Robust Threshold ECDSA with Online-Friendly Design in Three Rounds

Guofeng Tang, Haiyang Xue

IEEE S&P 2025

content

目录

01 引言与背景

阈值签名概念

01

定义与原理

一个(t,n)阈值签名将私钥分散至n个参与者，确保任何t个诚实参与者可生成有效签名，而少于t个参与者无法恢复私钥。

02

增强安全性

通过分散私钥，阈值签名显著降低单点故障风险，提高系统整体的安全性和可靠性。

03

应用场景

广泛应用于区块链交易确认、分布式密钥管理、匿名凭证系统等，保障关键操作的安全执行。

04

核心优势

即使部分参与者失联或被攻陷，只要诚实参与者数量达到阈值，系统仍能正常运作，确保服务连续性和数据安全。

阙值ECDSA安全性

基本安全要求

能够容忍最多t-1个恶意（或损坏）参与者。

作弊者识别

可用于检测行为不端方。通常情况下，可以设计一个协议来消除作弊者，并重新启动签名过程，直到生成签名为止。

鲁棒性

在识别出作弊者后，只要重新参与的（半）诚实方仍然满足阈值t，就允许签名过程继续并生成有效签名。

02 技术挑战与贡献

核心问题

Schemes	Types	Rounds	CI	R	Online Cost
DKLs24 [36]	OT	3	×	×	Optimal
CDKS24 [31]	OT	7	✓	×	2×DDH-ZKP
CGG+20 [20]	LHE	4	✓	×	Enc, Paillier-ZKP
CCL+23 [25]	LHE	7	✓	×	2×DDH-ZKP
WMY+23 [58]	LHE	5+2⋆	✓	✓†	3×DDH-ZKP
WMC24 [57]	TLHE	4	✓	✓	Dec, CL-ZKP
Ours	LHE	3	✓	✓	DDH-ZKP

⋆ "+2" indicates the extra cost for CI.
† We show its vulnerability when combining CI and R properties.

表1：阈值ECDSAs在轮数、作弊者识别、鲁棒性和在线成本方面的比较。"最优"表示在线阶段除签名验证外没有更多的EC组操作。"Enc"、"Dec"、"Paillier-ZKP"和"CL-ZKP"表示加密、解密、Paillier或CL相关的ZKP。"DDH-ZKP"表示DDH关系的ZKP。

鲁棒性

确保即使存在恶意参与者，只要诚实参与者数量达到阈值，签名过程仍能顺利完成。

作弊者识别

及时检测出作弊行为的存在。一旦存在，可以确定地识别出作弊者并将其从系统中清除，使方案更具可操作性。

在线友好性需求

在线阶段应仅依赖简单的椭圆曲线操作，以降低计算开销，提高效率。

主要贡献

首次实现

提出首个三轮、在线友好的鲁棒阈值ECDSA方案，结合了效率与安全性。目前为阈值ECDSA增加鲁棒性的技术至少需要在三轮签名阶段额外增加一轮。协议成功地实现了CI和鲁棒性，无需额外的回合。此外，协议的在线阶段只依赖于少量的椭圆曲线操作。

方案对比

与WMY+23和WMC24进行了比较。在线阶段比WMY+23快大约2.5倍。此外，在线计算速度明显快于WMC24（0.5毫秒对308毫秒）。不过，总体性能只有在较小规模的情况下才优于WMC24：具体来说，当n≤12时，总体运行时间更快，而当n≤5时，总带宽更低。

公共校验

通过对WMY+23的攻击，说明公共校验对于在稳健的阈值签名方案中实现安全的CI至关重要。WMY+23的漏洞是由于使用了MtA和私人检查。私人校验需要秘密输入，而公开校验不依赖任何秘密，任何人都可以公开验证传输的信息。私人校验需要额外的步骤，可能会导致秘密泄露，从而达到公开验证的效果。

