Curve Treeを用いたリング署名における署名・検証時間の効率化

早川 慧梧[1,a)]　中西 透[1,b)]　北須賀 輝明[1,c)]　連　卓涛[1,d)]

概要：リング署名とは，複数のユーザの公開鍵集合（リング）に対して，リング内のいずれかの鍵に対応した秘密鍵で匿名で署名できる署名方式である．従来方式として DualRing と呼ばれる効率的なリング署名が提案されている．この DualRing ではリング内の公開鍵数を n とした場合，署名生成・検証時間が $O(n)$ であり，匿名性のために n を増加させると処理時間が実用的でなくなる．本研究では，Curve Tree と呼ばれるゼロ知識メンバシップ証明システムを用いることで，$O(\log n)$ の署名長は維持しつつ，リングが変化しない場合において，n に対して sublinear な署名生成・検証時間となるリング署名を提案する．そして PC 上で署名長と署名生成・検証時間を計測することで，提案方式の有効性を考察する．

Improving signing and verification efficiency in ring signatures using Curve Tree

Keigo Hayakawa[1,a)]　Toru Nakanishi[1,b)]　Teruaki Kitasuka[1,c)]　Zhuotao Lian[1,d)]

Abstract: In ring signature schemes, a signer can generate a signature using the public keys of multiple users (referred to as the ring), thereby anonymously demonstrating that their own key is included within the ring. An efficient ring signature scheme called DualRing has been proposed. In DualRing, given n keys in the ring, the signature generation and verification times are $O(n)$, and thus it is not scalable. In this paper, we propose a ring signature scheme that leverages a zero-knowledge membership proof system called Curve Tree, achieving sublinear signature generation and verification time with respect to n in cases that the ring is not changed, while maintaining the $O(\log n)$ signature size. We evaluate the effectiveness of the proposed scheme by measuring the signature size as well as the signing and verification times on a PC.

1. はじめに

ディジタル署名では，正当な署名者の秘密鍵によって作成されたものであることと，メッセージが偽装されていないことを署名者の公開鍵を用いて検証できる．このため正当な署名者であることを保証する一方で，署名者が特定されてしまう．これに対して，特定のグループに属するメンバーの一人であることを検証できる一方で，署名を行った人物を特定することができないリング署名 [1] が提案されている．リング署名では，あるグループに属する署名者がそのグループの全ユーザの公開鍵（リング）を用いて，リング内に自身の鍵が含まれていることを示す署名を生成する．署名の検証の際には署名者がどの公開鍵を保持しているのかを特定することが不可能となっており，匿名性を有する．また，信頼される管理者を必要とせず，署名者個人のみで署名を行うことができるため，内部告発や，プライバシーを保護した暗号通貨・ブロックチェーン取引などの非中央集権型のシステムに応用されている．従来方式として，DualRing [2] と呼ばれる効率的なリング署名が 2021 年に Yuen らによって提案されている．DualRing ではコミットメントとチャレンジの二重のリング構造の二つを用いることにより，シンプルな群演算で署名生成できる点と，ハッシュを行う回数が少ない点で効率化を図っている．安

[1] 広島大学大学院先進理工系科学研究科
Graduate School of Advanced Science and Engineering, Hiroshima University
[a)] m251973@hiroshima-u.ac.jp
[b)] t-nakanishihiroshima-u.ac.jp
[c)] kitasukahiroshima-u.ac.jp
[d)] lian-zhuotaohiroshima-u.ac.jp

全性の仮定は離散対数仮定とランダムオラクルモデルに基づいており，信頼できる公開パラメータの生成処理を必要としないため非中央集権的環境に適する．この DualRing ではリング内のユーザーの数を n とした場合，その署名長は $O(\log n)$ と効率的であるが，その署名生成・検証時間は $O(n)$ となっている．このため，匿名性のために n を増加させると処理時間が実用的でなくなるという問題がある．

この問題を解決するために，本研究では，Curve Tree [3] と呼ばれるゼロ知識メンバシップ証明システムを用いた効率的なリング署名を提案する．Curve Tree ではコミットメント集合を Merkle Tree により効率的に圧縮することができ，葉ノードの自身のコミットメントに対するセットメンバシップ証明をゼロ知識で効率的に sublinear の計算で証明できる．この Curve Tree の葉ノードをリング署名の離散対数型の公開鍵に置き換え，その秘密鍵の知識をゼロ知識証明する知識の署名（SPK）と組み合わせることにより，公開鍵がリングに含まれていることを証明しながら，署名を行うことができる．こうして提案するリング署名では，署名長は $O(\log n)$ を維持しながら，リングの公開鍵集合が変化せず以前に構成した木を利用できる場合において，署名生成・検証時間を n の sublinear となる時間に削減できる．本研究ではさらに PC 上で実装し，n の変化に対して署名長と処理時間をリングが変化しない場合と変化する場合において計測することで，提案方式の有効性を考察する．

