函数 $H_1:\{0,1\}^*\times\mathbb{G}\rightarrow\mathbb{Z}_q^*$，$H_2:\mathbb{G}\times\mathbb{G}\rightarrow\mathbb{Z}_q^*$，$H_3:\mathbb{G}\times\mathbb{G}\times\mathbb{G}\rightarrow\mathbb{Z}_q^*$，$H_4:\{0,1\}^*\rightarrow\mathbb{Z}_q^*$。公开参数 $\mathrm{params}=\{F_p,E(F_p),\mathbb{G},g,P_{\mathrm{pub}},H_1,H_2,H_3,H_4\}$。

2)部分私钥提取算法 PartialKeyExtract。KGC 收到签名者 i 提交的身份 ID_i，选择随机数 $y_i\in\mathbb{Z}_q^*$，计算 $q_i=H_1(\mathrm{ID}_i,Y_i)$，$u_i=y_i+\mathrm{msk}\cdot q_i$，$Y_i=y_ig$，将 (u_i,Y_i) 发送给签名者 i。签名者 i 验证等式 $u_ig=Y_i+q_iP_{\mathrm{pub}}$ 是否成立，若成立则接受 u_i，否则拒绝。

3)秘密值生成算法 SecretValue。签名者 i 选择随机数 $v_i\in\mathbb{Z}_q^*$ 作为秘密值。

4)密钥生成算法 KeyGen。签名者将部分私钥 u_i 和秘密值 v_i 相结合，得到私钥 $\mathrm{sk}_i=u_i+v_i$。签名者 i 计算 $X_i=u_ig$，$T_i=v_ig$，签名者 i 的公钥为 $\mathrm{pk}_i=X_i+T_i$。签名者 i 选取随机数 $r_i\in\mathbb{Z}_q^*$，计算 $R_i=r_ig$，$a_i=H_2(g,R_i)$，$b_i=H_1(\mathrm{ID}_i,\mathrm{pk}_i)$ 和 $\delta_i=((r_i+a_i)/(b_i\cdot\mathrm{sk}_i))\bmod q$，构造 sk_i 的持有证明 $\pi_i=(a_i,\delta_i)$，防止恶意密钥攻击。最后输出 $\mathrm{PK}_i=(\mathrm{pk}_i,\pi_i)$。

5)密钥验证算法 KeyVf。输入签名者 i 的 $\mathrm{PK}_i=(\mathrm{pk}_i,\pi_i)$，$\pi_i=(a_i,\delta_i)$，其余签名节点检查是否满足 $a_i=H_2(g,R_i)$，其中 $R_i=\delta_i\cdot b_i\cdot\mathrm{pk}_i-a_i\cdot g$。若满足则接受该公钥，否则拒绝。

6)密钥聚合算法 KeyAgg。签名节点将签名所涉及的每个 PK_i 解析为 $\mathrm{PK}_i=(\mathrm{pk}_i,\pi_i)$，计算并输出聚合公钥 $\mathrm{APK}=\sum\mathrm{pk}_i$，本文将密钥聚合算法运行在签名算法中的承诺阶段和挑战阶段。

7)签名算法 Sign。该签名算法引入了一种可扩展的树形结构，具体而言，该树并非严格的二叉树，而是具有可支配的分支因子，根据系统性能需求与节点数量，本文选择使用 B 阶树，其中 B 代表每个节点的最大子节点数，树的深度近似为 $O(\log_B N)$，

其中 N 为参与签名的节点数量。这种结构既保证了高效的通信和计算，也确保了良好的扩展性[6]。如图 1 所示，设当前签名节点为 S_i，C_{ij} 为 S_i 的子节点，P_i 为 S_i 的父节点。S_i 收集来自其子节点的数据，并将结果逐级传递给其父节点 P_i，最终到达根节点 S_0。在本文的设定中，若 S_i 没有子节点，则为叶子节点，仅承担自己的部分计算和初始传递任务；若 S_i 没有父节点，则为根节点 S_0，负责汇总和输出最终的聚合数据。以下是具体过程：

承诺：对于 S_i，随机选择秘密值 $l_i\in\mathbb{Z}_q^*$，计算 $L_i=l_ig=(x_i,y_i)$，等待每个子节点 C_{ij} 的部分承诺 $\tilde{L}_{ij}$ 和部分聚合公钥 $\widehat{\mathrm{APK}}_i=\mathrm{pk}_i+\sum\mathrm{APK}_{ij}$。当 S_i 收到所有部分承诺之后，计算 $\tilde{L}_i=L_i+\sum\tilde{L}_{ij}$。之后将 $\tilde{L}_i$ 和 $\widehat{\mathrm{APK}}_i$ 发送给其父节点 P_i，直到 S_0。

挑战：若 P_i 为领导者 S_0，在收到部分承诺 $\tilde{L}_{0j}$ 和部分聚合公钥 $\widehat{\mathrm{APK}}_{0j}$ 后，计算最终承诺 $\tilde{L}=\tilde{L}_0=L_0+\sum\tilde{L}_{0j}$ 和最后聚合的公钥 $\widehat{\mathrm{APK}}=\widehat{\mathrm{APK}}_0=\mathrm{pk}_0+\sum\mathrm{APK}_{0j}$。计算集体挑战 $c=H_3(g,\tilde{L},\widehat{\mathrm{APK}})$。

通告：领导者 S_0 收到 m 并广播。

响应：当 S_i 收到消息 m 后，计算 $e=H_4(m)$，$h_i=(e+x_i)$ 和 $s_i=(1+u_i+v_i)^{-1}\cdot(l_i\cdot c-h_i\cdot(u_i+v_i))\bmod q$，当收到其子节点 C_{ij} 发送的部分响应值 $\tilde{s}_{ij}$ 和 $\tilde{h}_{ij}$ 后计算 $\tilde{h}_i=h_i+\sum\tilde{h}_{ij}$、$\tilde{s}_i=s_i+\sum\tilde{s}_{ij}$ 并将 $\tilde{s}_i$、$\tilde{h}_i$ 发送给其父节点 P_i，以此类推，直到领导者 S_0。最后，由 S_0 计算最终的 $s=\tilde{s}_0=s_0+\sum\tilde{s}_{0j}$，$h=\tilde{h}_0=h_0+\sum\tilde{h}_{0j}$ 并输出最后的多重签名 $\sigma=(c,h,s)$。