扩展至BBS+

将技术扩展到鲁棒阈值BBS+签名，它面临着与鲁棒阈值ECDSA类似的挑战。阈值BBS+签名是一种关键的基本签名，稳健阈值BBS+也在三轮中实现了稳健性。由于在线阶段只需要对EC组进行操作，因此它比WMC24快得多，达到了2.48毫秒，而在双方设置中为287毫秒。同样，只有当n<10时，总体开销才会优于WMC24。

核心方法与创新

03

关键技术

乘法到加法(MtA)协议：MtA接收来自Alice和Bob双方的输入a和b，并为Alice和Bob安全地计算出$\alpha+\beta=ab(\mathrm{mod}q)$。

- 爱丽丝向鲍勃发送a的密文C_a及其完备性证明;
- Bob随机选取一个β，用Alice的公钥生成$-\beta$的密文$C_{-\beta}$，并回复密文$C_\alpha=b\odot C_a\oplus C_{-\beta}$;
- Alice解密C_α得到$\alpha=ba-\beta\mathrm{mod}q$。

假定Bob的输入b是公共点$X_{Bob}=bG$的离散对数，一致性检查机制分为以下两种方法。

私人检查

一方（Alice）可以使用其秘密共享验证另一方（Bob）的正确性。这就要求Alice通过公开自己的秘密共享来识别作弊者（Alice通过α,a检查Bob发来的C_α和$B_{Bob}=\beta G$）。如果只需要CI，这种方法是可以接受的，但如果还需要鲁棒性，这种方法就存在特定的漏洞——控制t−1方的对手最终可以根据Alice透露的共享信息和生成的签名（在鲁棒性要求下）重建秘密签名密钥。

公开检查

利用可公开验证的零知识证明（ZKPs），在MtA中使用公开检查来识别作弊者。这些零知识证明允许任何一方识别出作弊者。但是，由于使用了Paillier加密和相关的ZKPs，这种方法会带来巨大的计算开销，尤其是在在线阶段。

关键技术

CL方案

可以避免Paillier和JL加密所要求的范围证明。

- $\mathsf{KGen}(1^\lambda) \to (ek, dk)$: output $dk \leftarrow\!\!\$\ \mathcal{D}$ and $ek = \mathfrak{g}_q^{dk}$.
- $\mathsf{Enc}(ek, m; \rho) \to C_m$: pick $\rho \leftarrow\!\!\$\ \mathcal{D}_q$ and output $C_m = (\mathfrak{g}_q^{\rho}, \mathfrak{f}^m ek^{\rho})$.
- $\mathsf{Dec}(dk, (c_1, c_2)) \to m$: output $\mathsf{Dlog}(c_2/c_1^{dk})$.
- Addition: $C_m \oplus C_{m'} \to C_{m+m'}$: parse $C_m = (c_1, c_2)$ and $C_{m'} = (c_1', c_2')$, output $C_{m+m'} = (c_1 \cdot c_1', c_2 \cdot c_2')$.
- Constant Multiplication: $a \odot C_m \to C_{am}$: parse $C_m = (c_1, c_2)$, output $C_{am} = (c_1^a, c_2^a)$.

零知识证明

零知识证明（ZKP）允许证明者在不透露w的情况下说服验证者相信(x,w)∈R。

CL Key Pair. $\mathcal{R}_{\mathsf{key}} = \{(ek, dk) : dk \in \mathcal{D}, ek = \mathfrak{g}_q^{dk}\}$.

Well-formedness of a CL Ciphertext. Given ek as CL's encryption key, define

$$\mathcal{R}_{\mathsf{Enc}} = \{(C_m, (m \in \mathbb{Z}_q, \rho \in \mathcal{D}_q)) : C_m = \mathsf{Enc}(ek, m; \rho)\}.$$

Validity of Encrypted Shares. Given $\{ek_j\}_{j \in S}$ as a set of CL's encryption keys and threshold t, define[4]

$$\mathcal{R}_{\mathsf{Sh}} = \{(\{C_{f(j)}\}_{j \in S}, (f(\cdot), \rho)) : \deg(f(\cdot)) \le t-1, C_{f(j)} = \mathsf{Enc}(ek_j, f(j); \rho)\}.$$

