软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2023,34(2):915−931 [doi: 10.13328/j.cnki.jos.006505]

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

抗差分故障攻击的两方协同 EdDSA 签名方案*

严都力[1], 谢 敏[2], 赵艳琦[3], 王文发[1], 禹 勇[2]

[1](延安大学 数学与计算机科学学院, 陕西 延安 716000)
[2](陕西师范大学 计算机科学学院, 陕西 西安 710119)
[3](西安邮电大学 网络空间安全学院, 陕西 西安 710121)
通信作者: 禹勇, E-mail: yuyongxy@163.com

摘 要: 以区块链为底层技术的比特币、Libra 等密码货币掀起了数字经济的浪潮. 密码货币采用数字签名保证交易的可验证性和完整性, 其中签名私钥确保了货币资产的所有权. 若签名私钥丢失或被盗, 货币资产的安全将受到严重威胁. 相比于椭圆曲线数字签名算法 ECDSA, 基于爱德华曲线的数字签名算法 EdDSA 具备运算速度更快、密钥与签名空间更小等优势, 被用于 Libra 交易单的签名. 但因其是确定性签名, 容易遭受差分故障攻击, 造成密钥丢失或泄漏. 如何抵抗这一种攻击, 并设计可证明安全的 EdDSA 签名是一个挑战. 首先定义了抗差分故障攻击的数字签名方案需满足的安全性质, 利用差分故障攻击技术对 EdDSA 签名算法进行了分析, 提出了抗差分故障攻击的 EdDSA 签名方案, 并证明了方案满足存在不可伪造性和抗差分故障攻击性; 为了降低签名私钥泄漏风险, 借助 Paillier 同态加密技术, 设计了抗差分故障攻击的两方协同 EdDSA 签名方案, 并基于通用可组合安全模型 (universally composable, UC) 证明了方案的安全性; 最后, 对两方协同 ECDSA 签名算法与抗差分故障攻击的两方协同 EdDSA 签名算法计算复杂度分析与算法执行效率测试, 验证了方案的有效性.

关键词: 区块链; 数字签名; 差分故障攻击; 协同签名
中图法分类号: TP309

中文引用格式: 严都力, 谢敏, 赵艳琦, 王文发, 禹勇. 抗差分故障攻击的两方协同EdDSA签名方案. 软件学报, 2023, 34(2): 915–931. http://www.jos.org.cn/1000-9825/6505.htm
英文引用格式: Yan DL, Xie M, Zhao YQ, Wang WF, Yu Y. Two-party EdDSA Signature Scheme Against Differential Fault Attack. Ruan Jian Xue Bao/Journal of Software, 2023, 34(2): 915–931 (in Chinese). http://www.jos.org.cn/1000-9825/6505.htm

Two-party EdDSA Signature Scheme Against Differential Fault Attack

YAN Du-Li[1], XIE Min[2], ZHAO Yan-Qi[3], WANG Wen-Fa[1], YU Yong[2]

[1](School of Mathematics and Computer Science, Yan'an University, Yan'an 716000, China)
[2](School of Computer Science, Shaanxi Normal University, Xi'an 710119, China)
[3](School of Cyberspace Security, Xi'an University of Posts and Telecommunications, Xi'an 710121, China)

Abstract: Cryptocurrencies such as Bitcoin and Libra based on blockchain technology have set off a wave of digital economy, which can ensure the verifiability and integrity of transactions through digital signatures, in which the private key ensures the ownership of currency assets, if the private key was lost or stolen, the security of cryptocurrency assets will be significantly threatened. Compared with elliptic curve digital signature algorithm (ECDSA), Edwards curves digital signature algorithm (EdDSA) has the advantages of faster calculation speed, smaller key and signature space, and is widely used in the signature of Libra transactions. However, as a deterministic signature algorithm, it is vulnerable to differential fault attacks resulting in key loss and leakage. It is a challenge that how to resist this kind of

* 基金项目: 国家自然科学基金 (61872229, U19B2021); 教育部 2020 年度区块链核心技术战略研究项目 (2020KJ010301); 陕西省重点研发计划 (2020ZDLGY09-06, 2021ZDLGY06-04)
收稿时间: 2021-08-07; 修改时间: 2021-09-07; 采用时间: 2021-10-09; jos 在线出版时间: 2022-11-30
CNKI 网络首发时间: 2022-12-01

attack and design a provably secure EdDSA signature. Therefore, we firstly define the security properties are firstly defined that the digital signature scheme against differential fault attacks that must be meet, and differential fault attack technology is utilized to cryptanalyze the EdDSA signature algorithm, and an EdDSA signature scheme that resists differential fault attacks is proposed, and it is proved that the scheme satisfies the existence of unforgeable under adaptive selection message attack (EUF-CMA) and resistance to differential fault attack. In order to reduce the risk of signature private key leakage, with the help of Paillier homomorphic encryption technology, we design a two-party cooperative EdDSA signature scheme against differential fault attack is designed, and prove the security of the scheme based on the universally composable (UC) security model is proved. Finally, we implement the two-party cooperative ECDSA signature algorithm and the two-party cooperative EdDSA signature algorithm against differential fault attack are implemented, and the implementation demonstrates that the effectiveness of the proposed scheme.

Key words: blockchain; digital signature; differential fault attack; collaboration signature

近年来, 以区块链为底层技术的比特币、以太币等密码货币掀起了数字经济的浪潮[1]. 2019 年 6 月, 全球最大的社交平台 Facebook 公司推出了一种基于联盟链的稳定密码货币 Libra[2]. Libra 稳定币与比特币类似, 其具有公开透明、去中心化等特征, 但相比于比特币、以太币等数字货币, Libra 由 Libra 协会承担类似中央银行的职责, 负责 Libra 的发行与管理, 其以真实资产储备为担保, 显著改进了比特币存在币值波动较大的问题[3,4]. Libra 目前的交易速度约每秒 1 000 笔交易, 相比于比特币交易单 ECDSA 签名算法, Libra 采用运算效率更高的 EdDSA 签名算法保证交易单有效性, 其签名私钥确保了 Libra 货币资产的所有权. 若签名密钥被盗或丢失, Libra 资产安全将受到严重威胁.

EdDSA 签名算法作为一种确定性签名, 其容易遭受差分故障攻击, 导致签名私钥泄露[5]. 差分故障攻击[5]是侧信道攻击的一种, 攻击者利用电磁辐射、激光等物理手段在加密、签名过程中注入故障, 迫使密码算法执行过程中产生运算错误, 攻击者可以获得同一个消息在密钥作用下的正确密文, 以及诱发故障后产生的错误密文, 最后依据正确密文与错误密文推算出密码系统的密钥信息. Boneh 等人[6]最早提出了“故障攻击”的思想, 并采用故障攻击方法成功破解了 RSA 公钥加密算法. 随后, Biham 等人[7]提出了差分故障攻击的概念, 并采用差分故障攻击方法对 DES 算法分析, 将故障注入 DES 内部状态的右半部分某个随机比特, 成功破解了经典的分组密码 DES 算法. 2013 年, Biehl 等人[8]提出了一种差分故障攻击 ECC 密码系统的方法, 首先将故障点嵌入安全椭圆曲线点中, 使错误点位于弱椭圆曲线上, 然后通过分析弱椭圆曲线来获得安全椭圆曲线的密钥信息. 许盛伟等人[9]提出了一种改进的差分故障攻击方法, 主要针对椭圆曲线点乘算法失效问题进行攻击. 此外, 金雨璇等人[10]提出了一种针对国密 SM4 算法的差分故障攻击, 利用面向字节的随机故障模型和差分分析技术, 通过在 SM4 算法的第 28 轮第 4 个数据存储器中诱发一次单比特故障, 最终利用穷举搜索技术恢复出了 128 比特的密钥.

随着攻击手段的不断更新, 各类防御故障攻击的技术也在随之更新. Ambrose 等人[11]分析了确定性签名算法遭受差分故障攻击导致密钥被恢复的安全威胁, 并给出了增强哈希、添加噪声等抗差分故障攻击的对策. 朱磊等人[12]针对激光故障注入而设计了光检测电路, 一定程度防止了攻击者开盖注入的故障攻击. 段晓毅等人[13]针对电压毛刺故障注入设计了电压检测电路, 防止攻击者对电源引脚注入故障等, 也可以通过伪重复计算、逆运算等方式检测加密结果是否被正确的. 上述差分故障攻击的研究主要涉及到对称密码体制、国产密码和椭圆曲线密码体制, 针对密码货币中交易单签名算法的差分故障攻击及防御攻击技术还有待需进一步探索研究.

本文主要工作如下.

(1) 定义了抗差分故障攻击签名方案需满足的安全性质, 提出了抗差分故障攻击的 EdDSA 签名方案, 并对其不可伪造性和抗差分故障攻击安全性进行分析.

(2) 借助 Paillier 同态加密技术, 设计了抗差分故障攻击的两方协同 EdDSA 签名方案, 并在 UC 安全模型下证明了方案的安全性.

(3) 将设计的抗差分故障攻击的两方协同 EdDSA 签名算法与已有签名算法进行效率比较, 结果显示, 抗差分故障攻击的两方协同 EdDSA 签名算法在计算复杂度与算法执行效率方面都具备优势.

本文第 1 节回顾 EdDSA 签名算法、Paillier 同态加密和零知识证明等预备知识. 第 2 节定义抗差分故障攻击的签名方案需满足的安全性质. 第 3 节提出抗差分故障攻击的 EdDSA 签名方案, 并对其安全性进行分析. 第 4 节

设计抗差分故障攻击的两方协同 EdDSA 签名方案, 并分别对抗差分故障攻击的两方协同 EdDSA 签名方案的安全性及性能进行分析. 第 5 节对本文工作进行总结.

1 基础知识

1.1 EdDSA 签名

EdDSA 签名[14]是基于爱德华椭圆曲线 (Edwards25519) 的确定性签名算法, 曲线的基本参数记为 $PP=(q, F_q, c,d,B, n,H_1,H_2)$, 其中 $q=2^{255}-19$ 为 F_q 特征, 参数 $c,d\in F_q$ 确定了爱德华曲线 $E_{c,d}: cx^2+y^2=1+dx^2y^2$; 定义点 B 为曲线的基点 $B\in E_{c,d}(F_q)$, 素数 n 表示基点 B 的阶, 满足 $nB=0$ 且 $2^3n=\#E_{c,d}$; 令 $H_1:\{0,1\}^{\kappa}\to\{0,1\}^n$ 与 $H_2:\{0,1\}^*\to Z_n$ 为密码学哈希函数. 任意选取参数 b' 使其满足 $2^{b'-1}>q$, 其中 Edwards25519 曲线中 b' 固定取值为 256. 定义 $SS_{\text{EdDSA}}=(KeyGen, Sign, Verify)$ 为 EdDSA 签名方案. 签名算法描述具体如下.