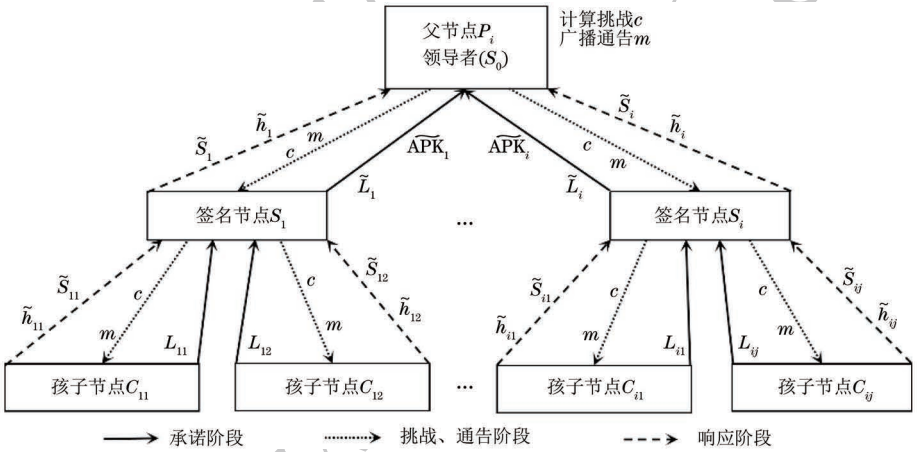

图 1 签名算法流程

Fig. 1 Signature algorithm process

本方案具有“线上-线下”签名过程。定义签名算法中的承诺阶段和挑战阶段为“线下”签名阶段，即在消息 m 到来之前，每个树形结构中的签名者将公钥进行聚合，并达成共识生成挑战值 c，c 作为签名的一部分，可以提前发送给验证者，并自顶向下发送给子节点，所有签名者都可以预先计算各自的 $l^i\cdot c$。通告阶段和响应阶段为“线上”签名阶段，在收到待签名消息 m 后，仅需进行响应值的计算即可输出多重签名。

8）验证算法 Verify。输入 $\widehat{\mathrm{APK}}$、m 和 σ，计算 $c=H_1(g,\tilde{L},\widehat{\mathrm{APK}})$，$(x',y')=[sg+(h+s)\mathrm{APK}]c^{-1}$，$H=(e+x')\bmod q$，其中 $e=H_4(m)$，最后检验 $H=h$ 是否成立，若成立，则验证通过，若不成立，则验证不通过。

4.2 正确性

本文方案正确性验证如下：

$$
\begin{aligned}
&(x',y')=[s\cdot g+(h+s)\mathrm{APK}]c^{-1}=\\
&(s\cdot g+h\cdot\mathrm{APK}+s\cdot\mathrm{APK})c^{-1}=\\
&(s+h\cdot\mathrm{sk}+s\cdot\mathrm{sk})g\cdot c^{-1}=\\
&[(1+\mathrm{sk})^{-1}(l\cdot c-h\cdot\mathrm{sk})(1+\mathrm{sk})+\mathrm{sk}\cdot h]g\cdot c^{-1}=\\
&(l\cdot c-h\cdot\mathrm{sk}+h\cdot\mathrm{sk})g\cdot c^{-1}=\\
&l\cdot c\cdot g\cdot c^{-1}=\\
&l\cdot g=(x,y)
\end{aligned}
\tag{3}
$$

4.3 不可伪造性

一个安全的多重签名方案应该满足 EUF-CMA，本文证明所提多重签名方案能实现不可伪造性分析如下：

定理 2 如果两类敌手 $\mathcal{A}_\mathrm{I}$、$\mathcal{A}_\mathrm{II}$ 赢得两类游戏的概率可以忽略不计，那么无证书多重签名方案被认为满足 EUF-CMA。

引理 4 如果敌手 $\mathcal{A}_\mathrm{I}$ 在基于 ECDLP 假设的随机预言机模型下概率多项式时间 t 内在游戏 Ⅰ 中取得胜利的概率 ε 可以忽略不计，则所提方案对于

敌手 $\mathcal{A}_{\mathrm{I}}$ 满足 EUF-CMA。

证明 假设 $\mathcal{A}_{\mathrm{I}}$ 是类型Ⅰ的概率多项式时间对手，能够以不可忽略的概率伪造本文的多重签名方案，从而解决 ECDLP。挑战者 $\mathcal{C}_{\mathrm{I}}$ 给定一个随机实例 g，令 $Y=\lambda\cdot g$，$\mathcal{C}_{\mathrm{I}}$ 的目标是通过与 $\mathcal{A}_{\mathrm{I}}$ 交互输出 ECDLP 的解 λ。$\mathcal{C}_{\mathrm{I}}$ 与 $\mathcal{A}_{\mathrm{I}}$ 的交互如下：

初始化阶段：挑战者 $\mathcal{C}_{\mathrm{I}}$ 通过执行 Setup 算法，输出系统参数发送给 $\mathcal{A}_{\mathrm{I}}$，将系统主密钥 msk 保密。

查询阶段：$\mathcal{A}_{\mathrm{I}}$ 选择身份 $\mathrm{ID}_i^*\in(\mathrm{ID}_1,\mathrm{ID}_2,\cdots,\mathrm{ID}_n)$，试图获取其他未被攻破的签名者的部分签名，将被允许自适应地查询以下预言机。

H_1 预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{H_1}=\{\mathrm{ID}_i,Y_i,q_i,\mathrm{pk}_i,b\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_{\mathrm{I}}$ 以 (ID_i,Y_i) 进行 H_1 查询时，$\mathcal{C}_{\mathrm{I}}$ 首先查找列表 L_{H_1} 中是否存在 q_i。若存在，将 q_i 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $q_i\in\mathbb{Z}_q^*$，并设置 $q_i=H_1(\mathrm{ID}_i,Y_i)$ 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 q_i 更新到列表 L_{H_1} 中。敌手 $\mathcal{A}_{\mathrm{I}}$ 以 $(\mathrm{ID}_i,\mathrm{pk}_i)$ 进行 H_1 查询时，$\mathcal{C}_{\mathrm{I}}$ 首先查找列表 L_{H_1} 中是否存在 b_i。若存在，将 b_i 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $b_i\in\mathbb{Z}_q^*$，并设置 $b_i=H_1(\mathrm{ID}_i,\mathrm{pk}_i)$ 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 b_i 更新到列表 L_{H_1} 中。

H_2 预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{H_2}=\{R_i,a_i\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_{\mathrm{I}}$ 以 (g,R_i) 进行 H_2 查询时，$\mathcal{C}_{\mathrm{I}}$ 首先查找列表 L_{H_2} 中是否存在 a_i。若存在，将 a_i 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $a_i\in\mathbb{Z}_q^*$，并设置 $a_i=H_2(g,R_i)$ 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 a_i 更新到列表 L_{H_2} 中。

H_3 预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{H_3}=\{l_i,\mathrm{pk}_i,c\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_{\mathrm{I}}$ 以 (g,l_i,pk_i) 进行 H_3 查询时，$\mathcal{C}_{\mathrm{I}}$ 首先查找列表 L_{H_3} 中是否存在 c。若存在，将 c 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $c\in\mathbb{Z}_q^*$，并设置 $c=H_3(g,\widetilde{L},\widehat{\mathrm{APK}})$ 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 c 更新到列表 L_{H_3} 中。

H_4 预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{H_4}=\{m,e\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_{\mathrm{I}}$ 以 m 进行 H_4 查询时，$\mathcal{C}_{\mathrm{I}}$ 首先查找列表 L_{H_4} 中是否存在 e。若存在，将 e 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $e\in\mathbb{Z}_q^*$，并设置 $e=H_4(m)$ 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 e 更新到列表 L_{H_4} 中。