CL Decryption and Discrete Logarithm.

$$\mathcal{R}_{\mathsf{Dec\text{-}DL}} = \{((C_a, A), (a \in \mathbb{Z}_q, dk \in \mathcal{D})) : A = aG, a = \mathsf{Dec}(dk, C_a)\}.$$

DDH Relation.

$$\mathcal{R}_{\mathsf{DDH}} = \{((G, A, B, C) \in \mathbb{G}^4, a \in \mathbb{Z}_q) : A = aG, C = aB\}.$$

分布式随机性生成

每个参与者都能为一个秘密值x←Z(q)获得一个t门限份额x_i，同时相应的EC点X=xG和每个公开份额$X_i=x_iG$都会被公开。

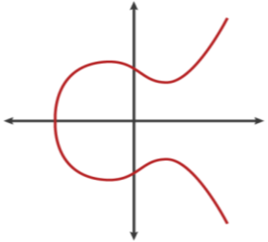
- Round 1: Each party $\mathcal{P}_i$ broadcasts the *shares distribution* message $(\{C_{x_{j,i}}\}_{j \in S}, \pi_i) \leftarrow \mathsf{ShareDist}(\{ek_j\}_{j \in S}, t)$:
  a) pick a $(t-1)$-degree polynomial $f(\cdot)$ and $\rho \leftarrow\!\!\$\ \mathcal{D}_q$
  b) compute $C_{x_{j,i}} = \mathsf{Enc}(ek_j, f(j); \rho)$ for each $j \in S$
  c) generate $\pi_i \leftarrow \mathsf{Prove}_{\mathsf{Sh}}(\{C_{x_{j,i}}\}_{j \in S}, (f(\cdot), \rho))^a$.
- Round 2: After receiving $(\{C_{x_{\nu,j}}\}_{\nu \in S}, \pi_j)$ from each $\mathcal{P}_j$, $\mathcal{P}_i$ first verifies π_j and sets $S = S \setminus \{j\}$ if it fails. $\mathcal{P}_i$ generates the *shares combination* message $(x_i, X_i, \pi_i') \leftarrow \mathsf{ShareComb}(\{C_{x_{i,j}}\}_{j \in S}, dk_i)$:
  a) decrypt $x_{i,j} \leftarrow \mathsf{Dec}(dk_i, C_{x_{i,j}})$ for each $j \in S$
  b) compute $x_i = \sum_{j \in S} x_{i,j} \mod q$ and $X_i = x_i G$.
  c) generate $\pi_i' \leftarrow \mathsf{Prove}_{\mathsf{Dec\text{-}DL}}((C_{x_i}, X_i), (x_i, dk_i))$ with $C_{x_i} = \oplus_{j \in S} C_{x_{i,j}}$.

  Broadcast (X_i, π_i').
- Output: On receiving (X_j, π_j') from each $\mathcal{P}_j$, verify π_j' and set $S = S \setminus \{j\}$ if it fails. $\mathcal{P}_i$ stores its own secret share x_i and others' public shares $\{X_j\}_{j \in S}$, as well as $X = \sum_{j \in S} \lambda_{j,S} X_j$.

a. The $\{C_{x_{j,i}}\}_{j \in S}$'s validity proof π_i is constant-size, proving that their plaintexts come from a same polynomial of degree $\le t-1$ [22].

ECDSA介绍

ECDSA Sign(x,m):



$$k \leftarrow Z_q$$

$$R = k \cdot G$$

$$m = H(msg)$$

$$s = \frac{m}{k} + \frac{x \cdot r_x}{k} = \frac{m + x \cdot r}{k} \cdot \frac{[\gamma]}{[\gamma]}$$

$$\sigma = (s, R)$$

output σ

DKLs:

- Round 1 and 2 (offline phase): Each party randomly selects k_i and γ_i. Using MtA, the parties compute additive shares δ_i of γk, such that $\gamma k = \delta_1 + \cdots + \delta_{n'}$, and additive shares ζ_i of γx, satisfying $\gamma x = \zeta_1 + \cdots + \zeta_{n'}$. Simultaneously, they execute a standard commit-and-release mechanism to assemble the pre-signature $R = \sum_i k_i G$. All tasks are completed within two rounds.
- Round 3 (online phase): Given a message $m \in \mathbb{Z}_q$, broadcast δ_i and $\chi_i = m\gamma_i + r\zeta_i$ where $r = R|_{x\text{-axis}} \in \mathbb{Z}_q$.

