安全高效的两方协同 ECDSA 签名方案

王婧[1]，吴黎兵[1,2]，罗敏[2]，何德彪[2]
（1. 武汉大学计算机学院，湖北 武汉 430070；2. 武汉大学国家网络安全学院，湖北 武汉 430070）

摘 要：为了解决签名私钥易泄露和签名权利过度集中的问题，针对基于区块链技术的网络交易系统，提出了一种安全高效的两方协同 ECDSA 签名方案。通过预计算一次一密的 Beaver 三元组，进而利用基于 Beaver 三元组的安全两方乘法技术，有效避免使用计算繁重的同态加密和通信开销较大的不经意传输等操作，实现高效的两方协同 ECDSA 签名，保证 2 个签名参与方在不重构完整签名私钥的情况下输出合法的 ECDSA 签名。方案的安全性在通用可组合框架中的混合模型下被证明。理论分析与实验结果表明，与现有的 2 种两方协同 ECDSA 签名方案相比，所提方案在协同签名运行效率和带宽要求方面均具有明显优势。

关键词：私钥泄露；密钥保护；签名效率；两方签名
中图分类号：TP309
文献标识码：A
DOI: 10.11959/j.issn.1000−436x.2021019

Secure and efficient two-party ECDSA signature scheme

WANG Jing[1], WU Libing[1,2], LUO Min[2], HE Debiao[2]
1. School of Computer Science, Wuhan University, Wuhan 430070, China
2. School of Cyber Science and Engineering, Wuhan University, Wuhan 430070, China

Abstract: To solve the easy disclosure of signature private key and excessive concentration of signature rights, a secure and efficient two-party ECDSA signature scheme was proposed for the blockchain based network trading systems. By pre-computing one-time pad Beaver's triple, and utilizing the Beaver's triple based secure two-party multiplication technology, some computationally intensive homomorphic encryption operations and oblivious transfer operations with high communication overhead were effectively avoided, and thereby an efficient two-party ECDSA signing was realized, which could ensure that the two signing parties output valid ECDSA signature without reconstructing the complete private key. The proposed scheme was proved to be provably secure under the hybrid model of the universally composable framework. Theoretical analysis and simulation results demonstrate that the proposed scheme has significant advantages in terms of signing efficiency and bandwidth requirements when compared with the existing two two-party ECDSA signature schemes.

Keywords: private key leakage, key protection, signing efficiency, two-party signature

1 引言

椭圆曲线数字签名算法（ECDSA, elliptic curve digital signature algorithm）是椭圆曲线加密（ECC, elliptic curve cryptography）与数字签名算法（DSA, digital signature algorithm）的结合，于 1999 年成为

收稿日期：2020−09−02；**修回日期**：2020−12−05
通信作者：吴黎兵，whuwlb@126.com
基金项目：国家自然科学基金资助项目（No.61932016, No.61972294, No.61772377, No.61672257, No.91746206）；湖北省自然科学基金资助项目（No.2017CFA007）；深圳市科技计划基金资助项目（No.JCYJ20170818112550194）
Foundation Items: The National Natural Science Foundation of China (No.61932016, No.61972294, No.61772377, No.61672257, No.91746206), The Natural Science Foundation of Hubei Province (No.2017CFA007), The Science and Technology Planning Project of Shenzhen (No.JCYJ20170818112550194)

美国国家标准学会（ANSI, America National Standards Institute）标准，并于 2000 年成为电气和电子工程师协会（IEEE, Institute of Electrical and Electronics Engineers）、美国国家标准与技术研究院（NIST, National Institute of Standards and Technology）标准[1]。与 DSA 和 RSA（Rivest-Shamir-Adleman）相比，ECDSA 具有计算量小、处理速度快、存储空间占用小、带宽要求低等特点，适用于计算能力、存储空间、带宽与功耗受限的应用场景。因此，ECDSA 被广泛应用于电子商务系统[2]和其他网络领域，如安全传输层（TLS, transport layer security）协议[3]和域名系统安全扩展（DNSSEC, domain name system security extension）协议[4]，用于提供身份认证、数据完整性验证、不可否认性等安全服务。随着比特币系统的成功部署与应用，ECDSA 受到更广泛的关注，并逐渐成为当前主流区块链平台及项目的默认签名机制，如以太坊和 Hyperledger Fabric。

众所周知，数字签名方案的安全性依赖于签名者私钥的安全性。在区块链系统中，私钥是用户掌控其密码货币的关键，也是实施隐私保护方案的基础。用户发布交易时，需要先使用签名私钥对交易进行签名，只有被共识节点验证通过的交易才能成功上链，实现密码货币的转移或信息发布。恶意攻击者一旦非法窃取了区块链用户的签名私钥，就可以任意转移该用户的密码货币或冒充该用户发布非法交易信息。近年来，密码货币被盗事件频发，造成严重的经济损失[5-6]。与传统银行电子交易系统不同，一旦区块链密码货币被转移，就无法追回。即使用户已知私钥被滥用，成功上链的交易也无法撤回。2019 年 2 月，加拿某平台误将 102 枚比特币发送到私钥仅被已去世的首席执行官所知的冷钱包，导致这些比特币被永久锁死[7]。因此，防止私钥泄露并避免签名权利过度集中是当前基于签名技术的网络交易系统和区块链系统亟待解决的问题。

为了解决这一问题，常见的解决方法有多重签名、基于 Shamir 秘密共享的(t,n)−门限签名和基于安全两方/多方计算的签名。其中，多重签名通常发生在区块链上，区块链需要引入一种对多重签名进行编码的方法，并对区块链进行重塑以完成基于多重签名交易的发布与验证，现有区块链难以支持这种技术。此外，不同签名者在区块链上进行多次签名，访问结构（签名者的身份、数量）容易暴露在区块链上，从而可能导致用户隐私泄露，且签名长度随签名者数量呈线性增长，签名验签时需要多个公钥参与验证，开销较高[8]。基于 Shamir 秘密共享的(t, n)−门限签名往往需要借助一个可信中心为预先选定的签名群体分发私钥份额，并且需要至少 t 个签名方重构出完整的私钥才能完成签名。在实际情况下，颁发私钥的中心和签名时重构的私钥持有者便成为系统的安全瓶颈[9]。基于安全两方/多方计算的签名可在区块链下实施，不需要在链上执行所有过程，而且两方/多方计算签名中签名者信息被折叠成常规的区块链交易格式，因而能够兼容现有区块链系统并降低开销，并在一定程度上保证用户隐私。

MacKenzie 等[10]首次提出了针对 DSA 的两方协同签名方案，可以使 2 个签名参与方对给定的公钥生成有效的 DSA 签名，而任何一方都不能单独完成签名，签名过程中不需要重构私钥。Gennaro 等[11]针对比特币钱包安全提出了一种基于门限加法同态加密技术的(t,n)−门限 DSA/ECDSA 签名方案，并在恶意模型下证明了其安全性。Boneh 等[12]对文献[11]方案进行了性能优化，提出了基于 level-1 全同态加密的门限 DSA/ECDSA 签名方案。然而，这些方案均使用了分布式同态加密密钥生成技术，签名参与方的通信开销和计算开销都非常高，难以实际应用于处理能力受限的比特币钱包服务或计算能力受限的区块链客户端。

Lindell[13]提出了基于 Paillier 同态加密算法的两方协同 ECDSA 签名方案，与上述方案不同的是，该方案不需要执行分布式 Paillier 密钥算法，直接利用 Paillier 同态属性即可完成两方协同签名，提高了两方协同签名的运行效率。为了满足用户对安全的差异性需求，Doerner 等[14]提出了基于秘密共享和不经意传输技术的$(2,n)$−门限 ECDSA 签名方案，该方案不需要引入计算开销较高的 Paillier 同态加密及其相关的零知识证明，从而大大提高了协同签名的计算性能。但是 Doerner 方案引入了不经意传输协议[15-16]，与 Lindell 方案[13]相比，增加了近千倍的通信开销，且不能应用于带宽受限的区块链客户端。Castagnos 等[17]使用哈希证明系统技术代替 Lindell 方案[13]中的 Paillier 同态加密技术，设计了一种新的两方协同 ECDSA 签名方案，进而完善了 Lindell 方案[13]的安全性证明。然而，对于 128 bit 安全的 ECDSA 签名方案，Castagnos 方案[17]的运行效率比 Lindell 方案[13]更低。

除两方协同的 ECDSA 签名方案之外，多方门限 ECDSA 签名方案也陆续被提出。Lindell 等[18]设计了一种安全隐私乘法协议，用于将原始的两方协同签名方案扩展成多方门限签名方案。Doerner 等[19]同样将 Lindell[13]原始的两方门限 ECDSA 签名方案扩展成了多方门限 ECDSA 签名方案。Gennaro 等[20]设计了一种基于 Paillier 同态加密的 MtA 协议，该协议可以将基于乘法的共享份额转换为基于加法的共享份额，进而基于该协议提出一种新的多方门限 ECDSA 协同签名方案。但是，这些方案仍然沿用了两方协同签名方案中所使用的同态加密算法或不经意传输协议，因而计算开销和通信开销非常高。

除 ECDSA 之外，国内外学者还提出了对其他密码算法的私钥保护安全解决方案。针对基于身份的数字签名，He 等[21]和 Feng 等[22]分别提出针对 IEEE P1363 标准的两方协同签名方案和多方协同签名方案。侯红霞等[9]针对我国商用密码算法 SM2，结合 Lindell 方案[13]的思想设计了一种两方协同 SM2 签名方案。Zhang 等[23]同样基于 Paillier 同态加密算法，设计了一种新的两方协同 SM2 签名方案。Mu 等[24]针对我国商用密码算法 SM9，设计了基于 Paillier 的两方协同 SM9 签名方案。然而，这些方案一方面计算开销或通信开销较高，难以适用于计算能力或带宽受限的客户端；另一方面，难以兼容现有的区块链系统。