部分私钥预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{\mathrm{ppk}}=\{Y_i,\mathrm{ID}_i,u_i\}$，在开始阶段初始化为空表。令 $Y_i^*=\alpha\cdot g$，放入列表 L_{ppk} 中，$\mathcal{A}_{\mathrm{I}}$ 想要查询身份为 ID_i 的部分私钥 u_i，$\mathcal{C}_{\mathrm{I}}$ 在收到 $\mathcal{A}_{\mathrm{I}}$ 的 $H_1(\mathrm{ID}_i,Y_i)$ 查询时，若 $\mathrm{ID}_i\neq\mathrm{ID}_i^*$，查找列表 L_{ppk} 中是否存在对应的 u_i，若存在，将 u_i 返回给 $\mathcal{A}_{\mathrm{I}}$，否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $y_i\in\mathbb{Z}_q^*$，计算 $Y_i=y_ig$，提取列表 L_{H_1} 中对应的 q_i，计算 $u_i=y_i+q_i\cdot\mathrm{msk}$ 返回给 $\mathcal{A}_{\mathrm{I}}$ 并更新到列表 L_{ppk} 中；若 $\mathrm{ID}_i=\mathrm{ID}_i^*$，取消查询。

秘密值预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{\mathrm{ssv}}=\{\mathrm{ID}_i,v_i\}$，在开始阶段初始化为空表。$\mathcal{C}_{\mathrm{I}}$ 在收到 $\mathcal{A}_{\mathrm{I}}$ 关于 ID_i 的秘密值查询时，首先查找列表 L_{ssv} 中是否存在 v_i。若存在，将 v_i 返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $v_i\in\mathbb{Z}_q^*$ 并返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 v_i 更新到列表 L_{ssv} 中。

公钥预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_{\mathrm{pk}}=\{\mathrm{ID}_i,X_i,T_i\}$，在开始阶段初始化为空表。在收到 $\mathcal{A}_{\mathrm{I}}$ 关于 ID_i 的公钥查询时，若 $\mathrm{ID}_i\neq\mathrm{ID}_i^*$，查找列表 L_{pk} 中是否存在对应的 (X_i,T_i)。若存在，将其返回给 $\mathcal{A}_{\mathrm{I}}$。否则，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $v_i\in\mathbb{Z}_q^*$，计算 $T_i=v_ig$，并查询列表 L_{ppk} 中对应的 u_i，若不存在，则随机选择一个 $u_i\in\mathbb{Z}_q^*$，并计算 $X_i=u_ig$，将 (X_i,T_i) 返回给 $\mathcal{A}_{\mathrm{I}}$。最后 $\mathcal{C}_{\mathrm{I}}$ 将 (X_i,T_i) 更新到列表 L_{pk} 中，将 u_i 更新到列表 L_{ppk} 中。若 $\mathrm{ID}_i=\mathrm{ID}_i^*$，令 $X^*=Y^*+q_iP_{\mathrm{pub}}$，$T_i=v_ig$。

公钥替换预言机查询：在收到 $\mathcal{A}_{\mathrm{I}}$ 输入新的 T_i' 请求时，查找列表 L_{pk} 中是否存在对应的 T_i。若存在，用 $(T_i',\perp)$ 替换原有的 (T_i,v_i)，否则，将新的 T_i' 添加到列表 L_{pk} 中，其中 $\perp$ 表示此记录已经进行过公钥替换。对替换的公钥进行密钥验证，验证公钥的合法性。

签名预言机查询：挑战者 $\mathcal{C}_{\mathrm{I}}$ 维护一个列表 $L_s=\{\mathrm{ID}_i,m,\sigma\}$，在开始阶段初始化为空表。在收到 $\mathcal{A}_{\mathrm{I}}$ 关于 (ID_i,m) 的签名查询时，若 $\mathrm{ID}_i\neq\mathrm{ID}_i^*$，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $l_i\in\mathbb{Z}_n^*$，计算 $h_i=(e+x_i)$，$s_i=(1+u_i+v_i)^{-1}\cdot(l_i\cdot c-h_i\cdot(u_i+v_i))\bmod q$，其中 c、e、u_i、v_i 是查找相应的列表获取的。将 $\sigma=(c,h,s)$ 返回给 $\mathcal{A}_{\mathrm{I}}$ 并将 $\{\mathrm{ID}_i,m,\sigma\}$ 更新到列表 L_s 中。若 $\mathrm{ID}_i=\mathrm{ID}_i^*$，$\mathcal{C}_{\mathrm{I}}$ 随机选择一个 $l_i\in\mathbb{Z}_n^*$，计算 $(x_i,y_i)=l_ig$，$h_i=e+x_i$，其中 e 从列表 L_{H_4} 中提取，令 $c=-h_i\cdot l_i^{-1}$，计算：

$$\begin{aligned}s_i&=(1+u_i+v_i)^{-1}\cdot(l_i\cdot c-h_i\cdot(u_i+v_i))=\\&(1+u_i+v_i)^{-1}\cdot[l_i(-h_i\cdot l_i^{-1})-h_i(u_i+v_i)]=\\&(1+u_i+v_i)^{-1}\cdot[-h_i-h_i(u_i+v_i)]=-h_i\end{aligned}\tag{4}$$

伪造阶段：敌手 $\mathcal{A}_{\mathrm{I}}$ 在进行多项式预言机查询后，输出签名者集合 $\{\mathrm{ID}_1,\mathrm{ID}_2,\cdots,\mathrm{ID}_n\}$ 对消息 m^* 的多重签名 σ^*，其中，签名者的聚合公钥为 APK，

伪造的多重签名 $\sigma^*=\{c^*,h^*,s^*\}$ 满足验证等式 $(x',y')=[sg+(h+s)\mathrm{APK}]c^{-1}$。

根据一般分叉引理，$\mathcal{C}_\mathrm{I}$ 可设定相同的随机值和不同的哈希值来执行上述过程两轮，$\mathcal{A}_\mathrm{I}$ 可生成两个不同的、有效的伪造多重签名 $\sigma^{*(1)}=\{c^*,h^{*(1)},s^{*(1)}\}$ 和 $\sigma^{*(2)}=\{c^*,h^{*(2)},s^{*(2)}\}$，则有以下等式：

$$\begin{aligned}&s^{*(1)}-s^{*(2)}=\\&(1+u^*+v^*)^{-1}\cdot(l_i\cdot c^*-h^{*(1)}\cdot(u^*+v^*)-\\&l_i\cdot c^*+h^{*(2)}\cdot(u^*+v^*))=\\&(1+u^*+v^*)^{-1}\cdot[(h^{*(2)}-h^{*(1)})\cdot\\&(u^*+v^*)]s^{*(1)}-s^{*(2)}=\\&(h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)})u^*+\\&(h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)})v^*\end{aligned}\tag{5}$$