2. 数学的準備

2.1 群の定義と表記

有限体 $\mathbb{F}_q$ 上の楕円曲線 $\mathbb{E}$ に対して，楕円曲線上の点群 $\mathbb{E}[\mathbb{F}_q]$ はアーベル群である．群の位数 $p := |\mathbb{E}[\mathbb{F}_q]|$ は素数とし，こうして $\mathbb{E}[\mathbb{F}_q]$ は巡回群である．また，G を $\mathbb{E}[\mathbb{F}_q]$ の生成元とする．$\mathbb{F}_p \cong \mathbb{Z}/(p\mathbb{Z})$ を $\mathbb{E}[\mathbb{F}_q]$ のスカラー体と呼び，$[s] \cdot G$ と表記する．ただし，$s \in \mathbb{F}_p, G \in \mathbb{E}[\mathbb{F}_q]$ である．また $\langle \vec{s}, \vec{G} \rangle = \sum_i [s_i] \cdot G_i$ はスカラーのベクトルである $\vec{s} \in \mathbb{F}_p^n$ と楕円曲線上の点のベクトルである $\vec{G} \in \mathbb{E}[\mathbb{F}_q]^n$ の内積を意味する．

2.2 安全性仮定

提案方式は，Curve Tree [3] を利用しており，以下の仮定に基づく．

一般化離散対数仮定：$\mathcal{G}(1^\lambda)$ が新しい楕円曲線をサンプリングするとき，全ての PPT（確率的多項式時間）攻撃者 $\mathcal{A}$ に対して，$m \geq 2$ のとき以下は無視可能な確率である．

$$\Pr\left[\begin{array}{l|c} (\mathbb{E}, \mathbb{F}_q, \mathbb{F}_p) \leftarrow \mathcal{G}(1^\lambda) & \langle \vec{a}, \vec{G} \rangle = 0 \in \mathbb{E}[\mathbb{F}_q] \\ \vec{G} \xleftarrow{\$} \mathbb{E}[\mathbb{F}_q]^m & \wedge\ \vec{a} \neq \vec{0} \in (\mathbb{F}_p)^m \\ \vec{a} \leftarrow \mathcal{A}((\mathbb{E}, \mathbb{F}_q, \mathbb{F}_p), \vec{G}) & \end{array}\right]$$

2.3 コミットメント

コミットメントとは，送信者がある情報をコミットメントとして秘匿し，コミットメントのみを公開することで，第三者がその情報の正当性を後から検証することができる方式である．コミットメントは以下の二つの安全性を持つ．

binding: 異なるデータから同一のコミットメントを作成することはできない．

hiding: あるコミットメントから元のコミットされた値を知ることはできない．

提案方式では，Curve Tree と同様，コミットメントとして Pedersen commitment [4] 方式を用いる．入力 $\vec{v} \in \mathbb{F}_p^l$ に対して乱数 $r \xleftarrow{\$} \mathbb{F}_p$ を用いて計算されるコミットメントを以下に示す．

$$C = \mathsf{Com}(\vec{v}; r) = \langle \vec{v}, \vec{G} \rangle + [r] \cdot H \in \mathbb{E}[\mathbb{F}_q]$$

ただし，$\mathbb{F}_p = |\mathbb{E}[\mathbb{F}_q]|, G_1, .., , G_l, H \in \mathbb{E}[\mathbb{F}_q]$ である．Pedersen commitment 方式ではコミットメントの rerandomize を行うことができる．新たな乱数 $\delta \xleftarrow{\$} \mathbb{F}_p$ を用いた rerandomize を以下に示す．

$$\begin{aligned} C^* &= C + [\delta] \cdot H \\ &= \langle \vec{v}, \vec{G} \rangle + [r + \delta] \cdot H \end{aligned}$$

このとき，C^* は $\vec{v} \in \mathbb{F}_p^l$ に対して乱数 $r + \delta \in \mathbb{F}_p$ を用いて計算されるコミットメントであり，元の C を識別することができない．