(1) 密钥生成算法 ($KeyGen$)

① 任意选取 b' 位的随机字符串 α 作为私钥, 计算 $H_1(\alpha)=(h_0,h_1,\ldots,h_{2b'-1})$, 令 $a=(h_0,h_1,\ldots,h_{b'-1})$, $b=(h_{b'},h_{b'+1},\ldots,h_{2b'-1})$;

② 利用 $a=(h_0,h_1,\ldots,h_{b'-1})$ 计算整数 $x=(2^{b'-2}+\sum_{i=3}^{b'-3}2^i\cdot h_i) \bmod n$ 作为签名辅助私钥;

③ 计算签名公钥 $A=xB$.

(2) 签名算法 ($Sign$)

① 计算签名消息 m 的哈希值 $e=H_2(m)$;

② 计算临时密钥 $r=H_2(b,e) \bmod n$;

③ 计算 $R=rB$, 若 R 为无穷远点, 则返回①;

④ 计算 $h=H_2(R,A,e) \bmod n$;

⑤ 计算 $s=(r+hx) \bmod n$, 若 $s=0$ 返回到①;

⑥ 签名者对消息 m 的签名为 $\sigma=(R,s)$, 发送签名 σ 至验证者.

(3) 验证算法 ($Verify$)

① 验证者计算 $h=H_2(R,A,e) \bmod n$;

② 验证者验证等式 $sB=R+hA$ 是否成立, 若成立, 则签名 $\sigma=(R,s)$ 为消息 m 的有效签名.

1.2 Paillier 同态加密

Paillier 公钥加密算法[15]由密钥生成、加密算法和解密算法这 3 部分组成. $Enc_{pk}(\cdot)$ 表示用公钥 pk 加密运算, $Dec_{sk}(\cdot)$ 表示用私钥 sk 解密运算, Paillier 加密算法具体描述如下.

(1) 密钥生成 (Gen)

任意选取 2 个素数 p 和 q, 满足 $gcd(pq,(p-1)(q-1))=1$, 计算 $N=p\times q$ 和 $\lambda=lcm(p-1,q-1)$, 其中 lcm 表示计算最小公倍数公式;

任意选取 $g\in Z_{N^2}^*$, 计算 $\mu=\phi(N)^{-1} \bmod N$;

得到加密公钥为 $pk=(N,g)$, 解密私钥为 $sk=(\lambda,\mu)$.

(2) 加密算法 (Enc)

利用公钥 (N,g) 对消息 m 进行加密, 任意选取一个随机数 $r\in Z_N^*$, 计算密文 $c=Enc_{pk}(m)=g^m r^N(\bmod N^2)$, 其中 m 表示加密的消息.

(3) 解密算法 (Dec)

利用解密私钥 (λ,μ) 对密文 c 进行解密得到明文 m, $m=Des_{sk}(c)=L(c^{\lambda} \bmod N^2)\cdot\mu \bmod N$, 其中函数 L 被定义为 $L(u)=(u-1)/N$.

Paillier 加密同态性质: 对于消息 $m_1,m_2\in Z_N$, 任意选取整数 $r_1,r_2\in Z_N^*$ 进行加密, 可以得到密文 $c_1=Enc_{pk}(m_1)$,

$c_2 = Enc_{pk}(m_2)$, 满足以下关系.

① $Enc_{pk}(m_1)\cdot Enc_{pk}(m_2) \bmod N^2 = Enc_{pk}(m_1+m_2 \bmod N)$;

② $Enc_{pk}(m_1)^{m_2} \bmod N^2 = Enc_{pk}(m_1 m_2 \bmod N)$;

③ $Enc_{pk}(m_1)^k \bmod N^2 = Enc_{pk}(km_1 \bmod N)$.

本文中定义同态加法运算为$\oplus$, 定义同态乘符号为$\odot$, 例如$c_1 \oplus c_2 = Enc_{pk}(m_1+m_2)$, $k \odot c_1 = Enc_{pk}(m_1)^k$.

1.3 承诺协议

承诺协议[15]包括两个参与方: 发送方和接收方. 发送方暂时以隐藏的方式向接收方承诺一个值, 承诺后不再对该值做任何修改. 数字承诺协议包括初始化、承诺阶段和揭示阶段.

(1) 系统初始化$pp \leftarrow Setup(1^\lambda)$: 以安全参数$\lambda$作为输入, 输出公开参数$pp$.

(2) 承诺阶段$commit \leftarrow Commit_{pp}(m)$: 输入消息$m$, 输出承诺值$commit$.

(3) 揭示阶段$decommit \leftarrow Decommit(m, commit)$: 输入消息$m$和承诺$commit$, 输出打开证据$decommit$.

数字承诺协议满足隐藏性和绑定性两个性质.

(1) 隐藏性: 对任意消息$m_0, m_1 \in M$, 有$\{Commit_{pp}(m_0)\} \approx \{Commit_{pp}(m_1)\}$. 即在承诺之后, 接收者无法获得发送者所承诺消息的任何信息.

(2) 绑定性: 给定公开参数pp, 输出承诺$commit$两个正确打开证据$(m_0, decommit_0)$和$(m_1, decommit_1)$是不可行的, 其中$m_0 \neq m_1$. 即关于消息m的承诺生成后, 发送方不能将已承诺的消息打开为另一不同消息m'.

1.4 零知识证明

零知识证明[16]包括两个参与方: 证明者P和验证者V. 对于陈述$x \in L$与证据w满足关系R使得$(x,w) \in R$. 一个非交互式零知识证明$NIZK\{x|(x,w) \in R\}$包含系统生成、证明和验证这 3 个算法. 具体算法如下.

(1) 系统生成$(CRS) \leftarrow Setup(1^\lambda)$: 以安全参数$\lambda$作为输入, 输出公共参考串$CRS$;

(2) 证明$(\pi) \leftarrow Prove(CRS, x, w)$: 证明者$P$生成对陈述$x \in L$的证明$\pi$;

(3) 验证$0/1 \leftarrow Verify(CRS, x, \pi)$: 验证者$V$验证证明$\pi$是否有效, 若有效, 输出 1; 否则, 输出 0.

零知识证明协议满足完备性、可靠性和零知识性.

(1) 完备性: 如果P向V证明知道某一秘密, 则V将接受P的证明;

(2) 可靠性: 如果P能以一定的概率使V相信P的证明, 则P知道相应的秘密;

(3) 零知识性: 如果P发送给V的证明是真的, V在不违背协议的前提下, 无论V采取任何方法, 除了接收到P给出的证明外, V无法获取有关P的其他任何信息.

1.5 通用可组合安全模型

通用可组合 (universally composable, UC) 安全模型[17]是一种依据协议现实执行环境和理想执行环境的不可区分方法来定义安全性的框架, 简称为 UC 框架. UC 框架采用模块化设计思想, 只要每个单一协议满足 UC 安全, 就可以保证与其他协议组合、并行运行的复杂协议的安全性[18]. UC 安全框架主要采用“模拟”证明方法, 其由现实模型、理想模型和$\mathcal{F}$-混合模型这 3 个模型组成[18], 本文第 4 节证明协议的安全性主要采用 UC 安全框架的$\mathcal{F}$-混合模型对方案进行安全性证明. 该混合模型是在现实模型的基础上, 增加了使其在执行过程中调用理想功能函数$\mathcal{F}$的能力, $\mathcal{F}$可看做可信第三方. $\mathcal{F}$-混合模型是介于现实模型和理想模型之间的特殊协议执行模式, 称该混合协议为$\mathcal{F}$-hybrid 协议[19].

$\mathcal{F}$-hybrid 协议运行实例中, 参与者可以多次调用理想功能函数$\mathcal{F}$, 参与者将输入交给$\mathcal{F}$, 并从$\mathcal{F}$获得输出, 每一个$\mathcal{F}$的实例通过会话标识sid进行区分. 本文设计的两方协同 EdDSA 签名算法中参与方之间采用理想承诺函数$\mathcal{F}_{com}$[20]、理想零知识函数$\mathcal{F}_{zk}$和承诺的非交互式零知识理想函数$\mathcal{F}_{com\text{-}zk}$[21]这 3 个理想功能函数[18], 其中$P_b$ ($b \in \{1,2\}$) 表示参与方. 理想零知识函数之间满足两种特定的关系R, 下面分别定义上述 3 种理想函数和两种特定的关系R.

(1) 理想承诺函数$\mathcal{F}_{com}$

① $\mathcal{F}_{com}$ 从参与方 P_b ($b\in\{1,2\}$) 接收到 $(commit, sid, x)$, 记录 (sid, b, x) 并发送 $(receipt, sid)$ 给参与方 P_{3-b}, 如果 $(commit, sid, *)$ 已被存储, 则忽略此消息.

② $\mathcal{F}_{com}$ 从参与方 P_b 接收到 $(decommit, sid)$, 如果 (sid, b, x) 已被记录, 则发送 $(decommit, sid, x)$ 给参与方 P_{3-b}.

(2) 理想零知识函数 $\mathcal{F}_{zk}^{R}$

① $\mathcal{F}_{zk}^{R}$ 从参与方 P_b ($b\in\{1,2\}$) 接收到 $(prove, sid, x, w)$, 如果 $(x, w)\notin R$ 或 sid 已经被使用过, 则忽略此消息; 否则, $\mathcal{F}_{zk}^{R}$ 发送 $(proof, sid, x)$ 给参与方 P_{3-b}. 其中零知识证明满足 $((x, w), \lambda)\rightarrow(\lambda, R(x, w))$, λ 表示空字符串, 对于承诺 $x\in L$、证据 w 和关系 R 满足 $(x, w)\in R$.

(3) 满足关系 R 的承诺非交互式零知识理想函数 $\mathcal{F}_{com-zk}^{R}$, 其满足以下定义

① $\mathcal{F}_{com\text{-}zk}^{R}$ 从参与方 P_b ($b\in\{1,2\}$) 接收到 $(com\text{-}prove, sid, x, w)$, 如果 $(x, w)\notin R$ 或 sid 已经被使用过, 则忽略此消息; 否则, $\mathcal{F}_{com\text{-}zk}^{R}$ 保存 (sid, b, x) 并将 $(proof\text{-}receipt, sid)$ 发送给参与方 P_{3-b}.

② $\mathcal{F}_{com\text{-}zk}^{R}$ 从参与方 P_b 接收到 $(decom\text{-}proof, sid)$, 如果 (sid, b, x) 已被保存, 则 $\mathcal{F}_{com\text{-}zk}^{R}$ 将 $(decom\text{-}proof, sid, x)$ 发送给参与方 P_{3-b}.

(4) 零知识理想函数 $\mathcal{F}_{zk}$ 是基于以下两种关系 R, R 满足以下定义

① Paillier 加密算法公钥正确性证明关系[22]:

$$R_p=\{(N, (p.q))|N=p\cdot q\text{且}p, q\text{是素数}\}.$$

② 椭圆曲线上离散对数知识证明关系[22]:

$$R_{DL}=\{(E, G, q, Q, x)|Q=x\cdot G\}.$$

此外, 本文还需要证明在给定对应密文中加密的值是椭圆曲线点的离散对数. 该语句的零知识证明不是知识证明, 不满足上述定义的关联关系 R, 该语言定义如下:

$$T_{PDL}=\{(c, pk, Q, G, G, n)|\exists x_1\in Z_n, \text{满足}c=Enc_{pk}(x_1)\text{且}Q_1=x\cdot G\},$$

其中, pk 是给定的 Paillier 公钥, sk 是对应的私钥. T_{PDL} 提供了一个高效的零知识证明, 该证明在 Paillier 加密和椭圆曲线群之间架起了桥梁. T_{PDL} 不是知识的证明, 无法借助 $\mathcal{F}_{zk}$ 混合模型证明, 而是直接采用标准的零知识证明方式证明, 详细证明过程见参考文献 [23].