由式(5)可以得到：

$$\frac{s^{*(1)}-s^{*(2)}-(h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)})v^*}{h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)}}=u^*\tag{6}$$

由于 $u^*=\lambda+q_i\cdot\mathrm{msk}$,，则由式(6)可以解出：

$$\lambda=\frac{s^{*(1)}-s^{*(2)}-(h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)})v^*}{h^{*(2)}-h^{*(1)}+s^{*(2)}-s^{*(1)}}-q_i\cdot\mathrm{msk}\tag{7}$$

由此，$\mathcal{C}_\mathrm{I}$ 破解了 ECDLP。然而，ECDLP 是数学困难问题，故 $\mathcal{A}_\mathrm{I}$ 在游戏Ⅰ中获胜的假设不成立。因此，本文方案针对类型Ⅰ的敌手 $\mathcal{A}_\mathrm{I}$ 满足 EUF-CMA。

引理 5 如果敌手 $\mathcal{A}_\mathrm{II}$ 在基于 ECDLP 假设的随机预言机模型下概率多项式时间 t 内在游戏Ⅱ中取得胜利的概率 ε 可以忽略不计，则本文方案对于敌手 $\mathcal{A}_\mathrm{II}$ 满足 EUF-CMA。

证明 假设 $\mathcal{A}_\mathrm{II}$ 是类型Ⅱ的概率多项式时间对手，能够以不可忽略的概率伪造本文的多重签名方案，从而解决 ECDLP。挑战者 $\mathcal{C}_\mathrm{II}$ 给定一个随机实例 g，令 $T_i=v\cdot g$，$\mathcal{C}_\mathrm{II}$ 的目标是通过与 $\mathcal{A}_\mathrm{II}$ 交互输出 ECDLP 的解 v。$\mathcal{C}_\mathrm{II}$ 与 $\mathcal{A}_\mathrm{II}$ 的交互如下：

初始化阶段：挑战者 $\mathcal{C}_\mathrm{II}$ 通过执行 Setup 算法输出系统参数和系统主密钥，并发送给敌手 $\mathcal{A}_\mathrm{II}$。

查询阶段：$\mathcal{A}_\mathrm{II}$ 选择 $\mathrm{ID}_i^*\in(\mathrm{ID}_1,\mathrm{ID}_2,\cdots,\mathrm{ID}_n)$，试图获取其他未被攻破的签名者的部分签名，将被允许自适应地查询以下预言机。

H_1 预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{H_1}=\{\mathrm{ID}_i,Y_i,q_i,\mathrm{pk}_i,b\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_\mathrm{II}$ 以 (ID_i,Y_i) 进行 H_1 查询时，$\mathcal{C}_\mathrm{II}$ 首先查找列表 L_{H_1} 中是否存在 q_i。若存在，将 q_i 返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $q_i\in\mathbb{Z}_q^*$，并设置 $q_i=H_1(\mathrm{ID}_i,Y_i)$ 返回给 $\mathcal{A}_\mathrm{II}$。最后，$\mathcal{C}_\mathrm{II}$ 将 q_i 更新到列表 L_{H_1} 中。敌手 $\mathcal{A}_\mathrm{II}$ 以 $(\mathrm{ID}_i,\mathrm{pk}_i)$ 进行 H_1 查询时，$\mathcal{C}_\mathrm{II}$ 首先查找列表 L_{H_1} 中是否存在 b_i。若存在，将 b_i 返回给 $\mathcal{A}_\mathrm{II}$，否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $b_i\in\mathbb{Z}_q^*$，并设置 $b_i=H_1(\mathrm{ID}_i,\mathrm{pk}_i)$ 返回给 $\mathcal{A}_\mathrm{II}$。最后，$\mathcal{C}_\mathrm{II}$ 将 b_i 更新到列表 L_{H_1} 中。

H_2 预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{H_2}=\{R_i,a_i\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_\mathrm{II}$ 以 (g,R_i) 进行 H_2 查询时，首先查找列表 L_{H_2} 中是否存在 a_i。若存在，将 a_i 返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $a_i\in\mathbb{Z}_q^*$，并设置 $a_i=H_2(g,R_i)$ 返回给 $\mathcal{A}_\mathrm{II}$。最后，$\mathcal{C}_\mathrm{II}$ 将 a_i 更新到列表 L_{H_2} 中。

H_3 预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{H_3}=\{l_i,\mathrm{pk}_i,c\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_\mathrm{II}$ 以 (g,l_i,pk_i) 进行 H_3 查询时，首先查找列表 L_{H_3} 中是否存在 c。若存在，将 c 返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $c\in\mathbb{Z}_q^*$，并设置 $c=H_3(g,\widetilde{L},\widehat{\mathrm{APK}})$ 返回给 $\mathcal{A}_\mathrm{II}$。最后，$\mathcal{C}_\mathrm{II}$ 将 c 更新到列表 L_{H_3} 中。

H_4 预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{H_4}=\{m,e\}$，在开始阶段初始化为空表。敌手 $\mathcal{A}_\mathrm{II}$ 以 m 进行 H_4 查询时，首先查找列表 L_{H_4} 中是否存在 e。若存在，将 e 返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $e\in\mathbb{Z}_q^*$，并设置 $e=H_4(m)$ 返回给 $\mathcal{A}_\mathrm{II}$。最后 $\mathcal{C}_\mathrm{II}$ 将 e 更新到列表 L_{H_4} 中。

部分私钥预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{\mathrm{ppk}}=\{Y_i,\mathrm{ID}_i,u_i\}$，在开始阶段初始化为空表。$\mathcal{A}_\mathrm{II}$ 想要查询身份为 ID_i 的部分私钥 u_i，$\mathcal{C}_\mathrm{II}$ 在收到 $\mathcal{A}_\mathrm{II}$ 的 $H_1(\mathrm{ID}_i,Y_i)$ 查询时，首先查找列表 L_{ppk} 中是否存在对应的 (u_i,Y_i)。若存在，将 (u_i,Y_i) 返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $u_i\in\mathbb{Z}_q^*$，提取列表 L_{H_1} 中对应的 q_i，计算 $Y_i=u_ig-q_iP_{\mathrm{pub}}$ 并返回给 $\mathcal{A}_\mathrm{II}$。最后 $\mathcal{C}_\mathrm{II}$ 将 (u_i,Y_i) 更新到列表 L_{ppk} 中。

秘密值预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{\mathrm{ssv}}=\{\mathrm{ID}_i,v_i\}$，在开始阶段初始化为空表。$\mathcal{C}_\mathrm{II}$ 在收到 $\mathcal{A}_\mathrm{II}$ 关于 ID_i 的秘密值查询时，若 $\mathrm{ID}_i\neq\mathrm{ID}_i^*$，查找列表 L_{ssv} 中是否存在 v_i，若存在，将 v_i 返回给 $\mathcal{A}_\mathrm{II}$，否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $v_i\in\mathbb{Z}_q^*$ 并返回给 $\mathcal{A}_\mathrm{II}$，然后更新到列表 L_{ssv} 中。若 $\mathrm{ID}_i=\mathrm{ID}_i^*$，取消查询。