因此，为了提高协同签名的运行效率、降低签名过程的通信开销、保证签名私钥安全并兼容当前区块链系统，对两方/多方协同 ECDSA 签名方案进行进一步研究具有非常重要的理论意义和现实意义。针对这一目标，并考虑到两方签名对客户端的友好性与便捷性，本文设计了一种新的安全高效两方协同 ECDSA 签名方案。本文的主要贡献可分为以下 3 个方面。

1) 设计了一种安全高效的两方协同 ECDSA 签名方案，通过预计算 Beaver 三元组[25]，避免使用计算开销高昂的 Paillier 同态加密和通信开销高昂的不经意传输技术，2 个参与方在不暴露各自私钥的情况下共同完成签名。

2) 对本文所提方案提供了完整的安全性证明，结果表明，所提方案在不同时腐化 2 个签名方的情况下能够有效保护 ECDSA 签名私钥。

3) 从理论分析和模拟验证 2 个方面对本文所提方案进行了评估，并与 2 个相关的两方协同签名进行了比较，结果表明，本文所提方案在计算开销和通信开销上都具有明显优势。

2 预备知识

2.1 ECDSA 数字签名算法

系统建立 输入安全参数，输出算法公开参数 $\text{param}=\{E,G,P,p,q,h\}$，其中，$E$ 为定义在有限域 F_p 上的椭圆曲线，p 为一个素数，G 为椭圆曲线上所有整数点构成的加法群，P 和 q 分别为群 G 的生成元和素数阶，h 为输入映射到 Z_q^* 域的杂凑函数，即 $h:\{0,1\}^*\to Z_q^*$。Z_q^* 域由整数集合 $\{1,2,\cdots,q-1\}$ 构成。

密钥生成 输入算法公开参数 param，输出签名私钥 d 和验证公钥 Q，其中 $d\in Z_q^*$ 为随机秘密值，公钥 $Q=dP$ 可公开发布。

签名生成 输入算法公开参数 param、用户私钥 d 和待签名消息 m，输出签名 $\sigma=(r,s)$。具体步骤如下。

1) 选择一个随机数 $k\in Z_q^*$，计算 $R=k\cdot P=(r_x,r_y)$，其中 r_x 和 r_y 分别为椭圆曲线上点 R 的横坐标和纵坐标。

2) 计算 $r=r_x \bmod q$，若 $r=0$，则返回步骤 1)；否则，执行步骤 3)。

3) 计算 $s'=k^{-1}(e+dr) \bmod q$，其中，$e=h(m)$ 为消息摘要。

4) 输出签名信息 $\sigma=(r,s)$，其中 $s=\min\{s',q-s'\}$，$\min\{\}$ 函数表示取集合中较小的数值。

签名验证 输入待验证的消息 m 和签名 σ，输出验证结果“1”或“0”。具体步骤如下。

1) 解析签名 σ 获得 (r,s)，计算摘要 $e=h(m)$。

2) 计算 $R_v=s^{-1}(eP+rQ)=(x_v,y_v)$。

3) 若 $r=x_v \bmod q$，输出 1；否则输出 0。

2.2 Beaver 三元组乘法技术

Beaver 三元组乘法技术由 Beaver[25]于 1991 年首次提出，通过完全随机地设置电路中每个门的输入，然后进行校正的方法实现安全多方乘法计算。该方法避免了标准的安全两方或多方乘法协议中涉及的零知识证明或进一步秘密值共享等操作/协议，只需要进行简单的消息随机秘密值广播和重构即可实现安全两方或多方乘法计算。为了简便，本文将在后续描述中省略整数运算涉及的“$\bmod q$”

符号。

假设参与方 U_1 和参与方 U_2 已分别存有各自的秘密共享份额 $\{a_i,b_i,\Delta a_i,\Delta b_i,\Delta c_i\}$，$i\in\{1,2\}$，其中 Δa_i 和 Δb_i 是完全随机的整数，整数 Δc_1 和 Δc_2 满足条件 $\Delta c_1+\Delta c_2=(\Delta a_1+\Delta a_2)(\Delta b_1+\Delta b_2)$，$U_1$ 和 U_2 的目标为安全计算 $c=c_1+c_2=(a_1+a_2)(b_1+b_2)$。基于 Beaver 三元组的安全两方乘法协议步骤如下。

U_1 先分别计算其盲化数据 $u_1=a_1-\Delta a_1$ 和 $v_1=b_1-\Delta b_1$，再发送 (u_1,v_1) 给 U_2。

U_2 先分别计算其盲化数据 $u_2=a_2-\Delta a_2$ 和 $v_2=b_2-\Delta b_2$，再发送 (u_2,v_2) 给 U_1。

U_1 进而可计算中间数据 $u=u_1+u_2$、$v=v_1+v_2$ 和 $c_1=\Delta c_1+\Delta a_1 v+\Delta b_1 u$。

U_2 同理可计算中间数据 $u=u_1+u_2$、$v=v_1+v_2$ 和 $c_2=\Delta c_2+\Delta a_2 v+\Delta b_2 u+uv$。

最后，U_1 和 U_2 交换各自的信息 c_1 和 c_2，从而两方均可计算结果 $c=c_1+c_2=(a_1+a_2)(b_1+b_2)$。

正确性 令 $c=c_1+c_2$，$a=a_1+a_2$，$b=b_1+b_2$，$\Delta c=\Delta c_1+\Delta c_2$，$\Delta a=\Delta a_1+\Delta a_2$，$\Delta b=\Delta b_1+\Delta b_2$，则 $u=a-\Delta a$，$v=b-\Delta b$，进一步地，通过以上步骤可得

$$\begin{aligned}&c_1+c_2=\Delta c_1+\Delta a_1 v+\Delta b_1 u+\\&\Delta c_2+\Delta a_2 v+\Delta b_2 u+uv=(\Delta c_1+\Delta c_2)+\\&(\Delta a_1+\Delta a_2)v+(\Delta b_1+\Delta b_2)u+uv=\\&\Delta c+\Delta a(b-\Delta b)+\Delta b(a-\Delta a)+\\&(a-\Delta a)(b-\Delta b)=\Delta c+b\Delta a-\Delta a\Delta b+\\&a\Delta b-\Delta a\Delta b+ab-b\Delta a-a\Delta b+\\&\Delta a\Delta b=ab=(a_1+a_2)(b_1+b_2)=c\end{aligned}\tag{1}$$

因此，利用 Beaver 三元组可以进行两方乘法计算。

安全性 文献[25]指出 $\{\Delta a_i,\Delta b_i\}$ 是随机选取的，且每组数据仅使用一次，相当于独立且均匀分布的一次一密随机值，允许通信方在公开信道上发送具有完全隐私性的任意消息。此外，Beaver 分别证明了该方法在消息被篡改（详见文献[25]的安全性证明引理 4 和引理 6）和参与方被腐化（详见文献[25]的引理 5 和引理 7）情况下的安全性。因此，本文认为该方案在不需要引入承诺与零知识证明等额外密码原语的情况下，满足半诚实模型和恶意模型下的安全性。

一次性表 Beaver[25]提出了一次性表的概念，用以描述一系列支持安全两方或多方计算的三元组对集合，如 $\mathrm{SET}_{\mathrm{tri}}=\{(\Delta a_{1i},\Delta b_{1i},\Delta c_{1i}),(\Delta a_{2i},\Delta b_{2i},\Delta c_{2i})\}$，其中 $i=\{1,2,\cdots,N\}$，N 表示集合大小。其中 U_1 将获得大小为 N 的三元组集合 $\mathrm{SET}_{\mathrm{tri}}^1=\{(\Delta a_{1i},\Delta b_{1i},\Delta c_{1i})\}_{i\in\{1,2,\cdots,N\}}$，对称地，$U_2$ 将获得大小为 N 的三元组集合 $\mathrm{SET}_{\mathrm{tri}}^2=\{(\Delta a_{2i},\Delta b_{2i},\Delta c_{2i})\}_{i\in\{1,2,\cdots,N\}}$。此外，为了提高计算效率与安全性，Beaver 建议通过离线预计算的方式来获取 $\mathrm{SET}_{\mathrm{tri}}$。

安全离线预计算协议 $\mathcal{F}_{\mathrm{pre}}$ 在三元组构造过程中，$(\Delta a_1,\Delta b_1)$ 和 $(\Delta a_2,\Delta b_2)$ 分别由 U_1 和 U_2 随机选取，然后通过交互计算得到 c_1 和 c_2，可以观察到

$$\begin{aligned}&(\Delta a_1+\Delta a_2)(\Delta b_1+\Delta b_2)=\\&\Delta a_1\Delta b_1+\Delta a_1\Delta b_2+\Delta a_2\Delta b_1+\Delta a_2\Delta b_2\end{aligned}\tag{2}$$

其中，$\Delta a_1\Delta b_1$ 和 $\Delta a_2\Delta b_2$ 可分别由 U_1 和 U_2 本地计算，因此构建三元组的难点在于如何计算 $\Delta a_1\Delta b_2$ 和 $\Delta a_2\Delta b_1$。Feng 等[26]基于现有成果列举了 3 种生成 Beaver 三元组的安全预计算方法。①首先基于加法同态加密技术构造一个乘法共享转加法共享的协议，再基于此计算 $\Delta a_1\Delta b_2$ 和 $\Delta a_2\Delta b_1$。以基于 Paillier 同态加密的方法为例：U_1 首先生成 Paillier 的公私钥对 $(\mathrm{sk},\mathrm{pk})$，计算密文 $x_1=\mathrm{Enc}_{\mathrm{pk}}(\Delta a_1)$，将 x_1 和公钥 pk 发送给 U_2；U_2 收到消息后选取一个随机数 η，计算密文 $x_2=x_1^{\Delta b_2}\mathrm{Enc}_{\mathrm{pk}}(-\eta)$，再将密文 x_2 发送给 U_1；U_1 可解密获得 $y_1=\Delta a_1\Delta b_2-\eta$；类似地，重复上述步骤，$U_1$ 可获得 $y_1'=\Delta a_2\Delta b_1-\eta'$，进而可以计算出 $\Delta c_1=\Delta a_1\Delta b_1+y_1+y_1'$；$U_2$ 根据计算过程中选取的随机数可计算 $\Delta c_2=\Delta a_2\Delta b_2+\eta+\eta'$。该方法首次在文献[20]中被应用，并且其安全性也得到了证明。②与第一种方法类似，先计算 $\Delta a_1\Delta b_2$ 和 $\Delta a_2\Delta b_1$，再计算 Δc_1 和 Δc_2，不同点在于用不经意传输协议代替加法同态加密技术以节约计算开销，具体实施方案可参考文献[14]。③基于可信第三方颁发有效的 Beaver 三元组，如利用一个可信的服务器为 U_1 和 U_2 生成匹配的三元组，该方法在文献[27]中被应用。后文为了简便，令 $\mathcal{F}_{\mathrm{pre}}$ 表示安全的三元组离线预计算协议。