虽然DKLs24在最后一轮之前提供了有效的MAC式统计检查，但如果Pi在最后一轮中使用了不正确的份额γi来评估χi，诚实的各方只有在验证了最终签名之后才能发现作弊行为且无法识别作弊方。

关键技术

未确定临时随机数

采用不确定的临时秘密随机数γ，无需额外进行一轮交互来生成确定的份额，简化在线阶段计算，增强鲁棒性。尽管存在不确定性，但签名者仍能成功组合签名，因为该值最终会被抵消
。

线上阶段

使用简单的零知识证明来证明DDH关系，这只需要廉价的elliptic曲线操作。DDH关系的证明可启用CI机制，允许各方公开识别和排除行为不当的各方
。

$$s = \frac{m}{k} + \frac{x \cdot r_x}{k} = \frac{m + x \cdot r}{k} \cdot \frac{[\gamma]}{[\gamma]}$$

- 第一轮：各方使用γ的加法份额作为输入广播初始MtA信息，同时生成k的t个阈值份额。
- 第二轮：各方使用k和x的t个阈值份额执行第二轮MtA。
- 第三轮：在联机阶段对γ进行简单的公共检查。签名方程中组装的分母kγ和分子mγ+rxγ使用相同的γ值，γ在最后组装阶段被抵消。

公开检查γ的一致性问题：在以前的方法中，γG或γ的密文等特定公共值被用来验证γ的一致性。然而本方法由于γ的性质未定，没有与之相关的预定公共值。因此选择指导各方利用EC点γ_iR和γ_i(mG+rX)建立简单的DDH关系，分别验证每一方Pi的kγ份额和(mγ+rxγ)份额的正确性。

鲁棒阈值ECDSA方案

04

鲁棒阈值ECDSA方案

分布式密钥生成

- Round 1: $\mathcal{P}_i$ runs CL's key generation $\mathsf{KGen}(1^\lambda) \to (ek_i, dk_i)$ and generates the proof $\bar\pi_i \leftarrow \mathsf{Prove}_{\mathsf{key}}(ek_i, dk_i)$. Broadcast $ek_i, \bar\pi_i$.
- Round 2: Upon receiving $\bar\pi_j$ from each $\mathcal{P}_j$, $\mathcal{P}_i$ sets $\hat S = \hat S \setminus \{j\}$ if the verification fails. $\mathcal{P}_i$ broadcasts the shares-distribution message $(\{C_{x_{j,i}}\}_{j\in\hat S}, \pi_i) \leftarrow \mathsf{ShareDist}(\{ek_j\}_{j\in\hat S}, t)$.
- Round 3: On receiving π_j from each $j \in \hat S$, set $\hat S = \hat S \setminus \{j\}$ if the verification fails. $\mathcal{P}_i$ does:
  a) invoke $\mathsf{ShareComb}(\{C_{x_{i,j}}\}_{j\in\hat S}, dk_i) \to (x_i, X_i, \pi_i')$
  b) broadcast (X_i, π_i').
- Key Generation Output: Upon receiving $\{X_j, \pi_j'\}_{j\in\hat S}$, $\mathcal{P}_i$ sets $\hat S = \hat S \setminus \{j\}$ if π_j' is not valid. Each party gets the output:
  a) secret key share x_i, others' public key shares $\{X_j\}_{j\in\hat S}$
  b) ECDSA public key $X = \sum_{j\in\hat S} \lambda_{j,\hat S} X_j$.