公钥预言机查询：挑战者 $\mathcal{C}_\mathrm{II}$ 维护一个列表 $L_{\mathrm{pk}}=\{\mathrm{ID}_i,X_i,T_i\}$，在开始阶段初始化为空表。在收到 $\mathcal{A}_\mathrm{II}$ 关于 ID_i 的公钥查询时，首先查找列表 L_{pk} 中是否存在对应的 (X_i,T_i)。若存在，将其返回给 $\mathcal{A}_\mathrm{II}$。否则，$\mathcal{C}_\mathrm{II}$ 随机选择一个 $v_i\in\mathbb{Z}_q^*$，计算 $T_i=v_ig$，并查询列表 L_{ppk} 中对应的 u_i，若不存在，则随

机选择一个 $u_i \in \mathbb{Z}_q^*$，并计算 $X_i = u_i g$，将(X_i, T_i)返回给 $\mathcal{A}_{\text{Ⅱ}}$。最后，$\mathcal{C}_{\text{Ⅱ}}$ 将(X_i, T_i)更新到列表 L_{pk} 中，将 u_i 更新到列表 L_{ppk} 中。

签名预言机查询：挑战者 $\mathcal{C}_{\text{Ⅱ}}$ 维护一个列表 $L_s = \{\text{ID}_i, X_i, T_i\}$，在开始阶段初始化为空表。在收到 $\mathcal{A}_{\text{Ⅱ}}$ 关于消息 m 的签名查询时，若存在对应的 σ，将其返回给 $\mathcal{A}_{\text{Ⅱ}}$。否则，$\mathcal{C}_{\text{Ⅱ}}$ 随机选择一个 $l_i \in \mathbb{Z}_q^*$，查找相应的列表获取 c、e、u_i、v_i，计算 $h_i = e + x_i$，$s_i = (1+u_i+v_i)^{-1} \cdot (l_i \cdot c - h_i \cdot (u_i + v_i)) \bmod q$，然后将 $\sigma = (c, h, s)$返回给 $\mathcal{A}_{\text{Ⅱ}}$。最后，$\mathcal{C}_{\text{Ⅱ}}$ 将$\{\text{ID}_i, m, \sigma\}$更新到列表 L_s 中。

伪造阶段：敌手 $\mathcal{A}_{\text{Ⅱ}}$ 在进行多项式预言机查询后，输出签名者集合$\{\text{ID}_1, \text{ID}_2, \cdots, \text{ID}_n\}$对消息 m^* 的多重签名 σ^*。其中，签名者的聚合公钥为 APK，伪造的多重签名 $\sigma^* = \{c^*, h^*, s^*\}$满足多重签名的验证等式$(x', y') = [sg + (h+s)\text{APK}]c^{-1}$。

根据一般分叉引理，$\mathcal{C}_{\text{Ⅱ}}$ 可设定相同的随机值和不同的哈希值，执行两轮上述过程，$\mathcal{A}_{\text{Ⅱ}}$ 可生成两个不同的、有效的伪造多重签名 $\sigma^{*(1)} = \{c^*, h^{*(1)}, s^{*(1)}\}$和 $\sigma^{*(2)} = \{c^*, h^{*(2)}, s^{*(2)}\}$，则有以下等式：

$$
\begin{aligned}
&s^{*(1)} - s^{*(2)} = \\
&(1+u^*+v^*)^{-1} \cdot (l_i \cdot c^* - h^{*(1)} \cdot (u^*+v^*) - \\
&l_i \cdot c^* + h^{*(2)} \cdot (u^*+v^*)) = \\
&(1+u^*+v^*)^{-1} \cdot [(h^{*(2)} - h^{*(1)}) \cdot (u^*+v^*)] \\
&(s^{*(1)} - s^{*(2)}) + (s^{*(1)} - s^{*(2)})(u^*+v^*) = \\
&(h^{*(2)} - h^{*(1)}) \cdot (u^*+v^*) \\
&s^{*(1)} - s^{*(2)} = \\
&(h^{*(2)} - h^{*(1)} + s^{*(2)} - s^{*(1)})u^* + \\
&(h^{*(2)} - h^{*(1)} + s^{*(2)} - s^{*(1)})v^*
\end{aligned} \tag{8}
$$

由式(8)解得：

$$
v^* = \frac{s^{*(1)} - s^{*(2)} - (h^{*(2)} - h^{*(1)} + s^{*(2)} - s^{*(1)})u^*}{h^{*(2)} - h^{*(1)} + s^{*(2)} - s^{*(1)}} \tag{9}
$$

由此，$\mathcal{C}_{\text{Ⅱ}}$ 能够解出 v_i，即破解了 ECDLP。然而，ECDLP 是数学困难问题，故 $\mathcal{A}_{\text{Ⅱ}}$ 在游戏Ⅱ中获胜的假设不成立。因此，本文方案针对类型Ⅱ的敌手 $\mathcal{A}_{\text{Ⅱ}}$ 满足 EUF-CMA。

由上述引理，两类敌手 $\mathcal{A}_{\text{Ⅰ}}$、$\mathcal{A}_{\text{Ⅱ}}$ 赢得两类游戏的概率可以忽略不计，则所提无证书多重签名方案满足 EUF-CMA。

本文所提多重签名方案可抵御恶意密钥攻击。在恶意密钥攻击中，恶意签名者利用参与多重签名的诚实签名者的公钥构造自身的公钥 pk^*，取代聚合公钥 APK 来独立伪造多重签名 σ^*。本方案在密钥生成阶段，要求签名者生成密钥持有证明，在公钥聚合之前对签名者密钥的知识进行验证，检查是否满足 $a_i = H_2(g, R_i)$，其中 $R_i = \delta_i \cdot b_i \cdot \text{pk}_i - a_i \cdot g$。该持有证明中包含了公钥 pk_i 对其私钥 sk_i 的知识，但不会泄露其私钥。如果恶意签名者通过恶意密钥攻击伪造公钥 pk^*，则该公钥无法通过验证，则抛弃。由于恶意签名者无法通过诚实签名者的公钥集合生成可用于伪造多重签名的自身公钥，因此本方案可抵御恶意密钥攻击。

5 基于多重签名的区块链交易应用

本文将基于 SM2 的无证书多重签名方案应用于 Fabric 的交易流程中，基于 Fabric 框架进行修改。用户通过使用该无证书多重签名签署交易提案，实现 Fabric 交易流程中背书签名和验证效率的提升。定义客户端 Cl、KGC 和区块链节点 3 类实体，区块链节点包括背书节点 En_i、排序节点 Or 和记账节点 Vr。本文将原 Fabric 框架中的 CA 替换成 KGC，进一步优化完善 Fabric 网络中的节点准入原则，减少网络维护、证书管理和存储的开销。交易核心过程如图 2 所示，具体描述如下：