3　安全模型和设计目标

3.1　安全性定义

本节主要描述数字签名方案与两方协同数字签名方案的安全性定义。

1) 标准数字签名安全性定义

与文献[13]类似，出于完整性和参考性目的，

本文首先针对标准数字签名方案 $\pi=(\mathrm{Gen},\mathrm{Sign},\mathrm{Verify})$ 给出一个基于游戏的安全性定义。游戏 $\mathrm{GameSign}_{\mathcal{A},\pi}(1^\lambda)$ 中共 2 个角色，即概率多项式时间（PPT, probabilistic polynomial time）敌手 $\mathcal{A}$ 和模拟器 $\mathcal{C}$。其中，$\mathcal{C}$ 为 PPT 敌手 $\mathcal{A}$ 提供方案相关的公开参数，包括签名验证公钥 Q，同时提供对 $\mathcal{A}$ 的签名查询应答；PPT 敌手 $\mathcal{A}$ 的目标是在进行若干消息−签名对查询后，伪造消息 m^* 的签名 σ^*；若伪造成功，则游戏输出“1”，否则游戏输出“0”。$\mathcal{A}$ 和 $\mathcal{C}$ 的游戏流程如算法 1 所示。

算法 1 标准数字签名

游戏 1 $\mathrm{GameSign}_{\mathcal{A},\pi}(1^\lambda)$

① $(d,Q)\leftarrow\mathrm{Gen}(1^\lambda)$

② $\mathcal{A}$ 将消息 $m\in\{01\}^*$发送给 $\mathcal{C}$ 进行签名询问 $\mathcal{C}^{\mathrm{Sign}_{d(\cdot)}}$，$\mathcal{C}$ 返回 m 的合法签名 σ；/*该步骤执行次数在多项式范围内，所有查询过的消息 m 构成集合 Ω*/

③ $(m^*,\sigma^*)\leftarrow\mathcal{A}^{\mathrm{Sign}_{d(\cdot)}}(1^\lambda,Q)$; /*敌手 $\mathcal{A}$ 进行签名伪造*/

④ 当且仅当 $\mathrm{Verify}(m^*,\sigma^*)=1$ 且 $m^*\notin\Omega$ 时，游戏输出“1”；否则游戏输出“0”

定义 1 如果对于任意的 PPT 敌手 $\mathcal{A}$，在游戏 1 中输出“1”的概率是可忽略的，即

$$\Pr[\mathrm{GameSign}_{\mathcal{A},\pi}(1^\lambda)=1]\leqslant\varepsilon \tag{3}$$

其中，ε 为一个可忽略概率阈值，λ 为签名方案 π 的安全参数，则认为签名方案 π 满足选择消息攻击下的存在不可伪造性安全（EU-CMA, existential unforgeability against chosen-message attack）。

2) 两方协同签名安全性定义

两方协同签名可视为标准数字签名的一种分布式表现形式，其输出的签名仍能通过算法 Verify 的验证，因此，本文采用类似于签名方案 π 的方式定义 π 衍生的两方协同签名方案 $\Pi=(\mathrm{DistGen},\mathrm{DistSign},\mathrm{Verify})$ 的安全性。

与标准数字签名安全定义不同的是，本文假设 PPT 敌手 $\mathcal{A}$ 可以控制其中一个签名参与方 $U_i(i\in\{1,2\})$，并且知道 U_i 的所有秘密值。因此，$\mathcal{A}$ 和被腐化的 U_i 被认为是一体的，在后续描述中称 U_i 为腐化方或直接用 $\mathcal{A}$ 代替 U_i 来描述游戏模拟过程。在模拟过程中，$\mathcal{A}$ 选择要签名的任意信息，与诚实参与方 $U_j(j\neq i)$ 进行交互生成签名。同时，$\mathcal{A}$ 可以选择遵循协议设置或任意偏离协议设置，即敌手是恶意的。此外，敌手 $\mathcal{A}$ 是静态的，即在协议初始化之前，敌手 $\mathcal{A}$ 便确定被控制的参与方为 U_i，且在一个游戏结束之前不会转换为参与方 U_j。

令 $\mathrm{GameDistSign}^i_{\mathcal{A},\Pi}(1^\lambda)$ 表示两方协同签名方案 Π 的安全性模拟游戏，并且在该游戏中包含预言机 $\Pi_i(\cdot)$ 和敌手 $\mathcal{A}$ 这 2 个角色。$\Pi_i(\cdot)$ 是一个状态预言机，执行签名诚实参与方 $U_j(j\neq i)$ 的协议指令，且规定游戏模拟中只允许一次密钥生成过程，但可以多次运行签名过程。$\mathcal{A}$ 是具有上述能力的 PPT 敌手，其目标是在进行若干消息−签名对查询后，伪造消息 m^* 的签名 σ^*。若 $\mathcal{A}$ 伪造签名成功，则游戏输出“1”，否则游戏输出“0”。值得注意的是，$\mathcal{A}$ 在游戏模拟过程中可以获取被控制方 U_i 的所有视图，包含输入、随机数和输出，并可能从中获得对自己有利的信息。$\mathcal{A}$ 和 $\Pi_i(\cdot)$ 的游戏流程如算法 2 所示。

算法 2 两方协同数字签名

游戏 2 $\mathrm{GameDistSign}_{\mathcal{A},\Pi}(1^\lambda)$

① $Q\leftarrow\Pi_i(1^\lambda)$；/*诚实参与方输出签名验证公钥，$U_i$ 被敌手控制*/

② $\mathcal{A}$ 选择 $m\in\{01\}^*$作为待签名的消息，与 $\Pi_i(\cdot)$进行交互输出签名 σ；/*该步骤执行次数在多项式范围内，所有查询过的消息 m 构成集合 Ω*/

③ $(m^*,\sigma^*)\leftarrow A^{\Pi_{i(\cdot)}}(1^\lambda)$;/*敌手 $\mathcal{A}$ 进行签名伪造*/

④ 当且仅当 $\mathrm{Verify}(m^*,\sigma^*)=1$ 且 $m^*\notin\Omega$ 时，游戏输出“1”；否则游戏输出“0”

定义 2 如果对于任意的 PPT 敌手 $\mathcal{A}$ 以及被 $\mathcal{A}$ 腐化的任意参与方 U_i $(i\in\{1,2\})$，在游戏 2 中输出“1”的概率是可忽略的，即

$$\Pr[\mathrm{GameDistSign}^i_{\mathcal{A},\Pi}(1^\lambda)=1]\leqslant\varepsilon \tag{4}$$

其中，ε 为一个可忽略概率阈值，则认为签名方案 Π 满足选择消息攻击下的存在不可伪造性安全。

3.2 安全模型

通用可组合（UC, universally composable）安全是由 Canetti[28]提出的一种根据协议现实执行环境和协议理想执行环境不可区分方法来定义密码协议安全性的框架。该框架的显著特点是：当一个组合协议的各个子协议被证明是 UC 安全的，那么该组合协议是安全的。因此，可以模块化地选取或设计可证明安全的子协议，进而运用组合原理构造更复杂的安全密码协议。

在 UC 框架下，理想函数 $\mathcal{F}$ 是重要的组成部分，代表一个不可腐化的可信方，用于完成协议所需的理想功能与执行过程。PPT 敌手 $\mathcal{A}$ 与协议参与方的交互用于模拟协议的现实执行过程，即协议现实执行环境。协议参与方（包括可视为理想敌手的模拟器 $\mathcal{C}$）与理想函数的交互用于模拟协议的理想执行过程，即协议理想执行环境。本文提出的两方协同 ECDSA 签名方案是基于理想函数 $\mathcal{F}_{\text{com}}$、$\mathcal{F}_{\text{zk}}^{\mathcal{R}}$ 和 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}}$ 组合的混合模型构造的。同时，本文方案的安全性也是基于此模型进行证明的。其中，分布式密钥生成模拟只需执行一次，两方协同签名模拟可执行多项式次数。值得注意的是，如果承诺与零知识证明协议是 UC 安全的，那么在混合模型下协议模拟执行的输出和协议现实执行的输出在计算上是不可区分的。接下来，本文将对这几个理想函数进行简单介绍。

1) 承诺理想函数 $\mathcal{F}_{\text{com}}$

在随机预言机模型下，$\mathcal{F}_{\text{com}}$ 可通过简单地定义函数 $\text{Com}(x)=h(x,r)\leftarrow\{0,1\}^{\lambda}$ 来实现静态安全，且任意满足通用可组合安全的承诺函数 h 都满足 $\mathcal{F}_{\text{com}}$ 的形式化定义[13]，具体流程如下。

① 当收到参与方 U_i 的消息 $(\text{commit},\text{sid},x)$ 后，如果已经存储会话序号 sid 相关的承诺消息 $(\text{commit},\text{sid},*)$，则忽略此消息；否则，记录消息 (sid,i,x)，并且将 $(\text{receipt},\text{sid})$ 发送给参与方 U_j。

② 当收到参与方 U_i 的消息 $(\text{decommit},\text{sid})$ 后，如果已经存储消息 (sid,i,x)，则将消息 $(\text{decommit},\text{sid},x)$ 发送给参与方 U_j。