2 安全模型

抗差分故障攻击的两方协同 EdDSA 签名需满足存在性不可伪造和抵抗差分故障攻击的安全性质, 若敌手具备差分故障攻击分析能力, 却无法通过已知签名信息提取签名私钥, 则表明该签名方案具备抵抗差分故障攻击安全性. 下面对抗差分故障攻击的两方协同 EdDSA 签名的存在性不可伪造和密钥提取的安全性分别进行形式化定义.

定义 1. 定义数字签名方案 $\pi=(KeyGen, Sign, Verify)$, 对于任意概率多项式时间 (probabilistic polynomial time, PPT) 敌手 $\mathcal{A}$, $\mathcal{A}$ 与挑战者 C 执行游戏记为 $Expt\text{-}Sign_{\mathcal{A},\pi}(1^{\lambda})$, 若 $\mathcal{A}$ 在 $Expt\text{-}Sign_{\mathcal{A},\pi}(1^{\lambda})$ 游戏中成功伪造签名的概率是可忽略的. 则签名方案 π 是满足适应性选择消息攻击下存在性不可伪造 (existential unfogeability adaptive chosen-message attack, EUF-CMA) 安全性, $Expt\text{-}Sign_{\mathcal{A},\pi}(1^{\lambda})$ 游戏具体描述如下.

(1) 初始化: 挑战者 C 生成系统参数, 执行密钥生成算法 $KeyGen(1^{\lambda})$ 生成签名密钥对 (pk, sk), C 将公钥 pk 发送给敌手 $\mathcal{A}$.

(2) 签名询问: 敌手 $\mathcal{A}$ 选择任意消息 $m_i\in\{m_1,\ldots, m_\ell\}$ 进行签名询问, 挑战者 C 计算签名 $\sigma_i=Sign(m_i)$ 返回给 $\mathcal{A}$.

(3) 签名伪造: 敌手 $\mathcal{A}$ 返回一个伪造的签名 (m', σ'), 如果 $Verify(pk, m', \sigma')=1$ 且 $m'\notin\{m_1,\ldots, m_\ell\}$, 则 $\mathcal{A}$ 伪造签名成功, 否则失败.

对于上述游戏, 任意 PPT 敌手 $\mathcal{A}$, 存在一个可忽略的函数 ε, 使得对于每个 λ 都有:

$$\Pr[Expt\text{-}Sign_{\mathcal{A},\pi}(1^{\lambda})=1]\leqslant\varepsilon(\lambda).$$

定义 2. 令签名方案 Π 是由签名方案 π 派生的安全两方协同签名方案. 那么对于任意 PPT 敌手 $\mathcal{A}$ 和协同签名

参与方 P_b, 其中 $b \in \{1,2\}$, 设 $\mathcal{A}$ 与挑战者 C 执行游戏 $Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^{\lambda})$, 若 $\mathcal{A}$ 在 $Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^{\lambda})$ 游戏中成功伪造两方协同签名的概率是可忽略的. 则两方协同签名方案 Π 是满足 EUF-CMA 安全目标. $Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^{\lambda})$ 游戏具体描述如下.

(1) 初始化: 挑战者 C 生成系统参数, 执行密钥生成算法 $KeyGen(1^{\lambda})$ 生成签名密钥对 (pk, sk), C 将公钥 pk 发送给敌手 $\mathcal{A}$.

(2) 签名询问: 敌手 $\mathcal{A}$ 可以分别控制腐化参与方 P_b, 其中 $b \in \{1,2\}$, 选择任意消息 $m_i \in \{m_1, \ldots, m_\ell\}$ 进行签名询问, 挑战者 C 计算签名 $\sigma_i = Sign(m_i)$ 返回给 $\mathcal{A}$.

(3) 签名伪造: 敌手 $\mathcal{A}$ 在不访问签名预言机 O 情况下, 伪造一组签名 (m', σ) 且 $Verify(pk, m', \sigma') = 1$, 则 $\mathcal{A}$ 伪造协同签名成功, 否则失败. 其中 $m' \notin \{m_1, \ldots, m_{q_s}\}$, $Verify$ 表示协同签名验证算法, 与 π 签名方案验证算法相同.

对于上述游戏, 定义任意 PPT 敌手 $\mathcal{A}$, 存在一个可忽略的函数 ε, 使得对于每个 λ 都有:

$$\Pr[Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^{\lambda}) = 1] \leqslant \varepsilon(\lambda).$$

在上述实验中, 定义 $O_b(\cdot,\cdot)$ 为预言机, 诚实的参与者为 P_{3-b}, 敌手每次可以腐化两个参与者中的一个. 然后, 敌手与参与者 P_{3-b} 进行交互, 对任意的消息 m 协同生成一个合法的签名. 在该协议中, 密钥生成过程只执行一次, 签名生成过程可以执行多次. 敌手可以向 $O_b(\cdot,\cdot)$ 进行问询, $O_b(\cdot,\cdot)$ 输入参数分别是会话标识以及请求的消息或者下一条输入消息. $O_b(\cdot,\cdot)$ 预言机的执行过程如下.

(1) 当预言机第 1 次收到请求为 (0, 0), 预言机初始化一台机器 M, 其中 M 运行协议 Π 的两方密钥生成阶段 P_{3-b} 的指令. 如果参与者 P_{3-b} 发送了密钥生成阶段的第 1 条消息, 则该消息将作为预言机的回复.

(2) 当预言机收到请求为 $(0, m)$, 如果两方密钥生成阶段未完成, 则预言机将消息 m 作为下一个输入消息交给 M, 并作为 M 的回复. 如果此时密钥生成阶段已经完成, 则回复 $\perp$, 并中断此过程.

(3) 当预言机收到请求为 (sid, m), 如果 $sid \neq 0$, 但 M 密钥生成阶段还未完成, 则预言机返回 $\perp$.

(4) 当预言机收到请求为 (sid, m), 如果两方密钥生成阶段已经完成, 且此次会话标识 sid 是第 1 次请求, 则预言机调用 M_{sid} 执行参与方 P_{3-b} 在协议 Π 的指令, 其输入信息为 (sid, m). 在密钥生成阶段完成时, 使用密钥共享和 M 存储的任何状态初始化机器 M, 如果参与方 P_{3-b} 发送了签署协议的第 1 条消息, 则消息作为预言机的回复.

(5) 当预言机收到请求为 (sid, m), 两方密钥生成阶段已完成, 且 sid 不是第 1 次请求时, 即在之前的问询中已经请求过该序号, 则预言机将消息 m 发送给 M_{sid} 作为收到的消息, 并且返回 M_{sid} 的输出作为下一条消息.

上述过程中, 敌手能够腐化参与方 P_b, 其中 $b \in (1, 2)$, 敌手与协议中的诚实参与方 P_{3-b} 交互. 如果敌手能成功伪造一个未被问询的消息签名, 那么敌手获胜.

密钥提取 ($KeyExtract$): 通常情况下, 攻击者若想从正确签名中分析获得签名私钥是计算上不可行的. 假设存在差分故障攻击者利用电磁辐射、激光等物理手段, 在签名过程的任意时刻或任意位置注入故障, 针对同一消息 m 可以获得正确签名 σ 和错误签名 σ', 攻击者利用 σ, σ' 和已知签名参数建立关系, 可以有效推导出签名私钥 sk, 其定义如下.

定义 3. 令 $\mathcal{A}$ 是一个 PPT 敌手 (差分故障攻击者), $\pi = (KeyGen, Sign, Verify)$ 记为签名方案, $\mathcal{A}$ 与挑战者 C 执行游戏记为 $Expt\text{-}KeyExtract_{\mathcal{A},\pi}(1^{\lambda})$, 敌手 $\mathcal{A}$ 在时间 t 内最多进行 q_s 次签名询问, 若 $\mathcal{A}$ 在 $Expt\text{-}KeyExtract_{\mathcal{A},\pi}(1^{\lambda})$ 游戏中成功提取签名私钥的优势 $Adv^{KeyExtract}_{\mathcal{A},\pi}(\lambda)$ 是可忽略的, 则该签名方案具备抗差分故障攻击的安全性. $Expt\text{-}KeyExtract_{\mathcal{A},\pi}(1^{\lambda})$ 游戏具体描述如下.

(1) 初始化: 挑战者 C 生成系统参数, 执行密钥生成算法 $KeyGen(1^{\lambda})$ 生成签名密钥对 (pk, sk), C 将公钥 pk 发送给敌手 $\mathcal{A}$; 敌手 $\mathcal{A}$ 具备差分故障攻击分析能力, 可以在任意时刻任意位置将故障注入到签名算法中;

(2) 签名询问: $\mathcal{A}$ 选择任意消息 m_i 进行签名询问, 第 1 次问询后 C 将 m_i 的正确签名 σ 返回给 $\mathcal{A}$, 然后敌手 $\mathcal{A}$ 将故障注入签名算法的某个特定组件上, 再一次对消息 m_i 进行问询, C 将 m_i 的错误签名 σ' 应答给 $\mathcal{A}$;

(3) 密钥提取: $\mathcal{A}$ 通过询问得到消息签名对 (m_i, σ_i) 和 (m_i, σ'_i), 利用签名公钥 pk 和已知签名参数计算出 sk'. 若密钥提取成功, 即 $sk' = sk$, 输出 $b = 1$, 否则, 输出 $b = 0$.

敌手 $\mathcal{A}$ 成功提取密钥的优势定义为安全参数 λ 的函数:

$$Adv_{\mathcal{A},\pi}^{KeyExtract}(\lambda)=\Pr[KeyExtract_{\mathcal{A},\pi}(\lambda)=1].$$

3 抗差分故障攻击的 EdDSA 签名

本节简要回顾差分故障攻击 EdDSA 签名算法, 随后利用随机噪声对 EdDSA 签名算法改进, 提出了抗差分故障攻击的 EdDSA 签名算法, 最终对抗差分故障攻击 EdDSA 签名算法安全性进行分析, 并对其性能进行评估.

3.1 差分故障攻击 EdDSA 签名

本节简要回顾文献 [11] 对 EdDSA 签名算法中基点 $B\in E_{c,d}(F_q)$ 执行的差分故障攻击. EdDSA 签名算法实现的某个阶段生成元或基点 B 是公共的, 基点 B 的作用是执行确定性随机数的椭圆曲线标量乘法. 如果在基点 B 中引入故障, 将导致该值不再是椭圆曲线上的有效点, 对于同一个消息 m 可以获得一个有效的签名值 (R,s) 和一个无效的签名值 (R',s'). 攻击者试图利用获取的 (R,s) 和 (R',s') 提取签名私钥, 建立方程组如下:

$$\begin{cases}(R,s)=(rB,r+hx \bmod n)\\(R',s')=(rB',r+h'x \bmod n)\end{cases}.$$

上述方程式中 $h=H_2(R,A,e)$, 其中 R,A,e 已知, 则很容易计算 h, 同理 h' 也已知, 根据方程组变形得到:

$$s-s'=x(h-h') \bmod n.$$

综上, 利用上述方法可提取出签名私钥 x. 同理, 若签名组件中公钥 A 、确定性随机数组件 r 和 h 遭受攻击, 同样可以提取到签名私钥 x, 具体分析过程见参考文献 [11].