1)注册 ID。当一个新节点请求加入 Fabric 网络时，该节点通过客户端 Cl 与 KGC 进行通信，向 KGC 提交自己的身份标识 ID。

2)密钥生成。KGC 执行 PartialKeyExtract 算法，生成该节点对应的部分私钥 u_i。节点收到 u_i 后验证其有效性，验证通过后执行 KeyGen 和 SecretValue 算法，生成自己的公私钥对$(\text{sk}_i, \text{pk}_i)$。

3)同步信息。所有背书节点 En_i 工作在一个树结构中。根节点执行 KeyVf 算法核实每个背书节点 En_i 的公钥拥有对应私钥。每个背书节点 En_i 同步执行 Sign 算法的承诺阶段，生成聚合公钥 $\widehat{\text{APK}}$ 和承诺 $\tilde{L}$。客户端 Cl 作为根节点，执行 Sign 算法的挑战阶段，产生共同挑战值 c。挑战值 c 作为多重签名的一部分，发送到每一个签名节点。

4)交易提案。当客户端 Cl 请求一个交易提案时，客户端作为根节点执行 Sign 算法的通告阶段，从上到下发送交易信息 m 至每个背书节点 En_i。

5)背书签名。各背书节点 En_i 收到客户端 Cl 发出的 m 后，执行 KeyVf 算法核实客户端身份是否合法，模拟交易结果，并执行 Sign 算法的响应阶段，计算响应值 s_i 和 h_i。

6)提案回应。各背书节点 En_i 继续执行 Sign 算法的响应阶段，自底向上发送提案响应，包括模拟交易结果与部分响应值 s_i 和 h_i。客户端 Cl 检查模

拟交易结果并收集来自所有孩子节点的响应值，计算最终响应值 $s=\bar{s}_0=s_0+\sum\bar{s}_{0j}$，$h=\tilde{h}_0=h_0+\sum\tilde{h}_{0j}$，从而生成多重签名 $\sigma=(c,h,s)$。

7)交易提交。客户端 Cl 收到背书签名后，发送交易提案和 σ 给排序节点 Or。

8)区块交付。排序节点 Or 收到来自不同客户端 Cl 的交易后，对交易进行排序，构造交易区块并广播到区块链网络。

9)账本更新。记账节点 Vr 执行 Verify 算法，以验证区块中每笔交易的有效性，并将有效的交易区块更新到公开账本中。

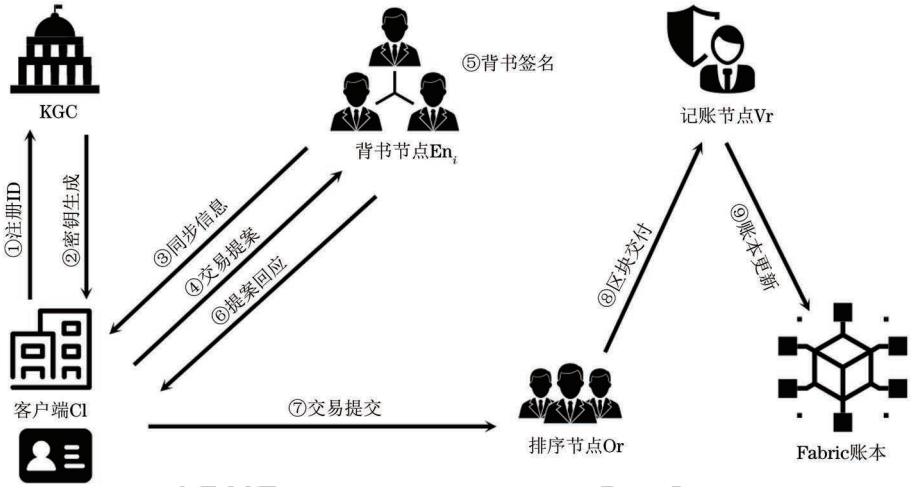

图 2 基于多重签名的区块链交易方案流程

Fig. 2 Blockchain transaction scheme process based on multi-signature

6 性能分析

6.1 计算开销分析

本文主要考虑签名算法和验证算法的计算开销。签名生成的计算开销主要来源于点乘、点加、数乘和数加运算。其中，点乘运算的计算开销为点加运算的约 300 倍，且远高于数乘和数加运算，因此，本文忽略点加、数乘和数加运算消耗。用 n 表示签名者数量，P 表示群 $\mathbb{G}$ 上的点乘运算，B 表示双线性配对运算。表 1 分析了 4 个方案的计算开销。

表 1 多重签名方案的计算开销比较

Table 1 Comparison of computational cost of multi-signature schemes

方案	签名(线下)	签名(线上)	验证	总成本
文献[6]方案	—	nP	$2P$	$(n+2)P$
文献[23]方案	—	$2nP$	$4B$	$2nP+4B$
文献[24]方案	—	$(2n-1)P$	$3nP$	$(5n-1)P$
本文方案	nP	—	$3P$	$(3+n)P$

由表 1 可知，本文方案在在线签名生成部分具有明显优势。因为本文方案中每个用户可在本地线下预计算自己的部分签名贡献，并且支持公开挑战 c，在消息到来后只需进行简单的运算和交互就能完成签名，在聚合签名贡献方面，本文方案在树形结构中进行交互，每个签名者在子节点的基础上进行计算，聚合开销也分担到每个签名者的计算中。在验证阶段，本文方案的 3 次点乘运算成本远小于文献[23-24]方案，并且不随签名者数量的增加而增加。由于生成树结构的引入，使得本文方案和文献[6]方案一样能够实现高扩展性，适合在多用户参与的应用环境下使用。

实验主要从时间开销方面验证本文方案的签名效率，利用 GmSSL 库和 MIRACL 库实现，PC 端配置为 Windows 11 操作系统，Intel Core i5 CPU@2.50 GHz，8 GB RAM，Python 3.9。使用标准的 SM2 曲线和 SM3 哈希算法。所有的签名者被创造在一个树结构中，将树的深度设置为 3 并根据签名者数量选择分支因子。本文使用 CPU 模拟所有运行在树结构上的签名者，均统一部署到一台物理机器上。测试方案的签名算法和验证算法的运行时间，测试结果如图 3、图 4 所示。

由图 3、图 4 可知，本文方案在生成签名和验证时间开销方面具有显著优势。相比文献[23]方案，本文方案没有耗时的双线性配对操作。本文方案在验证时只需要使用聚合后的公钥来进行一次验证即可，而文献[24]方案验证时间会随签名者数量的增多而增多。虽然本文方案和文献[6]方案在签名和验证的效率上差别不大，但是文献[6]方案已被证明无法抵御恶意密钥攻击，存在安全性问题。