2) 零知识证明理想函数 $\mathcal{F}_{\text{zk}}^{\mathcal{R}}$

根据文献[13, 18]可知，标准零知识理想函数可形式化定义为

$$((x,\omega),\text{str})\rightarrow(\text{str},(x,\mathcal{R})(x,\omega))$$

其中，str 表示空字符串，$\mathcal{R}$ 表示秘密信息 x 和证据 ω 之间的关系。在安全两方计算场景下，非交互式零知识证明理想函数 $\mathcal{F}_{\text{zk}}^{\mathcal{R}}$ 与 2 个参与方 U_i、U_j 的形式化交互流程如下。

当收到参与方 U_i 的消息 $(\text{prove},\text{sid},x,\omega)$ 后，如果会话序号 sid 已被使用过或秘密信息 x 和证据 ω 之间不满足关系 $\mathcal{R}$，即 $(x,\omega)\notin\mathcal{R}$，则忽略本次的消息；否则，将 $(\text{proof},\text{sid},x)$ 发送给参与方 U_j。

3) 承诺的非交互式证明理想函数 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}}$

$\mathcal{F}_{\text{com-zk}}^{\mathcal{R}}$ 的构造基于承诺函数和非交互式零知识证明，要求发送方首先发送对零知识证明的承诺，然后打开承诺，接收者验证零知识证明的合法性。在安全两方计算场景下，$\mathcal{F}_{\text{com-zk}}^{\mathcal{R}}$ 与参与方 U_i、U_j 的形式化交互流程如下。

① 当收到参与方 U_i 的消息 $(\text{com-prove},\text{sid},x,\omega)$ 后，如果会话序号 sid 已被使用过或 $(x,\omega)\notin\mathcal{R}$，则忽略本次的消息；否则，将 $(\text{proof-receive},\text{sid})$ 发送给参与方 U_j。

② 当收到参与方 U_i 的消息 $(\text{decom-proof},\text{sid})$ 后，如果已存储了 (sid,i,x)，则将 $(\text{decom-proof},\text{sid},x)$ 发送给参与方 U_j。

在本文的方案设计中，$\mathcal{R}$ 为椭圆曲线上的离散对数关系，定义为 $\mathcal{R}_{\text{DL}}=\{(G,P,q,Q,x)\,|\,Q=xP\}$。对于关系 $\mathcal{R}_{\text{DL}}$ 的零知识证明，本文可以采用经典的 Schnorr 证明[29]来实现。

3.3 设计目标

本文提出的两方协同 ECDSA 签名方案应满足以下 4 个属性。

① 隐私性。除协议的计算输出和协议本身所泄露的内容外，2 个签名参与方中的任何一方都不能从协议执行过程中获取其他任何信息，包括另一方的私有输入。

② 正确性。2 个签名参与方诚实地执行该协议时，协议输出合法的 ECDSA 签名。

③ 兼容性。协议输出的正确签名仍能通过原签名方案验证算法的检测，即使用 2.1 节描述的签名验证算法时，签名验证结果为“1”。

④ 高效性。充分考虑参与方的计算能力以及协同签名过程中的在线时延，签名过程应保证计算开销和通信开销尽可能小。

4 方案设计

本文提出的两方协同 ECDSA 签名方案共包括 4 个阶段，分别为系统建立、分布式密钥生成、两方协同签名和签名验证。系统建立和签名验证与 2.1 节中的描述一致，因此本节不再重复描述。分布式密钥生成和两方协同签名协议由 2 个签名参与方 U_1 和 U_2 协同完成，其中分布式密钥生成算法只需运行一次，两方协同签名算法可运行多次。

在本文方案中，假设在运行分布式密钥生成和协同签名协议之前，系统已执行过（如第 2.2 节所描述的）安全离线预计算协议 $\mathcal{F}_{\text{pre}}$，并安全输出了

签名参与方 U_1 和 U_2 所需的 Beaver 三元组集合 $\mathrm{SET}_{\mathrm{tri}}^1$ 和 $\mathrm{SET}_{\mathrm{tri}}^2$。换句话说，$U_1$ 和 U_2 在签名之前已秘密保存了配对的三元组集合 $\mathrm{SET}_{\mathrm{tri}}^1$ 和 $\mathrm{SET}_{\mathrm{tri}}^2$，从而降低了在线协同签名的通信开销与计算开销。Beaver 三元组的离线预计算理念已在多个方案中被应用，如文献[26-27, 30]。根据文献[25]的定义，为了保证协议在 UC 框架下的安全性，Beaver 三元组应为一次一密随机值。因此，本文可根据实际应用场景需求，选择定期执行安全预计算协议来更新集合 $\mathrm{SET}_{\mathrm{tri}}^1$ 和 $\mathrm{SET}_{\mathrm{tri}}^2$，以获得足够使用的三元组数量，或者一次性预计算出足够多的三元组。下面，将具体介绍本文提出的两方协同 ECDSA 签名方案的分布式密钥生成协议与两方协同签名协议。

4.1 分布式密钥生成

分布式密钥生成协议由签名参与方 U_1 和 U_2 共同完成，输出签名验证公钥 Q 和各自的部分私钥 d_1, d_2。

1) $U_1 \to U_2 : \{\mathrm{com}(Q_1, \pi_1)\}$

① U_1 选取一个随机数 $d_1 \in Z_q^*$ 作为 U_1 的部分私钥，计算对应的部分公钥 $Q_1 = d_1 P$。

② U_1 发送消息 $(\text{com-prove}, 1, Q_1, d_1)$ 给理想函数 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_1 发送关于 Q_1 及其离散对数的零知识证明 π_1 的承诺 com 给 U_2）。

2) $U_2 \to U_1 : \{Q_2, \pi_2\}$

①当 U_2 接收到 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 (proof-receipt,1) 后，U_2 选取一个随机数 $d_2 \in Z_q^*$ 作为 U_2 的部分私钥，计算对应的部分公钥 $Q_2 = d_2 P$。

② U_2 发送消息 $(\text{prove}, 2, Q_2, d_2)$ 给理想函数 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_2 发送 Q_2 和关于 Q_2 的离散对数零知识证明 π_2 给 U_1）。

3) $U_1 \to U_2 : \{Q_1, \pi_1\}$

当 U_1 接收到 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 $(\text{proof}, 2, Q_2)$ 后，U_1 验证 proof 的有效性，若无效则协议终止；否则，打开承诺并发送消息 (decom-proof,1) 给 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_1 发送 Q_1 和关于 Q_1 的离散对数零知识证明 π_1 给 U_2）。

4) U_1 和 U_2 输出签名验证公钥 Q，并安全存储各自的隐私信息

① U_1 计算签名验证公钥 $Q = Q_1 + Q_2$，并安全存储 $(d_1, \mathrm{SET}_{\mathrm{tri}}^1)$。

② 当 U_2 接收到理想函数 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 $(\text{decom-proof}, 1, Q_1)$ 后，U_2 验证 proof 的有效性，若无效则协议终止；否则，计算签名验证公钥 $Q = Q_1 + Q_2$，并安全存储 $(d_2, \mathrm{SET}_{\mathrm{tri}}^2)$。

4.2 两方协同签名

给定待签名的消息 m，两方协同签名协议必须由参与方 U_1 和 U_2 共同完成，输出 ECDSA 签名 $\sigma = (r, s)$。

1) $U_1 \to U_2 : \{\mathrm{com}(R_1, \pi_3)\}$

① U_1 选取一个随机数 $k_1 \in Z_q^*$，计算 $R_1 = k_1 P$。

② U_1 发送消息 $(\text{com-prove}, \mathrm{sid} \| 1, R_1, k_1)$ 给理想函数 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_1 发送关于 R_1 及其离散对数的零知识证明 π_3 的承诺 com 给 U_2）。

2) $U_2 \to U_1 : \{R_2, \pi_4\}$

① 当 U_2 接收到 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 $(\text{proof-receipt}, \mathrm{sid} \| 1)$ 后，U_2 选取一个随机数 $k_2 \in Z_q^*$，计算 $R_2 = k_2 P$。

② U_2 发送消息 $(\text{prove}, \mathrm{sid} \| 2, R_2, k_2)$ 给理想函数 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_2 发送 R_2 和关于 R_2 的离散对数零知识证明 π_4 给 U_1）。

3) $U_1 \to U_2 : \{R_1, \pi_3, (u_1, v_1, w_1, t_1)\}$

① 当 U_1 接收到 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 $(\text{proof}, \mathrm{sid} \| 2, R_2)$ 后，U_1 验证 proof 的有效性，若无效则协议终止；否则，打开承诺并发送消息 $(\text{decom-proof}, \mathrm{sid} \| 1)$ 给 $\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\mathrm{DL}}}$（即 U_1 发送 R_1 和关于 R_1 的离散对数零知识证明 π_3 给 U_2）。

② U_1 依次计算 $R = R_1 + R_2 = (r_x, r_y)$、$r = r_x \bmod q$、$\delta_1 = \dfrac{e}{2} + rd_1$，其中，$e = h(m)$，如果 e 为奇数，U_1 进一步计算，令 $\delta_1 = \delta_1 + 1$。

③ U_1 先选择一个随机数，再按顺序选择 $\mathrm{SET}_{\mathrm{tri}}^1$ 中前 2 个三元组 $(\Delta a_{11}, \Delta b_{11}, \Delta c_{11})$ 和 $(\Delta a_{12}, \Delta b_{12}, \Delta c_{12})$，进而分别计算数据 $u_1 = k_1 - \Delta a_{11}$、$v_1 = \rho_1 - \Delta b_{11}$、$w_1 = \delta_1 - \Delta a_{12}$ 和 $t_1 = \rho_1 - \Delta b_{12}$。

④ U_1 发送消息 $\{u_1, v_1, w_1, t_1\}$ 给 U_2。

4) $U_2 \to U_1 : \{(u_2, v_2, w_2, t_2), (\alpha_2, \beta_2)\}$

① 当 U_2 接收到理想函数 $\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}}$ 的消息 $(\text{decom-proof}, \mathrm{sid} \| 1, R_1)$ 后，U_2 验证 proof 的有效性，若无效则协议终止；否则，依次计算 $R = R_1 + R_2 = (r_x, r_y)$、$r = r_x \bmod q$ 和消息杂凑值 $e = h(m)$，并且计算 $\delta_2 = \dfrac{e}{2} + rd_2$。