3.2 抗差分故障攻击的 EdDSA 签名算法

为了抵抗 EdDSA 签名算法遭受差分故障攻击, 签名者计算 $r=H_2(d,e) \bmod n$ 时, 引入随机数 $k\in Z_n^*$, 即计算 $r=H_2(d,e,k) \bmod n$, 使得 EdDSA 签名算法具备随机性, 针对同一消息 m 进行签名, 得到的签名将是随机的, 攻击者无法利用上述差分故障攻击能力来分析获取签名私钥 x, 使得 EdDSA 签名算法具备了抵抗差分故障攻击的特性.

定义 $\pi=(KeyGen,Sign,Verify)$ 为抗差分故障攻击的 EdDSA 签名方案, 参数设置与第 1.1 节中 EdDSA 签名算法参数设置相同, 令 $PP=(q,F_q,c,d,B,n,H_1,H_2)$, 抗差分故障攻击的 EdDSA 签名算法描述具体如下.

(1) 密钥生成算法 ($KeyGen$)

① 随机选取 b' 位的整数 α 作为私钥, 计算 $H_1(\alpha)=(h_0,h_1,\ldots,h_{2b'-1})$, 令 $a=(h_0,h_1,\ldots,h_{b'-1})$, $b=(h_{b'},h_{b'+1},\ldots,h_{2b'-1})$;

② 利用 $a=(h_0,h_1,\ldots,h_{b'-1})$ 计算整数 $x=2^{b'-2}+\sum_{i=3}^{b'-3}2^i\cdot h_i \bmod n$ 作为签名辅助私钥;

③ 计算签名公钥 $A=xB$.

(2) 签名算法 ($Sign$)

① 随机选取整数 $k\in Z_n^*$;

② 计算签名消息 m 的哈希值 $e=H_2(m)$;

③ 计算临时密钥 $r=H_2(b,e,k) \bmod n$;

④ 计算 $R=rB$, 若 R 为无穷远点, 则返回①;

⑤ 计算 $h=H_2(R,A,e) \bmod n$;

⑥ 计算 $s=(r+hx) \bmod n$, 若 $s=0$ 返回到①;

⑦ 签名者对消息 m 的签名为 $\sigma=(R,s)$, 发送签名 σ 至验证者.

(3) 验证算法 ($Verify$)

① 验证者计算 $h=H_2(R,A,e) \bmod n$;

② 验证者检验等式 $sB=R+hA$ 是否成立, 若成立, 则签名 $\sigma=(R,s)$ 为消息 m 的有效签名.

抗差分故障攻击的 EdDSA 签名方案 π 正确性分析如下.

由 $s=(r+hx) \bmod n$ 得:

$$sB=(r+hx)B=rB+hxB=R+hA.$$

验证者验证 $sB=R+hA$ 等式, 即左边=右边, 签名 $\sigma=(R,s)$ 验证通过, 该签名方案满足正确性要求.

3.3 安全性分析

(1) 存在性不可伪造

定理 1. 在随机预言机模型下, 如果离散对数问题是困难的, 则抗差分故障攻击的 EdDSA 签名是 EUF-CMA 安全.

证明: 若存在一个 PPT 敌手 $\mathcal{A}$, 在随机预言机模型中能够以不可忽略的概率 $\varepsilon(\lambda)$ (λ 是安全参数) 输出一个抗差分故障的 EdDSA 签名的有效伪造, 则存在模拟者 $\mathcal{S}$ 能够求解椭圆曲线离散对数困难问题. 具体证明如下.

模拟者 $\mathcal{S}$ 收到离散对数困难问题实例 $A=xB$, 将 A 作为签名公钥发送给敌手 $\mathcal{A}$. $\mathcal{A}$ 可以进行哈希询问、签名询问. 本文借鉴文献 [24] (Theorem 14) 的证明思路, 仅需证明本方案在随机预言机模型下, 无需签名者私钥的参与, 即可产生一个合法的签名元组, 且产生的签名元组与真实签名具有同样的分布. 然后根据分叉引理[24], 若敌手 $\mathcal{A}$ 输出一个有效伪造, 通过回放, 可以输出两个伪造签名元组, 以解决困难问题实例. 下面我们说明产生有效签名元组的方式具体操作: 给定一个消息 m, $\mathcal{S}$ 随机选择 $h,s\in Z_n^*$, 设置 $R=sB-hA$, 一个有效的签名元组即可表示为 (m,R,h,s), 其中 $\sigma=(R,s)$, 且这一签名元组的各个元素均与真实签名同分布.

伪造: 敌手 $\mathcal{A}$ 输出一个关于消息 m 的伪造签名 (m,R,h,s), 根据分叉引理, $\mathcal{A}$ 可以输出另一个伪造 (m,R,h',s'), 其中 $h\neq h'$. $\mathcal{S}$ 根据 $\mathcal{A}$ 伪造的两组有效签名 (m,R,h,s) 和 (m,R,h',s') 建立方程:

$$sB-hA=R=s'B-h'A.$$

$\mathcal{S}$ 可以很容易地求解出上述方程中 A 关于 B 的离散对数 x:

$$x=\frac{s-s'}{h-h'}.$$

综上, 如果存在敌手 $\mathcal{A}$ 能够以不可忽略的概率 $\varepsilon(\lambda)$ 有效伪造一个抗差分故障攻击的 EdDSA 签名, 那么一定存在一个 $\mathcal{S}$ 解决离散对数问题, 证毕.

(2) 抗差分故障攻击

定理 2. 如果敌手 $\mathcal{A}$ 在概率多项式时间内, 对同一消息 m 进行两次签名询问, $\mathcal{A}$ 可以分别获得一组正确和一组错误的抗差分故障攻击的 EdDSA 签名, $\mathcal{A}$ 利用定义 3 密钥提取的方法提取出私钥 d 的概率是可忽略的, 则该签名是抗差分故障攻击安全的.

证明: 若敌手 $\mathcal{A}$ 在抗差分故障攻击的 EdDSA 签名算法正常执行后得到消息 m 一组正确的签名 (R,s), $\mathcal{A}$ 在签名过程中任意位置注入故障, $\mathcal{A}$ 可以获得消息 m 的一组错误签名 (R',s'), $\mathcal{A}$ 试图利用获取的 (R,s) 和 (R',s') 来计算签名私钥, 建立以下方程组:

$$\begin{cases}(R,s)=(rB,r+hx \bmod n)\\(R',s')=(\tilde{r}B',\tilde{r}+h'x \bmod n)\end{cases}.$$

上述方程式中 $h=H_2(R,A,e) \bmod n$, 其中 R,A,e,s 已知, 则可以计算出 h, 同理 h' 也已知. 但是上述方程式中 r, $\tilde{r}$ 和 x 未知, 所以攻击者无法解方程式得到签名私钥 x'. 更无法判断 $x'=x$ 是否成立, 故 $\mathcal{A}$ 对抗差分故障攻击的 EdDSA 签名密钥 x 提取失败. 其提取优势为 $Adv_{\mathcal{A},\pi}^{KeyExtract}(\lambda)=0$.

同理, 当攻击者将故障注入 EdDSA 签名算法的其他组件, 如签名公钥 A, 签名组件 r, R, h 或随机数 k 等, 由于抗差分故障攻击的 EdDSA 签名算法有随机数参与, 针对同一消息执行多次得到的签名都不同, $\mathcal{A}$ 即使具备差分故障攻击能力, 也无法利用正确签名和错误签名计算出签名私钥. 其具体分析方法同上. 证毕.

3.4 性能分析

(1) 计算复杂度分析

本节针对 EdDSA 与抗差分故障攻击的 EdDSA 签名算法的模乘、模逆及点积运算进行分析, 一次模逆运算

约等于算法执行 9 次点积运算.本文采用 $[mc]$, $[mn]$ 和 $[dj]$ 分别表示模乘、模逆与点积运算, 设一次模乘运算的数据规模为 n, 则一次模乘、模逆与点积运算的复杂度分别 $O(n^2\log_2 n)$, $O(9n^2)$ 和 $O(n^2)$, 其中 $[mn]=9[dj]$. T_1, T_2 分别表示 EdDSA 签名方案与抗差分故障攻击的 EdDSA 签名方案算法的总运算量, 两种签名方案的运算量比较如表 1 所示.

表 1 EdDSA 与抗差分故障攻击的 EdDSA 的签名方案算法运算量比较

方 案	密钥生成			签名			验证			总 计
	$[mc]$	$[mn]$	$[dj]$	$[mc]$	$[mn]$	$[dj]$	$[mc]$	$[mn]$	$[dj]$	
EdDSA	0	0	1	1	0	1	1	0	1	T_1
抗差分故障攻击的EdDSA	0	0	1	1	0	1	1	0	1	T_2

表 1 中 $T_1=O[(2\log_2 n+3)n^2]$, $T_2=O[(2\log_2 n+3)n^2]$, 其中 T_1 与 T_2 相等. EdDSA 签名算法嵌入随机噪声使其具备了抗差分故障攻击的能力, 但对 EdDSA 签名算法的计算复杂度基本没有影响.

(2) 算法执行效率分析

本节针对 EdDSA 签名算法与抗差分故障攻击的 EdDSA 签名算法的执行效率进行测试分析, 对每个算法执行 1 000 次求其平均执行效率. 实验环境为 Inter(R) Core(TM) i5-7400 CPU@3.00 GHz 的 Ubuntu 16.0.4 LTS, 64 位操作系统, 内存容量为 6.00 GB 台式机. 实验编译环境采用 JetBrains PyCharm 2019.2.3 版本, 编程语言采用 Python 3.6.5 编译程序. 测试 EdDSA 签名算法和抗差分故障攻击的 EdDSA 签名算法的密钥生成、签名算法及验证算法执行效率, 如图 1 所示.

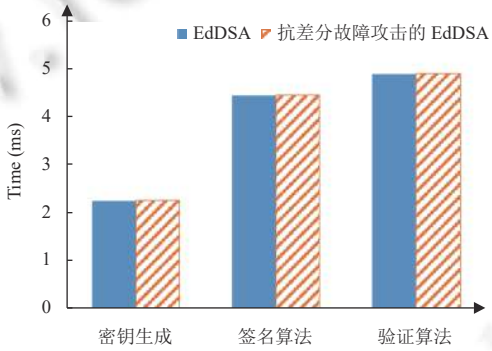

图 1 EdDSA 与抗差分故障攻击的 EdDSA 签名算法执行效率对比

依据图 1 分析可得, EdDSA 签名算法与抗差分故障攻击的 EdDSA 签名算法在密钥生成、签名算法和验证算法的执行效率近似相同, 抗差分故障攻击的 EdDSA 签名算法执行效率不仅与 EdDSA 签名算法执行效率相同, 同时还具备了抵抗差分故障攻击的安全特性.