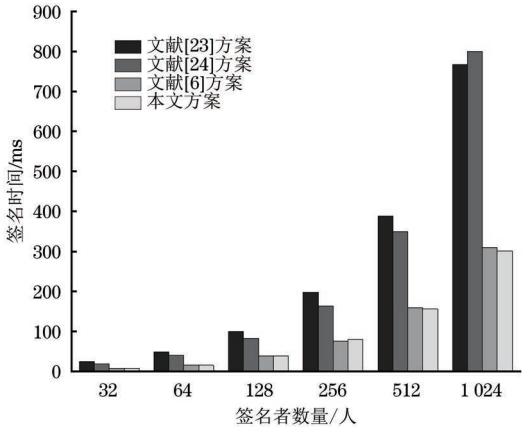

图 3 签名算法时间开销

Fig. 3 Signature algorithm time cost

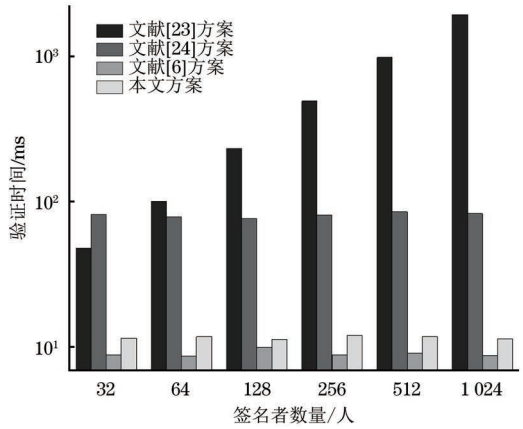

图 4 验证算法时间开销

Fig. 4 Verification algorithm time cost

6.2 通信开销分析

在通信开销方面，本文主要考虑公钥、私钥和多重签名的通信开销。使用$|\mathbb{G}|$和$|\mathbb{Z}_q^*|$分别表示$\mathbb{G}$和$\mathbb{Z}_q^*$中元素的比特长度，其中$|\mathbb{G}|=256$ bit，$|\mathbb{Z}_q^*|=128$ bit，n为签名者数量。表 2 分析了本文方案和文献[6,23-24]方案的通信开销。

表 2 多重签名方案的通信开销比较

Table 2 Comparison of communication cost of multi-signature schemes

方案	私钥	公钥	签名
文献[6]方案	$\|\mathbb{Z}_q^*\|$	$\|\mathbb{G}\|$	$2\|\mathbb{Z}_q^*\|$
文献[23]方案	$2\|\mathbb{G}\|$	$2\|\mathbb{G}\|$	$3n\|\mathbb{G}\|$
文献[24]方案	$2\|\mathbb{Z}_q^*\|$	$2\|\mathbb{G}\|$	$n(\|\mathbb{Z}_q^*\|+\|\mathbb{G}\|)$
本文方案	$\|\mathbb{Z}_q^*\|$	$\|\mathbb{G}\|$	$3\|\mathbb{Z}_q^*\|$

如图 5 所示，本文方案的签名通信开销是常数级别的，文献[6]方案使用树形结构，各个签名者以签名贡献的形式进行聚合生成最终的签名，聚合后的签名大小同单个签名大小一样。随着签名者数量的增多，本文方案相比文献[23-24]方案具有明显优势。

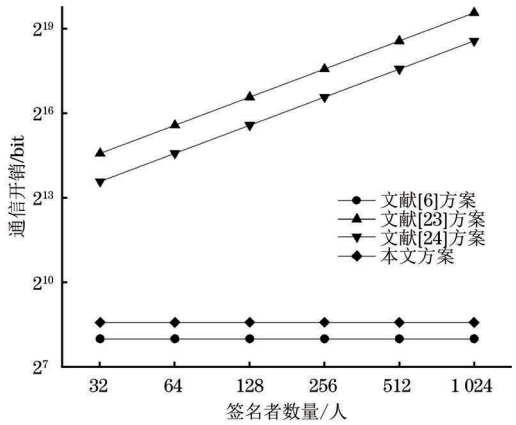

图 5 签名通信开销

Fig. 5 Signature communication cost

7 结束语

本文研究无证书多重签名算法中的安全性和效率问题，基于 SM2 算法设计了一个安全高效的无证书多重签名方案。该方案为抵抗恶意密钥攻击，设计持有证明，对签名者公钥进行验证，引入树形结构改进 SM2 签名算法，设计“线上-线下”的签名过程，实现高效的在线性能和常数级的验证效率。同时，在随机预言机模型下，证明了本文方案可抵御两类攻击者，将安全性归约到椭圆曲线离散对数问题上。此外，本文设计了一种基于多重签名的区块链交易方案，优化了 Fabric 交易流程。最后，通过理论和实验分析，本文方案相较于现有方案具有更好的扩展性和效率。在下一步的工作中，将研究基于区块链智能合约的多重签名算法。

参考文献

[1] NAKAMOTO S. Bitcoin: a peer-to-peer electronic cash system [EB/OL]. [2024-06-05]. http://bitcoin.org/bitcoin.pdf.

[2] National Institute of Standards and Technology. FIPS 186-4: Digital Signature Standard (DSS) [EB/OL]. [2024-06-05]. http://nvlpubs.nist.gov/nistpubs/FI- PS/NIST.FIPS.186-4.pdf.

[3] ITAKURA K. A public-key cryptosystem suitable for digital multisignature[J]. NEC Research and Development, 1983, 71: 1-8.

[4] BARSKI C. Bitcoin for the Befuddled[M]. San Francisco, USA: No Starch Press, 2014.

[5] KOKORIS-KOGIAS E, JOVANOVIC P, GAILLY N, et al. Enhancing Bitcoin security and performance with strong consistency via collective signing[C]//Proceedings of the 25th USENIX Conference on Security Symposium. San Diego, USA: USENIX Association, 2016: 279-296.

[6] SYTA E, TAMAS I, VISHER D, et al. Keeping authorities “honest or bust” with decentralized witness cosigning [C] // Proceedings of the IEEE Symposium on Security and Privacy (SP). Washington D. C., USA: IEEE Press, 2016: 526-545.

[7] AL-RIYAMI S S, PATERSON K G. Certificateless public key cryptography [EB/OL]. [2024-06-05]. https://link.