② 当U_2接收到U_1的消息(u_1,v_1,w_1,t_1)后，U_2先选择一个随机数，再按顺序选择$\mathrm{SET}_{\mathrm{tri}}^2$中前 2 个三元组$(\Delta a_{21},\Delta b_{21},\Delta c_{21})$和$(\Delta a_{22},\Delta b_{22},\Delta c_{22})$，进而分别计算数据$u_2=k_2-\Delta a_{21}$、$v_2=\rho_2-\Delta b_{21}$、$w_2=\delta_2-\Delta a_{22}$和$t_2=\rho_2-\Delta b_{22}$。

③ U_2先分别计算数据$u=u_1+u_2$、$v=v_1+v_2$、$w=w_1+w_2$、$t=t_1+t_2$，接着计算数据$\alpha_2=k_2v+\rho_2u+\Delta c_{21}$、$\beta_2=\delta_2t+\rho_2w+\Delta c_{22}$。

④ U_2删除当前$\mathrm{SET}_{\mathrm{tri}}^2$中的三元组$(\Delta a_{21},\Delta b_{21},\Delta c_{21})$和$(\Delta a_{22},\Delta b_{22},\Delta c_{22})$，然后将消息$\{(u_2,v_2,w_2,t_2),(\alpha_2,\beta_2)\}$发送给$U_1$。

5) U_1输出签名$\sigma=(r,s)$

① 当U_1收到U_2的消息$\{(u_2,v_2,w_2,t_2),(\alpha_2,\beta_2)\}$后，$U_1$先计算$u=u_1+u_2$、$v=v_1+v_2$、$w=w_1+w_2$、$t=t_1+t_2$、$\alpha_1=k_1v+\rho_1u+\Delta c_{11}-uv$和$\beta_1=\delta_1t+\rho_1w+\Delta c_{12}-tw$，然后删除$\mathrm{SET}_{\mathrm{tri}}^1$中的三元组$(\Delta a_{11},\Delta b_{11},\Delta c_{11})$和$(\Delta a_{12},\Delta b_{12},\Delta c_{12})$。

② U_2计算$s'=(\alpha_1+\alpha_2)^{-1}(\beta_1+\beta_2)$，令$s=\min\{s',q-s'\}$，输出 ECDSA 签名$\sigma=(r,s)$

若应用场景需要U_2也输出签名，U_1可将消息(α_1,β_1)给U_2，则U_2也可按照步骤 5)计算获得签名$\sigma=(r,s)$。由于在 Beaver 三元组乘法技术中，三元组必须遵循一次一密的原则才能满足多方计算的完美隐私性，因此U_1和U_2在每次签名之后删除已使用的三元组，避免签名过程中重复使用三元组。此外，当应用场景安全要求较低时，可删除方案涉及的所有承诺协议和零知识证明协议，即构成半诚实模型[31]下的安全两方协同 ECDSA 签名方案，从而进一步提高协同签名的效率。

4.3　正确性分析

根据分布式密钥生成算法可得，签名验证公钥为

$$Q=Q_1+Q_2=d_1P+d_2P=(d_1+d_2)P \tag{5}$$

根据两方协同签名算法可得

$$R=R_1+R_2=k_1P+k_2P=(k_1+k_2)P \tag{6}$$

此外，同理于 2.2 节中 Beaver 三元组乘法技术的正确性分析，即式(1)，可得

$$\begin{cases}\alpha_1+\alpha_2=(k_1+k_2)(\rho_1+\rho_2)\\ \beta_1+\beta_2=(\delta_1+\delta_2)(\rho_1+\rho_2)=[e+r(d_1+d_2)](\rho_1+\rho_2)\end{cases} \tag{7}$$

令$d=d_1+d_2$，$k=k_1+k_2$，$\rho=\rho_1+\rho_2$，可得

$$\begin{cases}Q=dP\\ R=kP=(r_x,r_y)\\ r=r_x \bmod q\\ s=(k\rho)^{-1}[(e+rd)\rho]=k^{-1}(e+rd)\end{cases} \tag{8}$$

由此可知，本文所提两方协同 ECDSA 签名方案输出的签名(r,s)和签名验证公钥Q与 2.1 节描述的 ECDSA 公钥签名(r,s)和验签公钥Q一致。因此，本文所提方案满足设计目标要求的正确性和兼容性。

5　安全性证明

本节将在$(\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\mathrm{DL}}},\mathcal{F}_{\mathrm{zk}}^{\mathcal{R}_{\mathrm{DL}}})$–混合模型下证明本文所提方案的安全性，并且在模拟中将对本文所提两方协同 ECDSA 签名方案的攻击归约到对原始 ECDSA 方案的攻击，即将本文所提两方协同 ECDSA 签名方案Π的安全性归约到原始 ECDSA 签名方案π的可证明安全性上。

引理 1　假设 ECDSA 签名方案满足 EU-CMA 安全性，那么本文所提两方协同 ECDSA 签名方案满足 EU-CMA 安全性。

证明　假设任意 PPT 敌手$\mathcal{A}$在腐化任意一个协同签名参与方$U_i(i\in\{1,2\})$的情况下，能够以不可忽略的概率ϵ在游戏$\mathrm{GameDistSign}_{\mathcal{A},\Pi}^{i}(1^{\lambda})$中成功伪造消息–签名对$(m^*,\sigma^*=(r^*,s^*))$，那么本文可以构造一个模拟器$\mathcal{C}$作为游戏$\mathrm{GameSign}_{\mathcal{C},\pi}(1^{\lambda})$的概率多项式敌手，使$\mathcal{C}$能够以与$\epsilon$几乎相等的概率在$\mathrm{GameSign}_{\mathcal{C},\pi}(1^{\lambda})$中成功伪造消息–签名对$(m^*,\sigma^*=(r^*,s^*))$。同理于文献[13]，本文将考虑敌手腐化$U_1$和敌手腐化$U_2$这 2 种情况，分别论述本文方案的安全性。其中，分布式密钥生成协议仅需模拟一次，两方协同签名协议可进行多次模拟。假设U_1和U_2在分布式密钥生成前已通过安全方式获得对应的$\mathrm{SET}_{\mathrm{tri}}^2$和$\mathrm{SET}_{\mathrm{tri}}^2$。由于 Beaver 三元组已被证明是安全的且满足完美隐私性[25]，因此，为了满足 UC 框架的安全模拟规则（即参与方之间、参与方与敌手间不直接交互），本文在模拟过程中引入一个 Beaver 三元组乘法理想函数$\mathcal{F}_{\mathrm{BT}}$，用于接收/发送与 Beaver 三元组直接相关的消息。

当$i=1$时，PPT 敌手$\mathcal{A}$腐化参与方U_1。本文构造一个针对游戏$\mathrm{GameDistSign}_{\mathcal{A},\Pi}^{1}(1^{\lambda})$的模拟器

$\mathcal{C}$，同时作为游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 的 PPT 敌手。本质上，$\mathcal{C}$ 在协议模拟过程产生一个令 $\mathcal{A}$ 不可区分的执行环境，然后利用 $\mathcal{A}$ 在游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中输出的伪造签名作为游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 的输出，以此攻击原始 ECDSA 签名方案。

1) 分布式密钥生成协议模拟

$\mathcal{C}$ 模拟游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中的预言机 $\Pi_1(\cdot)$，将当前状态查询设为 $(0,\cdot)$，并且仅在 $\mathcal{A}$ 向预言机 $\Pi_1(\cdot)$ 发送了分布式密钥生成协议模拟启动指令后，$\mathcal{C}$ 才能工作。此外，在该过程中不会对其他状态的查询 $(\text{sid},\cdot)$ 进行回答，即分布式密钥生成协议模拟结束之前，$\mathcal{C}$ 不会对两方协同签名协议模拟有关的任何询问进行回答。假设 $\mathcal{A}$ 已发送指令 $(0,0)$，则该过程的具体步骤如下。

① 在游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 中，$\mathcal{C}$ 向预言机 $\mathcal{F}_{\text{ECDSA}}$ 进行密钥生成查询，获得 $(1^\lambda,Q)$。其中，λ 为安全参数，Q 为 ECDSA 的验证公钥。

② 在游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中，当收到 $\mathcal{A}$ 发送给 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{com-zk}}$ 的消息 $(0,(\text{com-prove},1,Q_1,d_1))$ 后，$\mathcal{C}$ 验证 $Q_1=d_1P$ 是否成立，如果不成立，终止协议；否则计算 $Q_2=Q-Q_1$，然后模拟 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{zk}}$ 将 $(\text{proof},2,Q_2)$ 作为预言机回答发送给 $\mathcal{A}$。

③ 当收到 $\mathcal{A}$ 发送给 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{com-zk}}$ 的 $(\text{decom-proof},\text{sid}\|1)$ 后，若 U_2 不退出协议模拟，$\mathcal{C}$ 存储 $(d_1,Q,\text{SET}^1_{\text{tri}})$，然后结束分布式密钥生成协议。

2) 两方协同签名协议模拟

$\mathcal{C}$ 模拟游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中的预言机 $\Pi_1(\cdot)$，将当前状态查询设为 $(\text{sid},\cdot)$，sid 是一个新的会话标识符。给定一个待签名消息 m，$\mathcal{C}$ 在该过程的具体步骤如下。

① 在游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 中，$\mathcal{C}$ 向预言机 $\mathcal{F}_{\text{ECDSA}}$ 进行签名查询，获得 $\sigma=(r,s)$，并根据签名验证算法进一步计算得到 R。

在游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中，当收到 $\mathcal{A}$ 发送给 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{com-zk}}$ 的消息 $(\text{sid},(\text{com-prove},\text{sid}\|1,R_1,k_1))$ 后，$\mathcal{C}$ 验证 $R_1=d_1P$ 是否成立，如果不成立，则终止协议；否则计算 $R_2=R-R_1$，然后模拟 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{zk}}$ 将应答消息 $(\text{proof},\text{sid}\|2,R_2)$ 发送给 $\mathcal{A}$。