4 抗差分故障攻击的两方协同 EdDSA 签名

为降低签名私钥泄漏风险, 抵抗签名者权利过于集中、防止单点失效, 增强签名方案的安全性等问题. 本节受 Lindell 等人[23]提出的两方协同 ECDSA 签名算法思想启发, 借助零知识证明技术、数字承诺技术及 Paillier 同态加密技术, 设计了抗差分故障攻击的两方协同 EdDSA 签名方案, 其中零知识证明技术确保了参与者双方身份的真实性, 数字承诺技术确保了输出签名的正确性, Paillier 同态加密技术保证了通信双方在不知道对方签名私钥的情况下, 仍能够协作计算产生有效的抗差分故障攻击的两方协同 EdDSA 签名. 定义 $\Pi=(KeyGen, Sign, Verify)$ 是抗差分故障攻击的两方协同 EdDSA 签名方案, 该算法系统参数设置与 EdDSA 签名算法参数设置相同. 抗差分故障攻击的 EdDSA 签名方案的密钥生成算法和签名算法由两个参与方协同完成, 分别为参与方 P_1 和参与方 P_2. Π 签

名方案具体描述如下.

4.1 两方协同 EdDSA 签名算法描述

(1) 两方密钥生成算法

1) P_1 第 1 条信息

① P_1 任意选取 $\alpha_1 \in \{0,1\}^{b'}$ 作为私钥, 计算 $H_1(\alpha_1)=(h_0^1,h_1^1,\ldots,h_{2b'-1}^1)$, 令 $d_1^1=(h_0^1,h_1^1,\ldots,h_{b'-1}^1)$, $d_2^1=(h_{b'}^1,h_{b'+1}^1,\ldots,h_{2b'-1}^1)$, 计算 $x_1=\left(2^{b'-2}+\sum_{i=3}^{b'-3}2^i\cdot h_i^1\right) \bmod n$ 作为 P_1 的签名辅助私钥, 计算 $A_1=x_1B$ 作为 P_1 签名公钥;

② P_1 发送 $(com\text{-}prove,1,A_1,x_1)$ 给理想函数 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$.

2) P_2 第 1 条信息

① P_2 从 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$ 接收到 $(proof\text{-}receipt,1)$;

② P_2 任意选取 $\alpha_2 \in \{0,1\}^{b'}$ 作为私钥, 计算 $H_1(\alpha_2)=(h_0^2,h_1^2,\ldots,h_{2b'-1}^2)$, 令 $d_1^2=(h_0^2,h_1^2,\ldots,h_{b'-1}^2)$, $d_2^2=(h_{b'}^2,h_{b'+1}^2,\ldots,h_{2b'-1}^2)$, 计算 $x_2=\left(2^{b'-2}+\sum_{i=3}^{b'-3}2^i\cdot h_i^2\right) \bmod n$ 作为 P_1 签名辅助私钥, 计算 $A_2=x_2B$ 作为参与方 P_2 的签名公钥;

③ P_2 发送 $(prove,2,A_2,x_2)$ 给理想函数 $\mathcal{F}_{zk}^{R_{DL}}$.

3) P_1 第 2 条信息

① P_1 从 $\mathcal{F}_{zk}^{R_{DL}}$ 接收 $(proof,2,A_2)$, 如果 P_1 未接收到关于 x_2 的证明, 则退出该协议;

② P_1 发送 $(decom\text{-}proof,1)$ 到理想函数 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$;

③ P_1 生成 Paillier 同态加密密钥对 (pk,sk), 计算 $c_{key}=Enc_{pk}(x_1)$ 并将其发送给 P_2;

④ P_1 发送 $(prove,1,N,(p,q))$ 到 $\mathcal{F}_{zk}^{R_P}$, 其中 pk 满足 $pk=N=p\cdot q$;

⑤ P_1 生成零知识证明 $(c_{key},pk,A_1)\in T_{PDL}$, 发送给 P_2.

4) P_2 验证

① P_2 从 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$ 接收到 $(decom\text{-}proof,1,A_1)$, 从 $\mathcal{F}_{zk}^{R_P}$ 接收到 $(proof,1,N)$, P_2 从 P_1 接收到 c_{key};

② P_2 接受 P_1 关于 $(c_{key},pk,A_1)\in T_{PDL}$ 的零知识证明;

③ P_2 验证 Paillier 公钥 pk 的长度是否满足条件;

④ 若以上①②③条件都满足, 则 P_2 继续参与协议; 否则, P_2 退出该协议.

5) 输出

① P_1 计算公钥 $A=x_1\cdot A_2$, 并将 (x_1,A) 保存;

② P_2 计算公钥 $A=x_2\cdot A_1$, 并将 (x_2,A,c_{key}) 保存.

(2) 两方协同签名算法

本文在构造两方协同 EdDSA 签名算法时, 为了抵抗差分故障攻击, P_1 和 P_2 在计算 r_1,r_2 时, 分别引入随机数 $k_1,k_2\in Z_n^*$, 计算得到 $r_1=H_2(d_2^1,e,k_1) \bmod n$ 和 $r_2=H_2(d_2^2,e,k_2) \bmod n$. 此方法使得确定性 EdDSA 签名变成了具备抵抗差分故障攻击能力的两方协同 EdDSA 签名算法. 参与方 P_1 与 P_2 在交互之前, 首先在本地计算关于消息 m 的哈希值 $e=H_2(m)$, 同时验证唯一的会话标识 sid 在之前是否被使用过, 若使用过, 则不执行协议. 两方协同签名算法具体描述如下.

1) P_1 第 1 条信息

① P_1 任意选取随机数 $k_1\in Z_n^*$, 计算 $r_1=H_2(d_2^1,e,k_1) \bmod n$, 计算 $R_1=r_1\cdot B$;

② P_1 发送 $(com\text{-}prove,sid\|1,R_1,r_1)$ 给理想函数 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$.

2) P_2 第 1 条信息

① P_2 从 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$ 接收 $(proof\text{-}receipt,sid\|1)$;

② P_2 任意选取随机数 $k_2\in Z_n^*$, 计算 $r_2=H_2(d_2^2,e,k_2) \bmod n$, 计算 $R_2=r_2\cdot B$;

③ P_2 发送 $(prove,sid\|2,R_2,r_2)$ 给 $\mathcal{F}_{zk}^{R_{DL}}$.

3) P_1 第 2 条信息

① P_1 从 $\mathcal{F}_{zk}^{R_{DL}}$ 接收 $(proof, sid\|2, R_2)$, 若未接收到信息, 则 P_1 退出协议;

② P_1 发送 $(decom\text{-}proof, sid\|1)$ 给 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$.

4) P_2 第 2 条信息

① P_2 从 $\mathcal{F}_{com\text{-}zk}^{R_{DL}}$ 接收 $(decom\text{-}proof, sid\|1, R_1)$, 若未接收到信息, 则 P_2 退出协议;

② P_2 计算 $R = R_1 + R_2$;

③ P_2 计算 $h = H_2(R, A, e) \bmod n$, 计算 $c_1 = Enc_{pk}(r_2)$ 和 $v = x_2 \cdot h$, 然后计算 $c_2 = v \odot c_{key}$, 令 $c_3 = c_1 \oplus c_2$, 将 c_3 发送给 P_1;

5) P_1 输出签名

① P_1 计算 $R = R_1 + R_2$;

② P_1 解密 c_3, 即 $s' = Dec_{sk}(c_3)$, 计算 $s = r_1 + s' \bmod n$;

③ P_1 利用签名公钥 A 验证签名 $\sigma = (R, s)$ 是否为有效签名, 若验证通过, 则签名 (R, s) 作为消息 m 的签名输出; 否则, 退出协议.

(3) 验证算法

抗差分故障攻击的两方协同 EdDSA 签名验证算法与 EdDSA 签名验证算法相同, 验证者首先计算 $h = H_2(R, A, e) \bmod n$; 然后验证者检验等式 $sB = R + hA$ 是否成立, 若成立, 则签名 $\sigma = (R, s)$ 为消息 m 的有效签名, 否则, 签名无效.

4.2 安全性分析

由于抗差分故障攻击的 EdDSA 签名中有随机数的参与, 针对同一消息执行多次得到的签名都不同, 使其扩展到两方协同 EdDSA 签名方案也具备了抗差分故障攻击性, 即使攻击者具备差分故障攻击能力也无法提取签名私钥. 因此, 本节主要证明基于理想函数 $\mathcal{F}_{com}$、$\mathcal{F}_{zk}^{R}$ 和 $\mathcal{F}_{com\text{-}zk}^{R}$ 的 UC 混合模型设计的抗差分故障攻击的两方协同 EdDSA 签名方案满足 EUF-CMA 安全性. 文献 [17] 和文献 [25] 分别证明了单个理想函数 $\mathcal{F}_{com}$、$\mathcal{F}_{zk}^{R}$ 和 $\mathcal{F}_{com\text{-}zk}^{R}$ 的安全性和组合函数的安全性.

定理 3. 若抗差分故障攻击的 EdDSA 签名方案是 EUF-CMA 安全, Paillier 同态加密方案满足选择明文攻击下具备不可区分性安全, 则抗差分故障攻击的两方协同 EdDSA 签名方案是 EUF-CMA 安全的.

证明: 抗差分故障攻击的两方协同 EdDSA 签名算法是基于 $\mathcal{F}$-混合模型下证明其安全性. 对于任意 PPT 敌手 $\mathcal{A}$, 构造一个敌手 S (模拟者) 在 $Expt\text{-}Sign_{S,\pi}(1^\lambda)$ 游戏中伪造抗差分故障攻击 EdDSA 签名, 其概率接近于 $\mathcal{A}$ 在 $Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^\lambda)$ 游戏中伪造对应协同签名的概率. 简而言之, 如果 Paillier 同态加密算法在选择明文攻击下是不可区分安全的, 则对于任意概率多项式时间敌手 $\mathcal{A}$ 和 $b \in \{1,2\}$, 存在一个敌手 S 和一个可忽略的函数 ε, 使得对于每个 λ 都有:

$$\left|\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda) = 1] - \Pr[Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^\lambda) = 1]\right| \leqslant \varepsilon(\lambda) \tag{1}$$

其中, π 表示抗差分故障攻击的 EdDSA 签名方案, Π 是由 π 派生的抗差分故障攻击的两方协同 EdDSA 签名方案, 由 π 签名方案安全性可知, 存在一个可忽略函数 ε' 使得对于每个 λ, $\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda) = 1] \leq \varepsilon'(\lambda)$, 则游戏 $Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^\lambda)$ 满足:

$$\Pr[Expt\text{-}DistSign^{b}_{\mathcal{A},\Pi}(1^\lambda) = 1] \leqslant \varepsilon'(\lambda) + \varepsilon(\lambda) \tag{2}$$

显然, $\varepsilon'(\lambda) + \varepsilon(\lambda)$ 是可忽略的. 综上, 只需要证明当 $b \in \{1,2\}$ 时, 敌手 $\mathcal{A}$ 分别腐化参与方 P_1 或 P_2 时, 均无法攻破方案 Π, 即公式 (1) 成立即可.