2.4 ゼロ知識証明

知識のゼロ知識証明とは，証明者がもつ情報がある関係を満たしていることを，その情報を明らかにすることなく証明するものである．また，Fiat-Shamir heuristic を用いることで知識の証明（Proofs of Knowledge）を NIZK arguments や知識の署名へ変換することができる．

2.4.1 NIZK arguments

ある NP 関係 $R(x, w)$ を表す際に以下の Camenisch-Stadler notion を用いる．このとき，秘密証拠情報 w は括弧で囲み，公開情報 x は括弧以外の部分で定義する．

$$R := \{(z) : Y = [z] \cdot G\}$$

上記は，離散対数の関係を表しており，公開情報は楕円曲線 $\mathbb{E}$ 上の生成元 G，楕円曲線 $\mathbb{E}$ 上の点 $Y \in \mathbb{E}$，秘密証拠情報はスカラー $z \in \mathbb{F}_p$ である．

提案方式で使用している Curve Tree [3] では，Pedersen commitment に対する NIZK arguments として Bulletproofs [5] を用いている．2.3 節で述べた Pedersen commitment のような rerandomize 可能なコミットメントの集合 $S = \{C_1, ..., C_n\}$ に対して，以下を証明することを考える．

$$\{(r, i) : \hat{C} = \mathsf{Rerand}(C_i, r)\}$$

上記は，コミットメント $\hat{C}$ が S 内のあるコミットメントの rerandomize であるかをどのコミットメントを rerandomize したのかを明らかにすることなく示す．これにより，$C_i \in S$ の set membership を証明できる．

Bulletproofs では，演算回路における内積証明を可能にしており，演算回路の計算が正しいことを効率的に証明できる．こうして任意の NP 関係を証明可能であり，Curve Tree では Bulletproofs を用いることにより効率的に set membership を証明している．なお，Bulletproofs は statistical zero-knowledge を満たしており，検証者が選ぶチャレンジが与えられたときに，秘密情報なしでシミュレート可能であり，証明者と検証者間のプロトコルの通信内容とシミュレートした通信内容が統計的に区別不可能である．

2.4.2 知識の署名 SPK（Signature based on Proof of Knowledge）

提案方式では，$R := \{(x_1, x_2) : Y = [x_1] \cdot G_1 + [x_2] \cdot G_2\}$ なる関係を以下の離散対数に対する SPK を使用して署名生成している．SPK とは，ある関係 $R(x, w)$ を持つ秘密情報 w を知っていることを秘密情報を漏らさずに証明者と検証者の対話によって証明する（PK）を Fiat-Shamir heuristic を用いることで安全なディジタル署名に変換したものである．このとき，completeness，special soundness，special honest-verifier zero-knowledge を PK が満たしている場合に SPK に変換することができる．

元の PK についての安全性は以下のとおりである．

special soundness: 2 つの受理されるプロトコルの通信内容が与えられたとき，秘密情報 w を計算する多項式時間アルゴリズムが存在する．

special honest-verifier zero-knowledge: 検証者が選ぶチャレンジが与えられたときに，プロトコル通りに動くような検証者（honest-verifier）に対して秘密情報 w なしでシミュレート可能であり，証明者と検証者間のプロトコルの通信内容とシミュレートした通信内容の確率分布が等しい．

2.5 Curve Tree

Curve Tree [3] は，2023 年に Campanelli らによって提案されており，信頼できるパラメータの生成を必要とせず，set membership をゼロ知識で証明するための効率的なシステムである．安全性の仮定は，離散対数仮定とランダムオラクルモデルに依存している．特徴として，集合を圧縮する暗号データ構造である accumulator を浅い Merkle Tree と楕円曲線の 2 サイクルを組み合わせることで，実現している．Curve Tree の葉ノードや内部ノードは楕円曲線上の点として表される．また，圧縮を行うために，その層の楕円曲線に適切にインスタンス化された Pedersen commitment を各層で交互に用いることにより，ハッシュ関数を利用せずに Merkle Tree を構成している．これにより，Bulletproofs に基づいた効率的な set membership 証明を実現している．

2.5.1 Curve Tree の定義

楕円曲線の 2 サイクルは，以下を満たす 2 つの楕円曲線 $\{\mathbb{E}_{(\mathrm{evn})}, \mathbb{E}_{(\mathrm{odd})}\}$ と 2 つの素数体 $\{\mathbb{F}_p, \mathbb{F}_q\}$ から構成される．

$$p = |\mathbb{E}_{(\mathrm{evn})}[\mathbb{F}_q]| \wedge q = |\mathbb{E}_{(\mathrm{odd})}[\mathbb{F}_p]|$$