② 当收到 $\mathcal{A}$ 发送给 $\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{com-zk}}$ 的 $(\text{decom-proof},\text{sid}\|1)$ 和发送给 $\mathcal{F}_{\text{BT}}$ 的 $(\text{rand-value},\text{sid}\|1,u_1,v_1,w_1,t_1)$ 后，如果 $R_1=d_1P$ 不成立，$\mathcal{C}$ 模拟 U_2 终止协议；否则 $\mathcal{C}$ 先选择 5 个随机数 $(u_2,v_2,w_2,t_2,\alpha_2)\in Z_q^*$，依次计算 $u=u_1+u_2$、$v=v_1+v_2$、$w+w_1+w_2$、$t=t_1+t_2$、$\alpha_1=k_1v+\rho_1u+\Delta c_{11}-uv$、$\beta_1=\delta_1t+\rho_1w+\Delta c_{12}-tw$，然后计算 $\beta_2=s(\alpha_1+\alpha_2)-\beta_1$。$\mathcal{C}$ 将 $(u_2,v_2,w_2,t_2,\alpha_2,\beta_2)$ 发送给 $\mathcal{A}$。

不可区分性分析。在分布式密钥生成模拟过程中，可以观察到协同签名协议的现实执行环境与理想模拟执行环境仅有一个区别。在协议现实执行环境中，Q_2 是由诚实参与方 U_2 选择一个随机数后计算 $Q_2=d_2P$ 得来的；在协议模拟执行环境中，$\mathcal{C}$ 计算 $Q_2=Q-Q_1=Q-d_1P$，其中公钥 Q 是在游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 中查询预言机获取的。由于 Q 是随机选取的，因此 d_2P 和 $Q-d_1P$ 是同分布的。此外，若 U_2 不退出协议模拟，敌手 $\mathcal{A}$ 可在该协议模拟结束后输出公钥 $Q=Q_1+Q_2=Q_1+(Q-Q_1)$，与在协议现实执行环境中的输出相同；现实执行环境中的零知识证明与验证也同分布于 $(\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{com-zk}},\mathcal{F}^{\mathcal{R}_{\text{DL}}}_{\text{zk}})$-混合模型。因此，在分布式密钥生成协议模拟过程中，$\mathcal{A}$ 对模拟执行环境的视图和对现实执行环境中的视图是不可区分的。

在两方协同签名模拟过程中，可以观察到协议现实执行环境与理想模拟执行环境主要有 2 个区别。一是计算 R_2 的过程，同理于上述分析，对于 $\mathcal{A}$ 来说，模拟执行过程中的 $R_2=R-k_1P$ 和现实执行过程中的 $R_2=k_2P$ 是同分布且不可区分的。二是 $(u_2,v_2,w_2,t_2,\alpha_2,\beta_2)$ 的构造，在协议模拟环境中，u_2,v_2,w_2,t_2,α_2 均是 $\mathcal{C}$ 选取的随机数，而 β_2 是通过计算式 $\beta_2=s(\alpha_1+\alpha_2)-\beta_1$ 获得的；在现实执行环境中，$(u_2,v_2,w_2,t_2,\alpha_2,\beta_2)$ 是通过计算 $u_2=k_2-\Delta a_{21}$、$v_2=\rho_2-\Delta b_{21}$、$w_2=\delta_2-\Delta a_{22}$、$t_2=\rho_2-\Delta b_{22}$、$\alpha_2=k_2v+\rho_2u+\Delta c_{21}$、$\beta_2=\delta_2t+\rho_2w+\Delta c_{22}$ 获得的。由于 ρ_2、Δa_{21}、Δb_{21}、Δa_{22}、Δb_{22} 均为独立分布的随机数，故 $\mathcal{C}$ 随机选取的 $(u_2,v_2,w_2,t_2,\alpha_2)$ 分别与现实执行环境中诚实参与方 U_2 计算的 $(k_2-\Delta a_{21}$，$\rho_2-\Delta b_{21}$，$\delta_2-\Delta a_{22}$，$\rho_2-\Delta b_{22}$，$k_2v+\rho_2u+\Delta c_{21})$ 同分布；又因为 α_2 和 ρ_2 是随机的，所以 $s(\alpha_1+\alpha_2)-\beta_1$ 与 β_2 也是同分布的。此外，在模拟协议不终止的情况下，$\mathcal{A}$ 可在游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中输出当前模拟实例的签名 $r=r_x \bmod q$ 和 $s=(\alpha_1+\alpha_2)^{-1}\cdot$

$[\beta_1+(s(\alpha_1+\alpha_2)-\beta_1)]$，这与协议现实执行环境中的输出相同。因此，在$U_1$被腐化时的两方协同签名协议模拟的过程中，$\mathcal{A}$对模拟执行环境的视图和对现实执行环境的视图是不可区分的。

签名伪造。通过上述分析可知，在分布式密钥生成模拟和两方协同签名模拟过程中，$\mathcal{A}$在腐化参与方U_1时无法区分协议现实执行环境和协议模拟执行环境，故$\mathcal{C}$可以调用$\mathcal{A}$的伪造签名攻击原始ECDSA方案。

由于在模拟过程中，$\mathcal{A}$生成的签名验证公钥Q是通过游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$中预言机$\mathcal{F}_{\text{ECDSA}}$获得的，故$\mathcal{A}$在游戏$\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$中的成功伪造$(m^*,\sigma^*)$也可作为$\mathcal{C}$在游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$中的成功伪造，即$\mathcal{A}$在游戏$\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$输出一个伪造的消息签名对$(m^*,\sigma^*=(r^*,s^*))$，$\mathcal{C}$可随之在游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$中输出$(m^*,\sigma^*=(r^*,s^*))$。

由此可见，若敌手$\mathcal{A}$能够以不可忽略的概率ϵ赢得游戏$\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$，那么$\mathcal{C}$同样能够以概率$\epsilon$赢得游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$。原始ECDSA签名方案是可证明安全的[1]，即$\mathcal{C}$赢得游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$的概率是可以忽略不计的，故$\mathcal{A}$赢得游戏$\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$的概率也应是可忽略不计的（即满足定义1），即假设概率ϵ不可忽略是不成立的。因此，本文所提两方协同ECDSA签名方案在腐化U_1时是安全的。

当$i=2$时，PPT敌手$\mathcal{A}$腐化参与方U_2。类似地，本文构造一个针对游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$的PPT敌手$\mathcal{C}$，同时作为游戏$\text{GameDistSign}^2_{\mathcal{A},\Pi}(1^\lambda)$的模拟器，能够利用$\mathcal{A}$的伪造签名攻击原始ECDSA签名方案。

1) 分布式密钥生成协议模拟

$\mathcal{C}$模拟游戏$\text{GameDistSign}^2_{\mathcal{A},\Pi}(1^\lambda)$中的预言机$\Pi_2(\cdot)$，当前状态查询为$(0,\cdot)$，并且仅在$\mathcal{A}$向预言机$\Pi_2(\cdot)$发送了分布式密钥生成协议模拟开始指令后，$\mathcal{C}$才能工作。此外，在该过程中不会对其他状态的查询$(\text{sid},\cdot)$进行回答，即分布式密钥生成协议模拟结束之前，$\mathcal{C}$不会对两方协同签名协议模拟有关的任何询问进行回答。假设$\mathcal{A}$已发送指令$(0,0)$，则该过程的具体步骤如下。

① 在游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$中，$\mathcal{C}$向预言机$\mathcal{F}_{\text{ECDSA}}$进行密钥生成查询，获得$(1^\lambda,Q)$。其中，$\lambda$为安全参数，$Q$为ECDSA的签名验证公钥。

② 在游戏$\text{GameDistSign}^2_{\mathcal{A},\Pi}(1^\lambda)$中，$\mathcal{C}$先计算$(\text{proof-receipt},1)$作为预言机的应答发送给$\mathcal{A}$。

③ 当收到$\mathcal{A}$发送给$\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\text{DL}}}$的$(0,(\text{prove},2,Q_2,d_2))$后，$\mathcal{C}$验证$Q_2=d_2P$是否成立，如果不成立，则终止协议；否则计算$Q_1=Q-Q_2$，然后模拟$\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\text{DL}}}$将消息$(\text{decom-proof},1,Q_1)$作为预言机的应答发送给$\mathcal{A}$。若$U_1$不退出该协议模拟，$\mathcal{C}$存储$(d_2,Q,\text{SET}^2_{\text{tri}})$，然后结束分布式密钥生成协议。

2) 两方协同签名协议模拟

$\mathcal{C}$模拟游戏$\text{GameDistSign}^2_{\mathcal{A},\Pi}(1^\lambda)$中的预言机$\Pi_2(\cdot)$，当前状态查询设为$(\text{sid},\cdot)$，sid是一个新的会话标识符。给定一个待签名消息$m$，$\mathcal{C}$在该过程的具体步骤如下。

① 在游戏$\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$中，$\mathcal{C}$向预言机$\mathcal{F}_{\text{ECDSA}}$进行签名查询，获得$\sigma=(r,s)$，并根据签名验证算法进一步计算得到$R$。

② 在游戏$\text{GameDistSign}^2_{\mathcal{A},\Pi}(1^\lambda)$中，$\mathcal{C}$先计算$(\text{proof-receipt},\text{sid}\|1)$作为预言机回复发送给$\mathcal{A}$。

③ 当收到$\mathcal{A}$发送给$\mathcal{F}_{\text{zk}}^{\mathcal{R}_{\text{DL}}}$的消息$(\text{sid},(\text{prove},2,R_2,k_2))$后，$\mathcal{C}$验证$R_2=k_2P$是否成立，如果不成立，则终止协议；否则先计算$R_1=R-R_2$，再选择4个随机数$(u_2,v_2,w_2,t_2)\in Z_q^*$，然后模拟$\mathcal{F}_{\text{com-zk}}^{\mathcal{R}_{\text{DL}}}$和$\mathcal{F}_{\text{BT}}$将消息$(\text{decom-proof},\ \text{sid}\|1,R_1)$和随机数$(u_2,v_2,w_2,t_2)$作为预言机应答发送给$\mathcal{A}$。