线下阶段

- Round 1: $\mathcal{P}_i$ does the following
  a) $(C_{\gamma_i}, \tau_i) \leftarrow \mathsf{MPMtA}_1(\gamma_i, ek_i)$ with $\gamma_i \leftarrow\!\!\$\ \mathbb{Z}_q$
  b) generate $(\{C_{k_{j,i}}\}_{j\in S}, \pi_i) \leftarrow \mathsf{ShareDist}(\{ek_j\}_{j\in S}, t)$
  c) broadcast $(C_{\gamma_i}, \tau_i, \{C_{k_{j,i}}\}_{j\in S}, \pi_i)$.
- Round 2: On receiving τ_j, π_j from each $\mathcal{P}_j$, verify them and set $S = S \setminus \{j\}$ if one of them fails. $\mathcal{P}_i$ does
  a) $(k_i, R_i, \pi_i') \leftarrow \mathsf{ShareComb}(\{C_{k_{i,j}}\}_{j\in S}, dk_i)$
  b) $(\{C_{\alpha_{j,i}}, \beta_{i,j}\}_{j\in S}, \tau_i') \leftarrow \mathsf{MPMtA}_2(\{C_{\gamma_j}\}_{j\in S}, k_i)$
  c) $(\{C_{\hat\alpha_{j,i}}, \hat\beta_{i,j}\}_{j\in S}, \hat\tau_i') \leftarrow \mathsf{MPMtA}_2(\{C_{\gamma_j}\}_{j\in S}, x_i)$
  d) $B_{i,j} = \beta_{i,j}G, \hat B_{i,j} = \hat\beta_{i,j}G$ for each $j \in S$
  e) broadcast $(\{C_{\alpha_{j,i}}, C_{\hat\alpha_{j,i}}, B_{i,j}, \hat B_{i,j}\}_{j\in S}, R_i, \pi_i', \tau_i', \hat\tau_i')$.
- Offline Output: Upon receiving proofs $\pi_j', \tau_j', \hat\tau_j'$ from each $\mathcal{P}_j$, set $S = S \setminus \{j\}$ if one of them is invalid. $\mathcal{P}_i$ gets the output
  a) $\{\delta_{i,j}\}_{j\in S} \leftarrow \mathsf{MPMtAOut}(dk_i, \{\beta_{i,j}, C_{\alpha_{i,j}}\}_{j\in S})$
  b) $\{\zeta_{i,j}\}_{j\in S} \leftarrow \mathsf{MPMtAOut}(dk_i, \{\hat\beta_{i,j}, C_{\hat\alpha_{i,j}}\}_{j\in S})$
  c) $R = \sum_{j\in S} \lambda_{j,S} R_j$, $r = R|_{x\text{-axis}} \bmod q$.

$$(\sum_{i\in S} \lambda_{i,S} k_i) \cdot (\sum_{i\in S} \lambda_{i,S}\gamma_i) \text{ and } (\sum_{i\in S} \lambda_{i,S} x_i) \cdot (\sum_{i\in S} \lambda_{i,S}\gamma_i).$$

线上阶段

- One-Round Interaction: $\mathcal{P}_i$ does
  a) pick two random $(t-1)$-degree polynomials $f(\cdot), f'(\cdot)$ with $f(0) = f'(0) = 0$
  b) mask $\delta_{i,j}$: $\delta_{i,j} = \delta_{i,j} + f(j) \bmod q$ for each $j \in S$
  c) generate $(m + rx)\gamma$'s share: $\{\chi_{i,j} = m\gamma_i + r\zeta_{i,j} + f'(j) \bmod q\}_{j\in S}$
  d) generate $D_i = \gamma_i R, \Gamma_i = \gamma_i(mG + rX)$
  e) generate $\psi_i \leftarrow \mathsf{Prove}_{\mathsf{DDH}}((R, D_i, mG + rX, \Gamma_i), \gamma_i)$
  f) broadcast $(\{\delta_{i,j}, \chi_{i,j}\}_{j\in S}, \psi_i)$.
- Online Output: On receiving $\{\delta_{j,\nu}, \chi_{j,\nu}\}_{\nu\in S}$ for each $j \in S$, assemble $s = \frac{\sum_{j,\nu\in S} \lambda_{j,S}\lambda_{\nu,S} \cdot \chi_{j,\nu}}{\sum_{j,\nu\in S} \lambda_{j,S}\lambda_{\nu,S} \cdot \delta_{j,\nu}}$ if $|S| \geq t$. Run $b \leftarrow \mathsf{Verify}(X, (r, s), \mathsf{msg})$. It distinguishes two cases.
  a) If $b = 1$, output the signature (r, s).
  b) Else, the honest parties identify the cheaters:
     i) generate $D_j = \sum_{\nu\in S} \lambda_{\nu,S}(\delta_{j,\nu}G - B_{j,\nu} + B_{\nu,j})$
     ii) generate $\Gamma_j = \sum_{\nu\in S} \lambda_{\nu,S}(\chi_{j,\nu}G - r\hat B_{j,\nu} + r\hat B_{\nu,j})$
     iii) verify the DDH-ZKP ψ_j for the statement $(R, D_j, mG + rX, \Gamma_j)$ and set $S = S \setminus \{j\}$ if it fails.