(1) 当 $b = 1$, 即参与方 P_1 被 $\mathcal{A}$ 腐化. 令敌手 $\mathcal{A}$ 是游戏 $Expt\text{-}DistSign^{1}_{\mathcal{A},\Pi}(1^\lambda)$ 中的 PPT 的敌手, 构造一个游戏 $Expt\text{-}Sign_{S,\pi}(1^\lambda)$ 中敌手 S, 敌手 S 模拟实验 $Expt\text{-}DistSign^{1}_{\mathcal{A},\Pi}(1^\lambda)$ 且与敌手 $\mathcal{A}$ 交互, 具体模拟过程如下.

1) 在实验 $Expt\text{-}Sign_{S,\pi}(1^\lambda)$ 中, 敌手 S 收到 $(1^\lambda, A)$. 其中 A 是抗差分故障攻击的 EdDSA 签名公钥.

2) 敌手 S 模拟游戏 $Expt\text{-}DistSign^{1}_{\mathcal{A},\Pi}(1^\lambda)$ 中的预言机 O 对 $\mathcal{A}$ 的询问进行应答, 详细询问-应答如下.

① 在实验开始阶段, $\mathcal{A}$ 在两种情况下向 O 所有的询问 $(sid,\cdot)$, S 都以 $\perp$ 应答. 第 1 种情况是密钥生成协议之前, 第 2 种情况是 $\mathcal{A}$ 未按照要求向预言机问询, 直到 $\mathcal{A}$ 发起问询为 $(0,0)$ 时, 实验开始;

② $\mathcal{A}$ 发送消息 $(0,0)$ 给 O 后, S 收到 P_1 在密钥生成子协议中的第 1 条消息 $(0,m_1)$, S 模拟预言机 O 应答如下.

i. S 解析 m_1 为混合模型中 P_1 发送给 $\mathcal{F}_{com\text{-}zk}^{DL}$ 的消息 $(com\text{-}prove,1,A_1,x_1)$;

ii. S 验证 $A_1=x_1\cdot B$ 是否成立, 若成立, 则计算 $A_2=(x_1)^{-1}\cdot A$; 否则, S 随机选取椭圆曲线上的点作为 A_2;

iii. S 设置预言机 O 的应答为 $(proof,2,A_2)$, 然后发送给 $\mathcal{A}$.

③ S 收到 $\mathcal{A}$ 下一个询问消息 $(0,m_2)$, 处理如下.

i. S 解析 m_2 为以下 3 条信息: ① $\mathcal{A}$ 发送给 $\mathcal{F}_{com\text{-}zk}^{DL}$ 的消息 $(decom\text{-}proof,sid\|1)$, ② $\mathcal{A}$ 发送给 $\mathcal{F}_{com\text{-}zk}^{R}$ 的消息 $(proof,1,N,(p,q))$, ③ $\mathcal{A}$ 发送的密文 c_{key}.

ii. S 验证 $pk=N=p\cdot q$ 是否成立, 如果等式不成立, 则中断交互过程.

iii. 如果 $A_1\neq x_1\cdot B$ 或 $x_1\neq Dec_{sk}(c_{key})$ 或 $x_1\notin Z_n$, S 生成预言机 O 的应答为 P_2 退出协议. 若 S 模拟终止. 则游戏终止, 该情况下 S 不会输出任何信息.

iv. S 接收到下一个形式为 $(0,m_i)$ 的消息, 将其作为 $(c_{key},pk,A_1)\in T_{PDL}$ 的零知识证明的一部分进行处理. S 通过诚实验证程序来验证此证明. 该零知识证明详细细节见参考文献 [23].

v. 此游戏中, 如果 S 模拟没有终止协议, 则 S 存储 (x_1,A,c_{key}), 分布式密钥生成阶段模拟完成.

④ 当 S 收到形式为 (sid,m) 的问询后. 首先, S 利用 m 询问它在游戏 $Expt\text{-}Sign_{S,\pi}(1^{\lambda})$ 中的签名预言机, 该函数实例执行后, S 会收到签名预言机应答为 (R,s). S 从 $\mathcal{A}$ 接收到带有标识符为 sid 的问询处理如下.

i. S 收到 $\mathcal{A}$ 的第 1 条询问消息 (sid,m_1) 被解析为 $(com\text{-}prove,sid\|1,R_1,r_1)$ 来处理, 如果 $R_1=r_1\cdot B$, 则 S 设置 $R_2=R-R_1$, 否则, S 任意选取一个椭圆曲线上的点 R_2, 设置预言机 O 的应答为 $(proof,sid\|2,R_2)$, 此应答是 $\mathcal{A}$ 期望收到的应答.

ii. S 收到 $\mathcal{A}$ 第 2 条询问消息 (sid,m_2) 被解析为 $(decom\text{-}proof,sid\|1)$ 处理, 如果 $R_1\neq r_1\cdot B$, 则 S 模拟 P_2 退出协议; 否则, S 任意选取 $\rho\in Z_{n^2}$, 计算 $c_3=Enc_{pk}((s-r_1)\bmod n+\rho\cdot n)$, S 设置 c_3 为预言机 O 对 $\mathcal{A}$ 第 2 条询问消息的应答.

3) 当 $\mathcal{A}$ 终止并输出一个消息签名对 (m^*,σ^*), 敌手 S 也输出 (m^*,σ^*). 此时, 可以通过证明公式 (1) 成立, 即证明 S 模拟 $\mathcal{A}$ 的视图与 $\mathcal{A}$ 在真实协议中的视图是不可区分的.

在模拟环境中 S 为 $\mathcal{A}$ 生成的公钥 A, 实际上是它在 $Expt\text{-}Sign_{S,\pi}(1^{\lambda})$ 游戏中接收到的公钥. 由于当敌手 $\mathcal{A}$ 承诺 $A_1=x_1\cdot B$ 时, S 定义公钥 $A_2=(x_1)^{-1}\cdot A$. 因此, 公钥被定义为 $x_1\cdot A_2=x_1\cdot(x_1)^{-1}\cdot A=A$, 所以在实际执行协议的密钥生成阶段和签名阶段, S 模拟 $\mathcal{A}$ 的视图在统计上接近于真实协议执行的视图. 此情况意味着 $\mathcal{A}$ 可以输出有效签名对 (m^*,σ^*), 在模拟游戏和真实 $Expt\text{-}DistSign_{\mathcal{A},\Pi}^{1}(1^{\lambda})$ 协议中具有相同的概率, 因为模拟游戏中的公钥与 S 在其中接收到的公钥相同. 所以 $\mathcal{A}$ 在游戏 $Expt\text{-}DistSign_{\mathcal{A},\Pi}^{1}(1^{\lambda})$ 中输出的伪造签名有效构成了 S 在游戏 $Expt\text{-}Sign_{S,\pi}(1^{\lambda})$ 中的伪造. 下面主要证明在真实协议与模拟游戏中 $\mathcal{A}$ 的视图分布是不可区分的.

密钥生成阶段: $\mathcal{A}$ 在真实协议中的视图与模拟游戏中的视图唯一区别是公钥 A_2 的生成方式: 在真实协议中, 参与方 P_2 任意选取 $\alpha_2\in\{0,1\}^{b'}$, 计算 $H_1(\alpha_2)=(h_0^2,h_1^2,\ldots,h_{2b'-1}^2)$, 令 $d_1^2=(h_0^2,h_1^2,\ldots,h_{b'-1}^2)$, $d_2^2=(h_{b'}^2,h_{b'+1}^2,\ldots,h_{2b'-1}^2)$, 计算 $x_2=(2^{b'-2}+\sum_{i=3}^{b'-3}2^i\cdot h_i^2)\bmod n$ 作为 P_1 签名辅助私钥, 然后计算 $A_2=x_2B$ 作为参与方 P_2 的签名公钥; 而在模拟游戏中 S 计算 $A_2=(x_1)^{-1}\cdot A$, 由于 A 是在椭圆曲线上任意选取的, 故 $x_2\cdot B$ 与 $(x_1)^{-1}\cdot A$ 是具有同分布的.

签名阶段: A 在真实协议中的视图与模拟游戏中的视图是计算上不可区分的, 唯一区别是 c_3 的生成方式. 具体来说, 首先签名组件 R_2 是同分布的: 在真实协议中, $R_2=r_2\cdot B$, 其中, $r_2=H_2(d_2^2,e,k_2)\bmod n$. 在模拟游戏中 $R_2=(r_1)^{-1}\cdot R$, 其中, R 是 $Expt\text{-}DistSign_{\mathcal{A},\Pi}^{1}(1^{\lambda})$ 签名函数生成的, 显然 $r_2\cdot B$ 与 $(r_1)^{-1}\cdot R$ 是同分布的. 在 $\mathcal{F}_{zk}^{R}$ 和 $\mathcal{F}_{com\text{-}zk}^{R}$ 混合模型下, 零知识证明和验证也是同分布的, 因此在真实协议与模拟游戏中唯一的区别是 c_3 的生成方式. 在模拟游戏中 $c_3=Enc_{pk}((s-r_1)\bmod n+\rho\cdot n)$ 计算而来, 而在真实协议中 $c_3=Enc_{pk}(s')=Enc_{pk}((r_2+xh)\bmod n+\rho\cdot n)$, 其中 $\rho\in Z_{n^2}$. 签名组件 $s=(Dec_{sk}(c_3)+r_1)\bmod n$. 因此, 需要证明 S 模拟敌手 $\mathcal{A}$ 的视图在真实协议与模拟游戏中是不可区分的, 尽管存在差异, 这些值实际上统计上是接近的. 对于 EdDSA 签名公钥不变的情况下, 签名变形为 $s=$

$r+xh=(r_1+r_2)+x_1\cdot x_2\cdot h \bmod n$, 同时满足$r_2+x_1\cdot x_2\cdot h=(s-r_1) \bmod n$. 任意选取$\eta\in N$, 满足$0\leqslant\eta<n$, 其中$r_2+x_1\cdot x_2\cdot h=(s-r_1)+\eta\cdot n \bmod n$, η 满足$0\leqslant\eta<n$的原因是没有模块化规约运算$r_2 \bmod n$加上$x_1\cdot x_2\cdot h \bmod n$, 其结果不可能增加到超过$n^2$. 因此, 在真实协议与模拟游戏中$S$仿真的差异在于:

- 真实协议: c_3是由$[s-r_1 \bmod n]+\eta\cdot n+\rho\cdot n$加密生成.
- 模拟游戏: c_3是由$[s-r_1 \bmod n]+\rho\cdot n$加密生成.

上述$r_1,s,\eta\in Z_n$, 对于随机选择的$\rho\in Z_{n^2}$, 上述计算的值在统计上是接近的. 为了更加清晰分析, 对于$r_1,s,\eta\in Z_n$, 设置一个v. 若对于值φ存在$v\neq[s-r_1 \bmod n]+\varphi\cdot n$, 那么真实协议和模拟游戏的值都不等于$v$; 若对于值$\varphi$存在$v=[s-r_1 \bmod n]+\varphi\cdot n$, 则分为以下 3 种情况讨论.