④ 当收到$\mathcal{A}$发送给$\mathcal{F}_{\text{BT}}$的消息$(\text{sid},(u_2,v_2,w_2,t_2,\alpha_2,\beta_2))$后，$\mathcal{C}$验证$\mathcal{A}$的消息是否按照4.1节中步骤4)计算所得，若不是，则终止协议，否则$\mathcal{C}$输出$\sigma=(r,s)$。

不可区分性分析。在U_2被腐化时的分布式密钥生成模拟过程中，$\mathcal{A}$对模拟执行环境的视图和对现实执行环境中的视图是不可区分的，其原理与U_1被腐化时相同，所以本文不再重复描述。

对于U_2被腐化时的两方协同签名模拟过程，可以观察到协议现实执行环境与理想模拟执行环境同样有2个区别。一是计算R_2的过程，同理于上述分析，对于$\mathcal{A}$来说，模拟执行过程中的$R_1=R-k_2P$和现实执行过程中的$R_1=k_1P$是同分布且不可区分的。二是(u_1,v_1,w_1,t_1)的构造，在协议理想模拟环境中，u_1,v_1,w_1,t_1均为$\mathcal{C}$选取的随机数；在现实执行环

境中，(u_1,v_1,w_1,t_1) 则通过计算 $u_1=k_1-\Delta a_{11}$、$v_1=\rho_1-\Delta b_{11}$、$w_1=\delta_1-\Delta a_{12}$、$t_1=\rho_1-\Delta b_{12}$ 获得。由于 ρ_1、Δa_{11}、Δb_{11} 和 Δa_{12}、Δb_{12} 均为独立分布的随机数，故 $\mathcal{C}$ 随机选取的 (u_1,v_1,w_1,t_1) 分别与现实执行环境中诚实参与方 U_2 计算的 $(k_1-\Delta a_{11}$，$\rho_1-\Delta b_{11}$，$\delta_1-\Delta a_{12}$，$\rho_1-\Delta b_{12})$ 同分布。此外，在模拟协议不终止的情况下，$\mathcal{C}$ 可在游戏中输出签名 $\sigma=(r,s)$。由于 $\sigma=(r,s)$ 是向预言机 $\mathcal{F}_{\text{ECDSA}}$ 进行查询而获得的，故与协议现实执行环境中的输出相同。因此，在 U_2 被腐化的两方协同签名协议模拟过程中，$\mathcal{A}$ 对模拟执行环境的视图和对现实执行环境中的视图是不可区分的。

签名伪造。类似地，$\mathcal{A}$ 在腐化参与方 U_2 时无法区分协议现实执行环境和协议理想执行环境，$\mathcal{C}$ 可以调用 $\mathcal{A}$ 的伪造签名攻击原始 ECDSA 方案。由于模拟过程中 $\mathcal{A}$ 生成的签名验证公钥 Q 是通过游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 中预言机 $\mathcal{F}_{\text{ECDSA}}$ 获得的，故 $\mathcal{A}$ 在游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 中的成功伪造 (m^*,σ^*) 也为 $\mathcal{C}$ 在游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$ 中的成功伪造。因此，若敌手 $\mathcal{A}$ 能够以概率 ϵ 赢得游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$，那么 $\mathcal{C}$ 同样能够以概率 ϵ 赢得游戏 $\text{GameSign}_{\mathcal{C},\pi}(1^\lambda)$。而原始 ECDSA 签名方案被证明是安全的（满足定义 1）[1]，即概率 ϵ 被认为是可忽略不计的，从而推导出 $\mathcal{A}$ 赢得游戏 $\text{GameDistSign}^1_{\mathcal{A},\Pi}(1^\lambda)$ 的概率可忽略不计（即满足定义 2）。因此，本文提出两方协同 ECDSA 签名方案在腐化 U_2 时是安全的。

证毕。

综上所述，本文所提两方协同 ECDSA 签名方案是可证明安全的。换句话说，在仅一个签名参与方私钥泄露的情况下，攻击者无法完成签名，这保证了另一签名参与方的私钥隐私性，即满足设计目标的隐私性要求。

6 性能分析

本节将从理论分析与实验仿真 2 个方面对本文所提方案（以下简称本文方案）进行性能评估，并与 Lindell 方案[13]、Doerner 方案[14]进行比较。由于本文方案和 Lindell 方案[13]均为两方协同 ECDSA 签名方案，故在与 Doerner 方案[14]比较时，仅考虑门限为(2,2)时的性能。

6.1 理论分析

首先，通过统计协议执行过程中使用的密码运算操作个数和网络传输的元素个数，对 Lindell 方案[13]、Doerner 方案[14]和本文方案的计算开销和通信开销进行了理论分析与比较。由于分布式密钥生成协议只需在系统中执行一次，因此本文主要分析与比较两方协同签名协议的计算开销和通信开销。

与 Paillier 加密、Paillier 解密、Paillier 同态乘、椭圆曲线点乘操作的时间相比，椭圆曲线点加操作的时间可忽略不计，且在方案中使用次数较少，因此对该操作不予统计。令 Enc、Dec、HM 分别表示 Paillier 算法的加密、解密、同态乘法操作，PM 和 h 分别表示椭圆曲线点乘操作和一般哈希操作，N 和 λ 分别表示 Paillier 公钥长度和 Z_q^* 域上元素长度。此外，为了不失一般性，令输出签名的一方为 U_1，另一方为 U_2。

由于不同应用环境对方案的安全需求不同，因此，本文分别考虑了方案在半诚实模型下和恶意模型下的计算开销和通信开销。半诚实模型下，敌手企图获取额外的秘密信息，但在执行过程中不会偏移协议的设置。从方案设计角度，在半诚实模型下，本文方案与 Lindell 方案[13]中的参与方不需要执行关于离散对数的零知识证明协议，Doerner 方案[14]中的参与方不需要执行一致性检测协议。

表 1 和表 2 分别为恶意模型下和半诚实模型下的计算开销与通信开销统计结果。其中，Doerner 方案[14]中哈希操作 h 的总个数是在安全参数为 256 的情况下进行统计计算的。安全级别越高，Doerner 方案[14]中哈希操作的总数越高，这对计算开销有一定影响；同时 Lindell 方案[13]中的 Paillier 加密操作和解密操作的计算效率也会降低。

从表 1 可知，在计算开销方面，与 Lindell 方案[13]比较，本文方案在 U_1 端少一个椭圆曲线点乘操作和一个 Paillier 解密操作，在 U_2 端少一个 Paillier 加密操作、一个同态乘法操作和一个椭圆曲线点乘操作；与 Doerner 方案[14]比较，本文方案在 U_1 端和 U_2 端分别少一个和 2 个点乘操作，以及数千个哈希操作。因此，在恶意敌手模型下，本文方案的计算开销最低。

在通信开销方面，显然 Doerner 方案[14]是最高的；按照 NIST 的密钥长度建议[32]，对于同一安全级别的协议，基于大整数分解的协议密钥长度约为椭圆曲线密钥长度的 8 倍以上，即 $N>8\lambda$，从而可得本文方案的通信开销比 Lindell 方案[13]低。因此，本文方案的通信开销也是最低的。

从表 2 可知，去掉零知识证明协议或一致性检

表 1　恶意模型下两方签名协议计算开销与通信开销理论比较

方案	U_1 计算开销	U_2 计算开销	U_1+U_2 通信开销
Lindell 方案	Dec + 7PM + 4h	Enc + HM + 5PM + 3h	9λ+2N
Doerner 方案	7PM + 5 524h	6PM + 5 476h	λ(8λ+794)+2
本文方案	6PM + 5h	4PM + 3h	19λ

表 2　半诚实模型下两方签名协议计算开销与通信开销理论比较

方案	U_1 计算开销	U_2 计算开销	U_1+U_2 通信开销
Lindell 方案	Dec + 4PM + h	Enc + HM +2PM	4λ+2N
Doerner 方案	4PM + 5 012h	2PM + 4884h	λ(6λ+794)+2
本文方案	3PM + 2h	PM + h	14λ

测协议后，即在半诚实模型下，本文方案由于没有开销较高的 Paillier 加解密操作和大量的哈希操作，因此在用户 U_1 端和 U_2 端的计算开销和通信开销仍然比其他 2 种方案低。

6.2　实验仿真

在仿真测试中，本文以 NIST 标准化的 secp256k1 椭圆曲线为基准，即安全参数 λ=256，杂凑函数则采用 SHA-256，Paillier 算法的密钥长度 N=2 048。测试程序基于开源密码学库 RELIC[33]，并使用 C 语言编写。测试环境为一台配置为 64 位 Windows10 专业版操作系统、Intel(R) Core(TM) i5-6300U CPU @ 2.40 GHz 处理器、16.0 GB 内存的惠普笔记本电脑。

在不考虑网络时延的情况下，本文在局域网内分别对 3 种方案的两方协同签名协议运行 10 000 次，以平均运行时间和通信带宽为实验数据。图 1 和图 2 分别为各方案在恶意模型和半诚实模型下的运行时间。

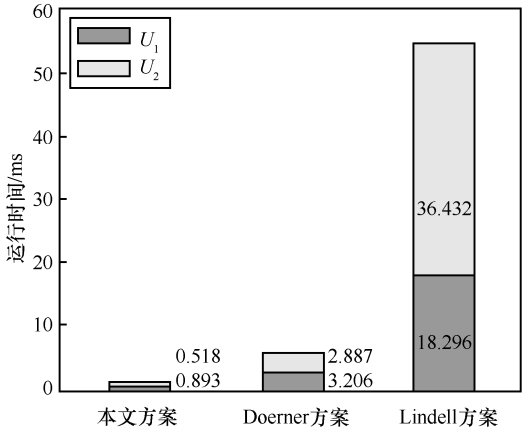


图 1　恶意模型下两方协同签名方案的运行时间比较

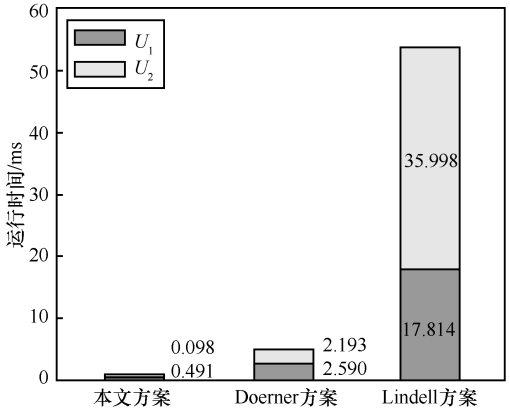


图 2　半诚实模型下两方协同签名方案运行时间比较

从图 1 和图 2 可知，与 Lindell 方案[13]和 Doerner 方案[14]相比，本文方案在两方协同签名计算开销上具有显著优势。其中，在恶意模型下，本文方案的签名运行效率比 Lindell 方案快 97.42%，比 Lindell 方案快 76.84%。值得注意的是，由于 Doerner 方案签名过程需要参与方进行多次交互，因此，若充分考虑网络时延，Doerner 方案的实际运行效率将比当前测试结果更低。