     If $|S| \geq t$ still holds, re-assemble $s = \frac{\sum_{j,\nu\in S} \lambda_{j,S}\lambda_{\nu,S} \cdot \chi_{j,\nu}}{\sum_{j,\nu\in S} \lambda_{j,S}\lambda_{\nu,S} \cdot \delta_{j,\nu}}$ and output (r, s).

Case b) is required only when misbehavior exists.

鲁棒阈值BBS+方案

线下阶段

- Round 1: $\mathcal{P}_i$ does the following
  a) pick $\gamma_i \leftarrow\$ \mathbb{Z}_q$ and generate $(C_{\gamma_i}, \tau_i) \leftarrow \mathsf{MPMtA}_1(\gamma_i, ek_i)$
  b) pick $e_i, s_i \leftarrow\$ \mathbb{Z}_q$ and compute $\bar{e}_i \leftarrow \mathsf{H}(e_i)$, $\bar{s}_i \leftarrow \mathsf{H}(s_i)$ where $\mathsf{H}(\cdot)$ is a random oracle
  c) broadcast $(C_{\gamma_i}, \tau_i, \bar{e}_i, \bar{s}_i)$.
- Round 2: On receiving τ_j from each $\mathcal{P}_j$, verify them and set $S = S \setminus \{j\}$ if it fails. $\mathcal{P}_i$ does
  a) $(\{C_{\hat{\alpha}_{j,i}}, \hat{\beta}_{i,j}\}_{j\in S}, \hat{\tau}'_i) \leftarrow \mathsf{MPMtA}_2(\{C_{\gamma_j}\}_{j\in S}, x_i)$
  b) $\hat{B}_{i,j} = \hat{\beta}_{i,j} G_2$ for each $j \in S$
  c) broadcast $(\{C_{\hat{\alpha}_{j,i}}, \hat{B}_{i,j}\}_{j\in S}, \hat{\tau}'_i, e_i, s_i)$.
- Offline Output: Upon receiving $\hat{\tau}'_j$ from each $\mathcal{P}_j$, set $S = S \setminus \{j\}$ if it is not valid, $\bar{e}_j \neq \mathsf{H}(e_j)$, or $\bar{s}_j \neq \mathsf{H}(s_j)$. $\mathcal{P}_i$ gets the output
  a) $\{\zeta_{i,j}\}_{j\in S} \leftarrow \mathsf{MPMtAOut}(dk_i, \{\hat{\beta}_{i,j}, C_{\hat{\alpha}_{i,j}}\}_{j\in S})$
  b) $e = \sum_{j\in S} e_j \bmod q$, $s = \sum_{j\in S} s_j \bmod q$.