① 若$\varphi<\eta$: 此情况下模拟游戏中的执行结果满足v的值, 其中$\rho<\eta$, 在真实协议中执行结果不满足v;

② 若$\varphi>n^2-1$: 此情况下真实协议的执行结果满足v的值, 其中$\rho\geqslant n^2-1-\eta$, 在模拟游戏中执行结果不满足v的值;

③ 若$\eta\leqslant\varphi\leqslant n^2-1$: 此情况下, v的值可能是真实协议也可能是模拟游戏的结果, 二者概率是相同的.

令X表示真实协议的分布, Y表示模拟游戏的分布, D表示区分器, 则存在以下定义:

$$\Delta(X,Y)=\max_{T\subseteq D}|\Pr[X\in T]-|\Pr[Y\in T]|.$$

若T表示当$\varphi<\eta$时v的值的集合, 则$\Pr[X\in T]=0$, $\Pr[Y\in T]\leqslant n/n^2=1/n$(其中, $0\leqslant\eta\in n$和$\rho\in Z_{n^2}$), 此时$\Delta(X,Y)=1/n$该值是可忽略的. 同理, 若T表示当$\varphi>n^2-1$时v的值的集合, 则$\Pr[Y\in T]=0$, $\Pr[X\in T]\leqslant n/n^2=1/n$, 此时$\Delta(X,Y)=1/n$该值是可忽略的. 因此, 在真实协议和模拟游戏中$c_3$的分布在统计上是接近的, 综上证明公式 (1) 在$b=1$的情况下成立.

(2) 当$b=2$, 即参与方P_2被$\mathcal{A}$腐化. 此种情况和P_1被腐化分析相同, 令$\mathcal{A}$是游戏$Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$中的 PPT 的敌手, 构造游戏$Expt\text{-}Sign_{S,\pi}(1^\lambda)$中的一个敌手$S$(模拟者), 敌手$S$本质上模拟实验$Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$且与敌手$\mathcal{A}$交互, 具体模拟过程与$P_1$被腐化模拟唯一区别是: P_2发送给P_1的c_3可能被恶意构造, 模拟者无法发现. 本文通过假设S在某个时刻模拟P_1终止协议, 解决P_2恶意构造c_3的问题. 具体的, S任意选取$i\in\{1,\ldots,p(\lambda)+1\}$, $p(\lambda)$表示敌手$\mathcal{A}$询问预言机O次数的上限, 如果S选择正确, 则模拟是完备的. S选择正确的概率是$1/(p(\lambda)+1)$, 此时意味着S能够以$1/(p(\lambda)+1)$的概率模拟$\mathcal{A}$的视图. 因此, S在实验$Expt\text{-}Sign_{S,\pi}(1^\lambda)$中伪造一个签名的概率至少是$\mathcal{A}$在游戏$Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$的$1/(p(\lambda)+1)$倍. 令$\mathcal{A}$是一个 PPT 敌手, S模拟如下所示.

1) 在实验$Expt\text{-}Sign_{S,\pi}(1^\lambda)$中, S收到$(1^\lambda,A)$, 其中A是抗差分故障攻击 EdDSA 签名公钥;

2) 令$p(\cdot)$表示$\mathcal{A}$在实验$Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$中向预言机$O$询问次数的上限, 则$S$随机选$i\in\{1,\ldots,p(\lambda)+1\}$;

3) S在游戏$Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$中模拟预言机$O$对敌手$\mathcal{A}$询问-应答如下.

① 在密钥生成子协议结束之前, $\mathcal{A}$向O所有的询问$(sid,\cdot)$, S应答均为$\perp$, 直到O收到问询$(0,0)$后交互正常开始;

② $\mathcal{A}$发送请求$(0,0)$给预言机O, S计算应答为$(proof\text{-}receipt,1)$, 该应答正是$\mathcal{A}$所期望的.

③ S收到$\mathcal{A}$的下一个询问消息$(0,m_1)$, 处理如下.

i. S解析m_1为混合模型中P_2发送给$\mathcal{F}^{DL}_{com\text{-}zk}$的消息$(prove,2,A_2,x_2)$;

ii. S验证A_2是否为椭圆曲线上的点, 其次, 验证$A_2=x_2\cdot B$是否成立, 若不成立, 则S模拟P_1终止协议;

iii. S生成有效的 Paillier 同态加密密钥对(pk,sk), 计算$c_{key}=Enc_{pk}(\widetilde{x_1})$, 其中任意选择$\widetilde{x_1}\in Z_n$;

iv. S设置预言机O对$\mathcal{A}$的消息应答为$(decom\text{-}proof,1,A_1)$和$(proof,1,N)$, 同时$S$模拟运行$(proof,1,(c_{key},N,Q_1))$零知识证明, 其中$A_1=(x_2)^{-1}A$;

v. S存储(x_2,Q,c_{key}), 两方协同密钥生成协议模拟完成.

④ 当S收到$\mathcal{A}$对消息(sid,m)的询问后, sid表示身份标识, S计算预言机O对敌手$\mathcal{A}$预期的应答为$(proof\text{-}receipt,sid\|1)$. 此外, S询问它在实验$Expt\text{-}Sign_{S,\pi}(1^\lambda)$中的签名预言机, S收到预言机返回的签名(R,s). 其中R表示曲线上的点. S从$\mathcal{A}$接收到的带有标识符为sid的询问处理如下.

i. $\mathcal{A}$ 第 1 条询问消息(sid,m_1)被 S 解析为$(com\text{-}prove,sid\|2,R_2,r_2)$, S 验证$R_2=r_2\cdot B$是否相等且R_2为爱德华曲线上的非零点. 若$R_2\neq r_2\cdot B$, 则 S 模拟 P_1 终止协议; 否则, S 计算 $R_1=R-R_2$并设置预言机 O 的应答 $(decom\text{-}proof,sid\|1,R_1)$, 该应答来自理想函数$\mathcal{F}_{com\text{-}zk}^{R_{DL}}$.

ii. 敌手$\mathcal{A}$ 的第 2 条询问消息(sid,m_2)被 S 解析, 其中 m_2 被解析为 c_3, 如果这是第 i 次$\mathcal{A}$ 向预言机 O 发起询问, 则 S 模拟 P_1 退出协议; 否则继续.

4) 当$\mathcal{A}$ 终止, 并输出一个消息签名对(m^*,σ^*), 敌手 S 也输出(m^*,σ^*)并终止.

在模拟环境中 S 为$\mathcal{A}$ 生成的公钥 A 实际上是它在$Expt\text{-}Sign_{S,\pi}(1^\lambda)$ 实验中接收到的公钥 A 计算而来的, 令 j 是以(sid,c_3)对预言机 O 的第 1 次调用. 其中 c_3 使得 P_1 不能获得有效签名对(R,s). 如果 $j=i$, 则$\mathcal{A}$ 在真实协议和模拟游戏之间的区别仅有c_{key}不同. 具体讲, 在真实协议中 $c_{key}=Enc_{pk}(x_1)$且 $A_1=x_1\cdot B$, 然而在模拟游戏中, $c_{key}=Enc_{pk}(\widetilde{x_1})$, 其中 $\widetilde{x_1}$ 任意选择且独立于$A_1=x_1\cdot B$. 通过观察, 在模拟游戏中, S 未采用 Paillier 同态加密的私钥, 因此, 真实协议与模拟游戏的不可区分性, 可归约到选择明文攻击下 Paillier 同态加密方案的不可区分性.

$$\left|\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda)=1|i=j]-\Pr[Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)=1]\right|\leqslant\varepsilon(\lambda).$$

推导得出:

$$\Pr[Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)=1]\leqslant\frac{\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda)=1\wedge i=j]}{\Pr[i=j]}+\varepsilon(\lambda)\frac{\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda)=1]}{1/(p(\lambda)+1)}+\varepsilon(\lambda)\ .$$

化简得出:

$$\Pr[Expt\text{-}Sign_{S,\pi}(1^\lambda)=1]\geqslant\frac{\Pr[Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)=1]}{p(\lambda)+1}-\varepsilon(\lambda).$$

以上情况意味着, 如果$\mathcal{A}$ 能在 $Expt\text{-}DistSign^2_{\mathcal{A},\Pi}(1^\lambda)$中以不可忽略的概率伪造签名, 则 S 就可以在 $Expt\text{-}Sign_{S,\pi}(1^\lambda)$中以不可忽略的概率伪造签名, 该情况显然与抗差分故障攻击的 EdDSA 签名安全性矛盾.

综上所述, 当 $b=2$, 参与方 P_2 被腐化的情况下公式 (1) 成立, 证毕.

4.3 性能分析

本节从算法的计算复杂度与算法执行效率两方面, 分别对两方协同 ECDSA 签名方案与抗差分故障攻击的两方协同 EdDSA 签名方案进行对比分析.

(1) 计算复杂度分析

本节针对抗差分故障攻击的两方协同 EdDSA 签名算法的模乘、模逆及点积运算进行分析, 一次模逆运算约等于算法执行 9 次点积运算. 本文采用 $[mc]$, $[mn]$ 和 $[dj]$分别表示模乘、模逆与点积运算, 设一次模乘运算的数据规模为 n, 则一次模乘、模逆与点积运算的复杂度分别 $O(n^2\log_2 n)$, $O(9n^2)$ 和 $O(n^2)$, 其中 $[mn]=9[dj]$. T_1, T_2 分别表示文献 [23] 两方协同 ECDSA 签名方案与抗差分故障攻击的两方协同 EdDSA 签名方案算法的总运算量, 两种协同签名方案的运算量比较如表 2 所示.

表 2 本文方案与文献 [23] 两方协同 ECDSA 的签名方案算法运算量比较

方 案	密钥生成			签名			验证			总 计
	$[mc]$	$[mn]$	$[dj]$	$[mc]$	$[mn]$	$[dj]$	$[mc]$	$[mn]$	$[dj]$	
文献[23]	6	1	15	8	2	13	2	1	2	T_1
本文	6	1	15	5	0	11	0	0	2	T_2

表 2 中 $T_1=O[(16\log_2 n+66)n^2]$, $T_2=O[(11\log_2 n+37)n^2]$, 文献 [23] 两方协同 ECDSA 签名方案的计算复杂度高于本文方案的计算复杂度. 具体对比如图 2 所示, 其中横坐标表示模乘运算的数据规模 n, 纵坐标表示签名方案的计算复杂度 $O(n)$.

(2) 算法执行效率分析

本节分别对两方协同 ECDSA 签名算法和抗差分故障攻击的两方协同 EdDSA 签名算法的效率进行测试, 并对每个算法执行 1 000 次求其效率平均值. 实验环境为 Inter(R)Core(TM) 惠普 i5-7400CPU@3.00 GHz 处理器、Ubuntu 16.0.4 LTS, 64 位操作系统, 内存容量为 6.00 GB 的台式机, 实验编译环境采用 JetBrains PyCharm 2019.2.3 版本, 编译语言采用 Python 3.6.5 编译程序. 两种签名方案的密钥生成、签名及验证算法执行效率分别对比如图 3 所示.