表 3 为恶意模型和半诚实模型下的通信开销统计结果。由表 3 可知，与 Lindell 方案和 Doerner 方案相比，本文方案的通信开销最小。其中，在恶意模型下，本文方案的通信开销比 Lindell 方案小 21.74%，比 Doerner 方案小 99.3%。

表 3　恶意模型和半诚实模型下的通信开销比较

方案	恶意模型通信开销/B	半诚实模型通信开销/B
Lindell 方案	768	608
Doerner 方案	85 713	72 812
本文方案	601	446

综上所述，本文所提两方协同 ECDSA 方案在计算开销和通信开销上均具有明显的优势，满足设计目标中的高效性要求。

7　结束语

由于签名私钥丢失和签名权利集中都可能损害区块链系统合法用户的权益，分布式安全计算 ECDSA 签名的研究应运而生。然而现有的两方或多方协同 ECDSA 签名算法存在计算开销或通信开销过高的问题，导致难以广泛应用于资源受限的用户端。为了降低签名私钥泄露风险并同时保证协同签名效率，本文提出一种安全高效的 ECDSA 两方协同签名方案，主要通过引入 Beaver 三元组预计算技术，避免在协同签名过程中使用计算繁重的同态加密和通信开销较大的不经意传输等操作，从而实现对签名私钥的隐私保护并极大地提高协同签名的效率。方案的安全性在通用可组合框架下被证明，仿真实验结果表明，本文方案在签名计算开销和通信开销方面优于现有的两方协同 ECDSA 签名算法。因此，本文的算法是可行且高效的。

参考文献：

[1] JOHNSON D, MENEZES A, VANSTONE S. The elliptic curve digital signature algorithm (ECDSA)[J]. International Journal of Information Security, 2001, 1(1): 36-63.

[2] AL-ZUBAIDIE M, ZHANG Z, ZHANG J. Efficient and secure ECDSA algorithm and its applications: a survey[J]. arXiv Preprint, arXiv: 1902. 10313, 2019.

[3] BLAKE-W S, BOLYARD N, GUPTA V, et al. Elliptic curve cryptography (ECC) cipher suites for transport layer security (TLS)[R]. RFC 4492, 2006.

[4] DALSKOV A, ORLANDI C, KELLER M, et al. Securing DNSSEC keys via threshold ECDSA from generic MPC[C]//European Symposium on Research in Computer Security. Berlin: Springer, 2020: 654-673.

[5] HENNING P J. A taxonomy of cryptocurrency enforcement actions[J]. Brooklyn Journal of Corporate, Financial and Commercial Law, 2020, 14(2): 227-257.

[6] LU H, JIN C, HELU X, et al. AutoD: intelligent blockchain application unpacking based on JNI layer deception call[J]. IEEE Network, 2020, PP(99): 1-7.

[7] JANPITAK N, LILAKIATSAKUN W, SATHITWIRIYAWONG C. The novel secure testament methodology for cryptocurrency wallet using mnemonic seed[J]. Information Security Journal: A Global Perspective, 2020, 29(4): 169-182.

[8] TOMESCU A, CHEN R, ZHENG Y, et al. Towards scalable threshold cryptosystems[C]//2020 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2020, doi.org/ 10.1109/SP40000.2020.00059.

[9] 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, 2019, 48(1): 1-8.
HOU H X, YANG B, ZHANG L N, et al. Secure two-party SM2 signature algorithm[J]. Acta Electronica Sinica, 2019, 48(1): 1-8.

[10] MACKENZIE P, REITER M K. Two-party generation of DSA signatures[J]. International Journal of Information Security, 2004, 2(3-4): 218-239.

[11] GENNARO R, GOLDFEDER S, NARAYANAN A. Threshold- optimal DSA/ECDSA signatures and an application to Bitcoin wallet security[C]//International Conference on Applied Cryptography and Network Security. Berlin: Springer, 2016: 156-174.

[12] BONEH D, GENNARO R, GOLDFEDER S. Using level-1 homomorphic encryption to improve threshold DSA signatures for bitcoin wallet security[C]//International Conference on Cryptology and Information Security in Latin America. Berlin: Springer, 2017: 352-377.

[13] LINDELL Y. Fast secure two-party ECDSA signing[C]//Annual International Cryptology Conference. Berlin: Springer, 2017: 613-644.

[14] DOERNER J, KONDI Y, LEE E, et al. Secure two-party threshold ECDSA from ECDSA assumptions[C]//2018 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2018: 980-997.

[15] CHOU T, ORLANDI C. The simplest protocol for oblivious transfer[C]//International Conference on Cryptology and Information Security in Latin America. Berlin: Springer, 2015: 40-58.

[16] KELLER M, ORSINI E, SCHOLL P. Actively secure OT extension with optimal overhead[C]//Annual Cryptology Conference. Berlin: Springer, 2015: 724-741.

[17] CASTAGNOS G, CATALANO D, LAGUILLAUMIE F, et al. Two-party ECDSA from hash proof systems and efficient instantiations[C]//Annual International Cryptology Conference. Berlin: Springer, 2019: 191-221.

[18] LINDELL Y, NOF A. Fast secure multiparty ECDSA with practical distributed key generation and applications to cryptocurrency custody[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2018: 1837-1854.

[19] DOERNER J, KONDI Y, LEE E, et al. Threshold ECDSA from ECDSA assumptions: the multiparty case[C]//2019 IEEE Symposium on Security and Privacy. Piscataway: IEEE Press, 2019: 1051-1066.

[20] GENNARO R, GOLDFEDER S. Fast multiparty threshold ECDSA with fast trustless setup[C]//Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM Press, 2018: 1179-1194.

[21] HE D, ZHANG Y, WANG D, et al. Secure and efficient two-party signing protocol for the identity-based signature scheme in the IEEE P1363 standard for public key cryptography[J]. IEEE Transactions on Dependable and Secure Computing, 2018, 17(5): 1124-1132.

[22] FENG Q, HE D, LIU Z, et al. Distributed signing protocol for IEEE P1363-compliant identity-based signature scheme[J]. IET Information Security, 2020, 14(4): 443-451.

[23] ZHANG Y, HE D, ZHANG M, et al. A provable-secure and practical

two-party distributed signing protocol for SM2 signature algorithm[J]. Frontiers of Computer Science, 2020, 14(3): 1-14.

[24] MU Y H, XU H X, LI P L, et al. Secure two-party SM9 signing[J]. SCIENCE CHINA Information Sciences, 2020, 63(8): 189101.

[25] BEAVER D. Efficient multiparty protocols using circuit randomization[C]//Annual International Cryptology Conference. Berlin: Springer, 1991: 420-432.

[26] FENG Q, HE D, LIU Z, et al. SecureNLP: a system for multi-party Privacy-preserving natural language processing[J]. IEEE Transactions on Information Forensics and Security, 2020, PP(99): 1.

[27] HUANG K, LIU X, FU S, et al. A lightweight privacy-preserving CNN feature extraction framework for mobile sensing[J]. IEEE Transactions on Dependable and Secure Computing, 2019, doi.org/10.1109/TDSC.2019.2913362.

[28] CANETTI R. Universally composable security: a new paradigm for cryptographic protocols[C]//Proceedings 42nd IEEE Symposium on Foundations of Computer Science. Piscataway: IEEE Press, 2001: 136-145.

[29] SCHNORR C P. Efficient identification and signatures for smart cards[C]//Conference on the Theory and Application of Cryptology. Berlin: Springer, 1989: 239-252.

[30] WU Y, WANG X, SUSILO W, et al. Efficient server-aided secure two-party computation in heterogeneous mobile cloud computing[J]. IEEE Transactions on Dependable and Secure Computing, 2020: doi.org/10.1109/TDSC.2020.2966632.

[31] LINDELL Y. How to simulate it–a tutorial on the simulation proof technique[M]. Berlin: Springer, 2017: 277-346.

[32] BARKE E. Recommendation for key management-part 1 (revised)[J]. Special Publication 800-57, 2020: doi.org/10.6028/NIST.SP.800-57ptlr5.

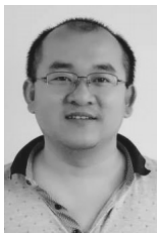
[33] KANENARI T, TAKAHASHI Y, HASHIMOTO Y, et al. A comparison of relic-toolkit and ELiPS libraries for a pairing-based homomorphic encryption[C]//2019 34th International Technical Conference on Circuits/Systems, Computers and Communications. Piscataway: IEEE Press, 2019: 1-4.

[作者简介]

王婧（1994− ），女，安徽安庆人，武汉大学博士生，主要研究方向为云存储安全、数字签名、安全多方计算等。

吴黎兵（1972− ），男，湖北武汉人，博士，武汉大学教授、博士生导师，主要研究方向为分布式计算、可信软件、无线传感网络等。

罗敏（1974− ），男，湖北武汉人，博士，武汉大学副教授、硕士生导师，主要研究方向为密码协议、信息安全、区块链技术与应用等。



何德彪（1980− ），男，湖北武汉人，博士，武汉大学教授、博士生导师，主要研究方向为密码协议、信息安全、区块链技术与应用等。