线上阶段

- One-Round Interaction: $\mathcal{P}_i$ does
  a) pick a random $(t-1)$-degree polynomial $f(\cdot)$ with $f(0) = 0$
  b) generate $(x+e)\gamma$'s share: $\{\chi_{i,j} = e\gamma_i + \zeta_{i,j} + f(j)\}_{j\in S}$
  c) generate $R_i = \gamma_i D$ and broadcast $(\{\chi_{i,j}\}_{j\in S}, R_i)$.
- Online Output: On receiving $\{\chi_{j,\nu}\}_{\nu\in S}$ from each $\mathcal{P}_j$, assemble $A = \dfrac{\sum_{j\in S}\lambda_{j,S} R_j}{\sum_{j,\nu\in S}\lambda_{j,S}\lambda_{\nu,S}\cdot\chi_{j,\nu}} \in \mathbb{G}_1$ if $|S| \geq t$. Run $b \leftarrow \mathsf{Verify}(X, (A, e, s), \mathbf{m})$. It distinguishes two cases.
  a) If $b = 1$, output the signature $\sigma = (A, e, s)$.
  b) Else, the parties identify the cheaters:
     i) generate $\Gamma_j = \sum_{\nu\in S}\lambda_{\nu,S}(\chi_{j,\nu} G_2 - \hat{B}_{j,\nu} + \hat{B}_{\nu,j})$
     ii) check $\mathsf{e}(R_j, eG_2 + X) \stackrel{?}{=} \mathsf{e}(D, \Gamma_j)$ and set $S = S \setminus \{j\}$ if it fails.
     If $|S| \geq t$ still holds, re-assemble $A = \dfrac{\sum_{j\in S}\lambda_{j,S} R_j}{\sum_{j,\nu\in S}\lambda_{j,S}\lambda_{\nu,S}\cdot\chi_{j,\nu}}$ and output $\sigma = (A, e, s)$.
  Case b) is required only when misbehavior exists.

05

性能评估

实验设置

测试环境

实验在MacBookPro上运行，搭载macOSMonterey12.3，配备16GBRAM及AppleM1Pro芯片。

软件库

使用BICYCL库进行类群组算术运算，mcl库实现双线性群运算，确保高效性能。

安全参数

设定计算安全级别λ=128位，统计安全参数λs=40位，采用SHA256哈希函数。

加密配置

ECDSA基于secp256k1椭圆曲线，BBS+基于BLS12381配对曲线，CL加密使用256位明文空间和1827位ΔK。

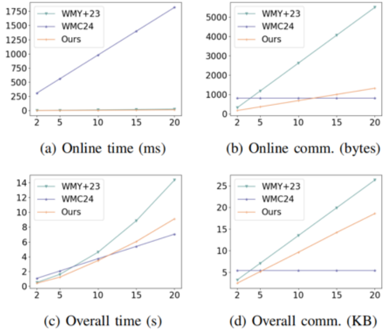
结果对比

TABLE 4: Threshold ECDSA's online and offline costs, with running time (ms) and sending communication (kilobytes) per party for a varying number of parties.

Schemes		2		4		5		10		15		20	
		Time	Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time	Comm.	Time	Comm.
online	DKLs24 [36]	0.1	0.06	0.1	0.06	0.1	0.06	0.12	0.06	0.14	0.06	0.15	0.06
	CGG$^+$20 [20]	502	0.03	638	0.03	775	0.03	1450	0.03	2282	0.03	2966	0.03
	WMY$^+$23 [58]	1.22	0.32	2.18	0.6	3.18	0.9	11.3	2.6	18.3	4.1	26.3	5.5
	WMC24 [57]	308	0.8	392	0.8	476	0.8	890	0.8	1400	0.8	1820	0.8
	Ours	0.5	0.2	0.78	0.24	1.18	0.3	5.1	0.7	9.1	1.0	14.1	1.32
offline	DKLs24 [36]	4	53	9	159	10.9	212	25.4	477	39.1	743	50.7	1008
	CGG$^+$20 [20]	830	9	1338	13.5	2175	18.6	7518	35	14465	46.4	23452	63
	WMY$^+$23 [58]	506	2.92	816	4.5	1326	6.2	4584	11.4	8820	15.8	14300	21.2
	WMC24 [57]	1023	4.6	1303	4.6	1584	4.6	2740	4.6	3980	4.6	5220	4.6
	Ours	406	2.3	780	3.6	1224	4.9	3460	9.0	6030	13.1	9080	17.2