springer.com/chapter/10.1007/978-3-540-40061-5_29.
[8] 秦艳琳, 吴晓平. 高效的无证书有序多重签名方案[J]. 通信学报, 2013, 34(7): 105-110.
QIN Y L, WU X P. Efficient certificateless sequential multi-signature scheme[J]. Journal on Communications, 2013, 34(7): 105-110. (in Chinese)
[9] 许艳, 黄刘生, 田苗苗, 等. 可证安全的高效无证书有序多重签名方案[J]. 通信学报, 2014, 35(11): 126-131.
XU Y, HUANG L S, TIAN M M, et al. Provably secure and efficient certificateless sequential multi-signature scheme in random oracle model[J]. Journal on Communications, 2014, 35(11): 126-131. (in Chinese)
[10] 国家密码管理局. SM2 椭圆曲线公钥密码算法: GM/T 0003—2012[S]. 北京:中国标准出版社, 2010.
State Cryptography Administration. Public key cryptographic algorithm SM2 based on elliptic curves: GM/T 0003—2012[S]. Beijing: China Standard Press, 2010. (in Chinese)
[11] 杨宏志, 袁凌云, 王舒. 基于 SM2 国密算法优化的区块链设计[J]. 计算机工程与设计, 2021, 42(3): 622-627.
YANG H Z, YUAN L Y, WANG S. Optimized blockchain design based on SM2 algorithm[J]. Computer Engineering and Design, 2021, 42(3): 622-627. (in Chinese)
[12] 沈荣耀, 马利民, 王佳慧, 等. 基于国密 SM2 算法的局部可验证聚合签名算法研究[J]. 信息安全研究, 2024, 10(2): 156-162.
SHEN R Y, MA L M, WANG J H, et al. Research on locally verifiable aggregate signature algorithm based on SM2[J]. Journal of Information Security Research, 2024, 10(2): 156-162. (in Chinese)
[13] 包子健, 何德彪, 彭聪, 等. 基于 SM2 数字签名算法的可否认环签名[J]. 密码学报, 2023, 10(2): 264-275.
BAO Z J, HE D B, PENG C, et al. Deniable ring signature scheme based on SM2 digital signature algorithm[J]. Journal of Cryptologic Research, 2023, 10(2): 264-275. (in Chinese)
[14] BELLARE M, NEVEN G. Multi-signatures in the plain public-key model and a general forking lemma[C]// Proceedings of the 13th ACM Conference on Computer and Communications Security. New York, USA: ACM Press, 2006: 390-399.
[15] BAGHERZANDI A, CHEON J H, JARECKI S. Multisignatures secure under the discrete logarithm assumption and a generalized forking lemma[C]//Proceedings of the 15th ACM Conference on Computer and Communications Security. New York, USA: ACM Press, 2008: 449-458.
[16] MA C S, WENG J, LI Y J, et al. Efficient discrete logarithm based multi-signature scheme in the plain public key model[J]. Designs, Codes and Cryptography, 2010, 54(2): 121-133.
[17] DRIJVERS M, GORBUNOV S, NEVEN G, et al. Pixel: multi-signatures for consensus[EB/OL]. [2024-06-05]. https://eprint.iacr.org/2019/514.pdf.
[18] XIAO Y, ZHANG P, LIU Y H. Secure and efficient multi-signature schemes for Fabric: an enterprise blockchain platform[J]. IEEE Transactions on Information Forensics and Security, 2021, 16: 1782-1794.
[19] DRIJVERS M, EDALATNEJAD K, FORD B, et al. On the security of two-round multi-signatures[C]//Proceedings of the IEEE Symposium on Security and Privacy (SP). Washington D.C., USA: IEEE Press, 2019: 1084-1101.
[20] MAXWELL G, POELSTRA A, SEURIN Y, et al. Simple Schnorr multi-signatures with applications to Bitcoin[J]. Designs, Codes and Cryptography, 2019, 87(9): 2139-2164.
[21] NICK J, RUFFING T, SEURIN Y, et al. MuSig-DN: Schnorr multi-signatures with verifiably deterministic nonces[C]// Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. New York, USA: ACM Press, 2020: 1717-1731.
[22] NICK J, RUFFING T, SEURIN Y. MuSig2: simple two-round Schnorr multi-signatures[EB/OL]. [2024-06-05]. https://link.springer.com/chapter/10.1007/978-3-030-84242-0_8.
[23] 王竹, 杨思琦, 李凤华, 等. 高效可证明安全的无证书有序聚合签名方案[J]. 通信学报, 2022, 43(5): 58-67.
WANG Z, YANG S Q, LI F H, et al. Efficient and provably-secure certificateless sequential aggregate signature scheme[J]. Journal on Communications, 2022, 43(5): 58-67. (in Chinese)
[24] 陈虹, 曹玥, 金海波, 等. 安全高效的无对运算无证书有序多重签名方案[J]. 计算机应用研究, 2023, 40(11): 3402-3407.
CHEN H, CAO Y, JIN H B, et al. Secure and efficient certificateless sequential multi-signature scheme with no pair operation[J]. Application Research of Computers, 2023, 40(11): 3402-3407. (in Chinese)
[25] 尚铭, 马原, 林璟锵, 等. SM2 椭圆曲线门限密码算法[J]. 密码学报, 2014, 1(2): 155-166.
SHANG M, MA Y, LIN J Q, et al. A threshold scheme for SM2 elliptic curve cryptographic algorithm[J]. Journal of Cryptologic Research, 2014, 1(2): 155-166. (in Chinese)
[26] 侯红霞, 杨波, 张丽娜, 等. 安全的两方协作 SM2 签名算法[J]. 电子学报, 2020, 48(1): 1-8.
HOU H X, YANG B, ZHANG L N, et al. Secure two-party SM2 signature algorithm[J]. Acta Electronica Sinica, 2020, 48(1): 1-8. (in Chinese)
[27] 范青, 何德彪, 罗敏, 等. 基于 SM2 数字签名算法的环签名方案[J]. 密码学报, 2021, 8(4): 710-723.
FAN Q, HE D B, LUO M, et al. Ring signature schemes based on SM2 digital signature algorithm[J]. Journal of Cryptologic Research, 2021, 8(4): 710-723. (in Chinese)
[28] 阮鸥, 陈吉晨, 毛浩. 一种高效的 SM2 数字签名批量验证算法[J]. 计算机工程与科学, 2021, 43(7): 1236-1242.
RUAN O, CHEN J C, MAO H. An efficient batch verification algorithm for SM2 signatures[J]. Computer Engineering & Science, 2021, 43(7): 1236-1242. (in Chinese)
[29] LIU D N, YUAN Y Y, XUE T F, et al. An efficient batch verification scheme for SM2 signatures[C]//Proceedings of the 8th International Conference on Dependable Systems and Their Applications (DSA). Washington D.C., USA: IEEE Press, 2021: 208-212.
[30] ZHANG Y L, WANG H Y, WANG Y C, et al. Signature scheme based on the SM2 algorithm in Fabric[C]// Proceedings of the International Conference on Networking and Network Applications (NaNA). Washington D.C., USA: IEEE Press, 2021: 443-448.
[31] 国家密码管理局. SM3 密码杂凑算法: GM/T 0004—2012[S]. 北京: 中国标准出版社, 2012.
State Cryptography Administration. SM3 cryptographic hash algorithm: GM/T 0004—2012[S]. Beijing: Standards Press of China, 2012. (in Chinese)
[32] BELLARE M, ROGAWAY P. Random oracles are practical: a paradigm for designing efficient protocols[C]// Proceedings of the 1st ACM Conference on Computer and Communications Security. New York, USA: ACM Press, 1993: 62-73.
[33] RISTENPART T, YILEK S. The power of proofs-of-possession: securing multiparty signatures against rogue-key attacks[EB/OL]. [2024-06-05]. https://rist.tech.cornell.edu/papers/pkreg.pdf.

文字编辑 吴云芳
栏目编辑 赖玉玲