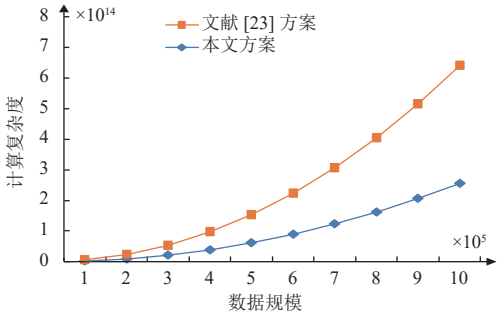

图 2 本文方案与文献 [23] 方案算法计算复杂度比较

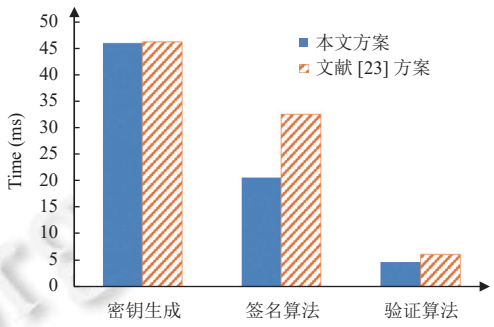

图 3 本文方案与文献 [23] 方案算法执行效率对比图

由图 3 分析, 本文抗差分故障攻击的两方协同 EdDSA 签名方案在密钥生成阶段耗时与文献 [23] 两方协同 ECDSA 签名方案的耗时近似相同, 在签名阶段算法耗时比文献 [23] 两方协同 ECDSA 算法耗时快约 1.6 倍, 验证算法比文献 [23] 两方协同 ECDSA 签名算法耗时短约 1.3 倍, 在签名长度相同情况下, 抗差分故障攻击的两方协同 EdDSA 签名方案算法的总执行效率相比文献 [23] 提高了约 13.3 ms , 同时具备了抵抗差分故障的安全特性.

5 总 结

针对 Libra 稳定币交易单 EdDSA 签名算法面临的安全威胁, 定义了能够抵抗差分故障攻击的数字签名方案需满足的安全性质, 并对 EdDSA 签名方案进行分析并改进, 提出了抗差分故障攻击的 EdDSA 签名方案, 并证明该方案的安全性. 为降低签名私钥泄漏风险, 防止单点失效等问题, 进一步设计了抗差分故障攻击的两方协同 EdDSA 签名方案, 并证明方案的安全性. 最后, 通过对抗差分故障攻击的两方协同 EdDSA 签名算法的计算复杂度分析与算法执行效率测试, 验证了抗差分故障攻击的两方协同 EdDSA 签名算法的有效性.

References:

[1] Sidhu JS. Syscoin: A peer-to-peer electronic cash system with Blockchain-based services for E-business. In: Proc. of the 26th Int'l Conf. on Computer Communication and Networks (ICCCN). Vancouver: IEEE, 2017. 1–6. [doi: 10.1109/ICCCN.2017.8038518]

[2] Amsden Z, Arora R, Bano S, *et al*. The Libra blockchain. 2020. https://cryptorating.eu/whitepapers/Libra/the-libra-blockchain.pdf

[3] Jiang OP, Zhang LL, Liu DZ. Overview of bitcoin, Libra, DCEP. Financial Technology Time, 2020(2): 57–68 (in Chinese with English abstract). [doi: 10.3969/j.issn.2095-0799.2020.02.009]

[4] Fu YJ. The impact of digital currency Libra on cross-border capital flows and foreign exchange regulation. Tsinghua Financial Review, 2020(7): 68–70 (in Chinese with English abstract). [doi: 10.19409/j.cnki.thf-review.2020.07.019]

[5] Romailler Y, Pelissier S. Practical fault attack against the Ed25519 and EdDSA signature schemes. In: Proc. of the 2017 Workshop on Fault Diagnosis & Tolerance in Cryptography (FDTC). Taipei: IEEE, 2017. 17–24. [doi: 10.1109/FDTC.2017.12]

[6] Boneh D, Demillo RA, Lipton RJ. On the importance of eliminating errors in cryptographic computations. Journal of Cryptology, 2001, 14(2): 101–119. [doi: 10.1007/s001450010016]

[7] Biham E, Shamir A. Differential fault analysis of secret key cryptosystems. In: Proc. of the 17th Annual Int'l Cryptology Conf. Santa Barbara on Advances in Cryptology. California: Springer, 1997. 513–525. [doi: 10.1007/BFb0052259]

[8] Biehl I, Meyer B, Müller V. Differential fault attacks on elliptic curve cryptosystems. In: Proc. of the 20th Annual Int'l Cryptology Conf. Santa Barbara on Advances in Cryptology. California: Springer, 2000. 131–146. [doi: 10.1007/3-540-44598-6_8]

[9] Xu SW, Chen C, Wang RR. Improved differential fault attack on scalar multiplication algorithm in elliptic curve cryptosystem. Journal of Computer Applications, 2016, 36(12): 3328–3332 (in Chinese with English abstract). [doi: 10.11772/j.issn.1001-9081.2016.12.3328]

[10] Jin YX, Yang HZ, Wang XB, Yuan QJ. Improved differential fault attack for SM4 cipher. Journal of Cryptologic Research, 2020, 7(4): 453–464 (in Chinese with English abstract). [doi: 10.13868/j.cnki.jcr.000380]

[11] Ambrose C, Bos JW, Fay B, Joye M, Lochter M, Murray B. Differential attacks on deterministic signatures. In: Proc. of the 2018 Cryptographers' Track at the RSA Conf. on Topics in Cryptology. San Francisco: Springer, 2018. 339–353. [doi: 10.1007/978-3-319-76953-0_18]

[12] Zhu L, Chen LY. Protection design of low-cost eSIM chip against laser fault injection attack. Electronic Component and Information Technology, 2019, 3(11): 7–10 (in Chinese with English abstract). [doi: 10.19772/j.cnki.2096-4455.2019.11.004]

[13] Duan XY, Li L, Wu YH, Jin JF. Advanced evolution of power glitch attack and resistance techniques. Computer Science, 2011, 38(S1): 428–431 (in Chinese with English abstract).

[14] Chalkias K, Garillot F, Kondi Y, Nikolaenko V. Non-interactive half-aggregation of EdDSA and variants of Schnorr signatures. In: Proc. of the 2021 Cryptographers' Track at the RSA Conf. on Topics in Cryptology. Online: Springer, 2021. 577–608. [doi: 10.1007/978-3-030-75539-3_24]

[15] Paillier P. Public-key cryptosystems based on composite degree residuosity classes. In: Proc. of the 2000 Int'l Conf. on the Theory and Application of Cryptographic Techniques Prague on Advances in Cryptology. Czech Republic: Springer, 2000. 223–238. [doi: 10.1007/3-540-48910-X_16]

[16] Blazy O, Chevalier C, Pointcheval D, Vergnaud D. Analysis and improvement of Lindell's UC-secure commitment schemes. In: Proc. of the 11th Int'l Conf. on Applied Cryptography and Network Security. Banff: Springer, 2013. 534–551. [doi: 10.1007/978-3-642-38980-1_34]

[17] Canetti R. Security and composition of multiparty cryptographic protocols. Journal of Cryptology, 2000, 13(1): 143–202. [doi: 10.1007/s001459910006]

[18] Hou HX, Yang B, Zhang LN, Zhang MR. Secure two-party SM2 signature algorithm. Acta Electronica Sinica, 2020, 48(1): 1–8 (in Chinese with English abstract). [doi: 10.3969/j.issn.0372-2112.2020.01.001]

[19] Luo ZQ. Formal analysis of security protocols based on universally composable framework [MS. Thesis]. Shanghai: Shanghai Jiao Tong University, 2008 (in Chinese with English abstract).

[20] Fujisaki E. Improving practical UC-secure commitments based on the DDH assumption. In: Proc. of the 10th Int'l Conf. on Security and Cryptography for Networks. Amalfi: Springer, 2016. 257–272. [doi: 10.1007/978-3-319-44618-9_14]

[21] Hazay C, Lindell Y. Efficient Secure Two-party Protocols: Techniques and Constructions. Berlin: Springer, 2010. 3–254. [doi: 10.1007/978-3-642-14303-8]

[22] Hazay C, Mikkelsen GL, Rabin T, Toft T, Nicolosi AA. Efficient RSA key generation and threshold Paillier in the two-party setting. Journal of Cryptology, 2019, 32(2): 265–323. [doi: 10.1007/s00145-017-9275-7]

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

[24] Pointcheval D, Stern J. Security arguments for digital signatures and blind signatures. Journal of Cryptology, 2000, 13(3): 361–396. [doi: 10.1007/s001450010003]

[25] Canetti R. Universally composable security: A new paradigm for cryptographic protocols. In: Proc. of the 42nd IEEE Symp. on Foundations of Computer Science. Newport Beach: IEEE, 2001. 136–145. [doi: 10.1109/SFCS.2001.959888]

附中文参考文献:

[3] 蒋鸥翔, 张磊磊, 刘德政. 比特币、Libra、央行数字货币综述. 金融科技时代, 2020(2): 57–68. [doi: 10.3969/j.issn.2095-0799.2020.02.009]

[4] 付英俊. 数字货币Libra对跨境资本流动和外汇监管的影响. 清华金融评论, 2020(7): 68–70. [doi: 10.19409/j.cnki.thf-review.2020.07.019]

[9] 许盛伟, 陈诚, 王荣荣. 针对椭圆曲线密码系统点乘算法的改进差分故障攻击. 计算机应用, 2016, 36(12): 3328–3332. [doi: 10.11772/j.issn.1001-9081.2016.12.3328]

[10] 金雨璇, 杨宏志, 王相宾, 袁庆军. 对SM4算法的改进差分故障攻击. 密码学报, 2020, 7(4): 453–464. [doi: 10.13868/j.cnki.jcr.000380]

[12] 朱磊, 陈力颖. 低成本eSIM芯片抗激光故障注入攻击的防护设计. 电子元器件与信息技术, 2019, 3(11): 7–10. [doi: 10.19772/j.cnki.2096-4455.2019.11.004]

[13] 段晓毅, 李莉, 武玉华, 靳济芳. 最新电压毛刺(Power Glitch)攻击与防御方法研究. 计算机科学, 2011, 38(S1): 428–431.
[18] 侯红霞, 杨波, 张丽娜, 张明瑞. 安全的两方协作SM2签名算法. 电子学报, 2020, 48(1): 1–8. [doi: 10.3969/j.issn.0372-2112.2020.01.001]
[19] 罗正钦. 基于UC框架的安全协议形式化分析[硕士学位论文]. 上海: 上海交通大学, 2008.

严都力(1995－), 女, 硕士, 助教, 主要研究领域为密码学, 区块链安全.

王文发(1968－), 男, 教授, CCF 高级会员, 主要研究领域为算法分析与设计, 软件项目研发.

谢敏(1997－), 男, 硕士生, 主要研究领域为区块链, 云存储安全.

禹勇(1980－), 男, 博士, 教授, CCF 高级会员, 主要研究领域为公钥密码理论及应用, 区块链, 密码货币, 云计算安全.

赵艳琦(1992－), 男, 博士, 副教授, CCF 专业会员, 主要研究领域为密码学, 区块链安全.