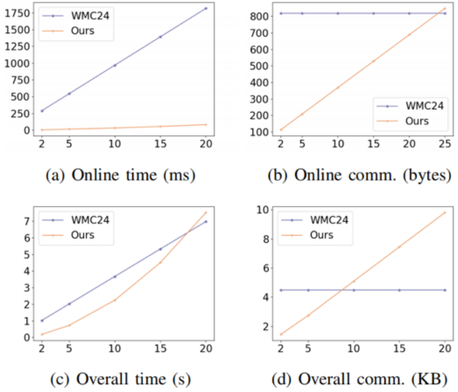
TABLE 5: Threshold BBS+'s online and offline costs, with running time (ms) and sending communication (kilobytes) per party for a varying number of parties.

Schemes		2		4		5	
		Time	Comm.	Time	Comm.	Time	Comm.
online	[57]	287	0.8	372	0.8	457	0.8
	Ours	2.48	0.11	4.78	0.14	7.28	0.17
offline	[57]	979	3.6	1225	3.6	1471	3.6
	Ours	174	1.3	327	1.7	510	2.2



(a) Online time (ms)
(b) Online comm. (bytes)
(c) Overall time (s)
(d) Overall comm. (KB)

Figure 3: Comparisons of robust threshold ECDSA for the number of parties $n = 2$ to 20.



(a) Online time (ms)
(b) Online comm. (bytes)
(c) Overall time (s)
(d) Overall comm. (KB)

Figure 4: Comparisons of robust threshold BBS+

与非鲁棒方案[20]、[31]、[36]的对比分析：

- **基于Paillier算法的CGG+20 [20]方案**：能够实现可识别中止机制，其在线计算成本显著高于WMC24方案，甚至更高。该方案的在线通信效率达到最优水平。但其离线计算和通信效率均不及我们方案，且通信开销是我们的三倍。
- **基于OT协议的DKLs24 [36]方案**：仅能通过中止机制实现安全性，其在线计算效率达到最优，仅需一次签名验证即可完成验证过程。此外，其离线阶段的处理速度也相当快。不过，其离线通信量明显高于我们团队及其他研究者的方法。
- **CDKS24 [31]**：在DKLs24基础上增加了作弊者识别（CI）功能，实现了可识别的中止机制。由于目前尚无开源代码，仅提供理论分析。他们的在线CI流程涉及两个与Pedersen承诺相关的零知识证明和两个与离散对数相关的零知识证明，成本大约是我们的两倍。另一方面，他们的离线阶段性能与DKLs24相当。因此，虽然其运行速度远超我们方案，但所需的带宽要求却显著更高。

结果对比

带宽消耗低

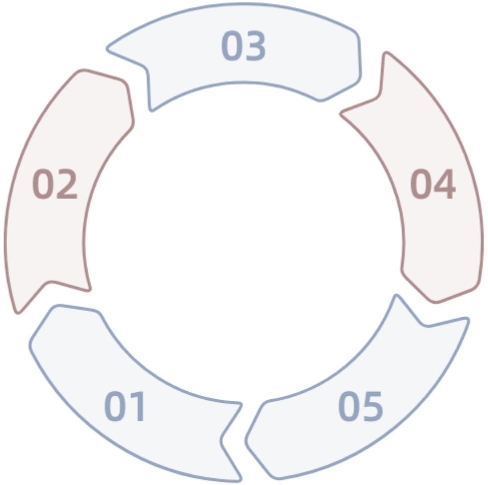
带宽消耗低，仅需0.2至1.32KB，特别适合网络资源有限的情况。

实时响应能力

展现出色的实时响应能力，适用于需要快速反应的应用场景。

处理速度提升

本方案在在线阶段处理速度上显著提升，比WMY+23快2.5倍，比WMC24快数百倍。



小规模场景优势

在小规模场景下（n≤12）运行时间更快，尤其当n≤5时带宽更低。

适用范围广泛

特别适合小型或关键应用，能够有效提高效率和性能。

THANKS