上記は，2 つの楕円曲線の基礎体とスカラー体が相補的であることを示す．したがって，点 $(\mathbb{X}, \mathbb{Y}) \in \mathbb{E}_{(\mathrm{evn})}[\mathbb{F}_q]$ は，一方の楕円曲線 $\mathbb{E}_{(\mathrm{odd})}$ 上のスカラーのペアとして扱うことができる．このことから，$\mathbb{E}_{(\mathrm{evn})}$ 上の点 $(\mathbb{X}_i, \mathbb{X}_i)$ のリスト $(\vec{\mathbb{X}}, \vec{\mathbb{Y}})$ に対する $\mathbb{E}_{(\mathrm{odd})}$ 上の Pedersen commitment を $\langle \vec{\mathbb{X}}, \vec{G}^{\mathsf{x}}_{\mathbb{E}_{(\mathrm{odd})}} \rangle + \langle \vec{\mathbb{Y}}, \vec{G}^{\mathsf{y}}_{\mathbb{E}_{(\mathrm{odd})}} \rangle \in \mathbb{E}_{(\mathrm{odd})}$ として計算できる．この圧縮関数を $f_{\mathbb{E}_{(\mathrm{odd})}} : {\mathbb{E}_{(\mathrm{evn})}}^l \mapsto \mathbb{E}_{(\mathrm{odd})}$ と表記する．$\mathbb{E}_{(\mathrm{evn})}$ と $\mathbb{E}_{(\mathrm{odd})}$ を入れ替えた場合についても，同様に $f_{\mathbb{E}_{(\mathrm{evn})}} : {\mathbb{E}_{(\mathrm{odd})}}^l \mapsto \mathbb{E}_{(\mathrm{evn})}$ と表記する．この二つの圧縮関数を木の各層で交互に使用することで，一方の圧縮関数の出力をもう一方の圧縮関数の入力として使用することができる．　以下，木の特定の層のノードがある楕円曲線上の点として表されているとき，その楕円曲線を $\mathbb{E}_{(\_)}$ と表記し，もう一方の楕円曲線を $\mathbb{E}_{other(\_)}$ と表記する．もし，$\mathbb{E}_{(\_)}$ が $\mathbb{E}_{(\mathrm{evn})}$ のとき，$\mathbb{E}_{\mathrm{other}(\_)}$ は $\mathbb{E}_{(\mathrm{odd})}$ である．

Curve Tree は次の 5 つの要素によってパラメータ化される．（I）木の深さ $\mathsf{D} \in \mathbb{N}$，（II）分岐数 $l \in \mathbb{N}$，（III）楕円曲線の 2 サイクル $(\mathbb{E}_{(\mathrm{evn})}, \mathbb{E}_{(\mathrm{odd})}, \mathbb{F}_p, \mathbb{F}_q)$，（IV）$2l$ 個の点 $\vec{G}^{\mathsf{x}}_{(\mathrm{evn})}, \vec{G}^{\mathsf{y}}_{(\mathrm{evn})} \in \mathbb{E}^l_{(\mathrm{evn})}$，（V）$2l$ 個の点 $\vec{G}^{\mathsf{x}}_{(\mathrm{odd})}, \vec{G}^{\mathsf{y}}_{(\mathrm{odd})} \in \mathbb{E}^l_{(\mathrm{odd})}$．このとき，Curve Tree の木は以下のように再帰的に定義される．

葉ノード: $(0, l, \mathbb{E}_{(\_)}, \mathbb{E}_{\mathrm{other}(\_)}) - \mathsf{CurveTree}$: 深さ 0 の木として葉ノードは楕円曲線上の点 $C \in \mathbb{E}_{(\_)}$ に対応づけられる．

親ノード: $(\mathsf{D}, l, \mathbb{E}_{(\_)}, \mathbb{E}_{\mathrm{other}(\_)}) - \mathsf{CurveTree}$: 深さ D の（部分）木としてその根ノードは l 個の深さ $\mathsf{D}-1$ の部分木 $(\mathsf{D}-1, l, \mathbb{E}_{\mathrm{other}(\_)}, \mathbb{E}_{(\_)}) - \mathsf{CurveTree}$ の根ノードとなる子ノードをもち，$C \in \mathbb{E}_{(\_)}$ に対応づけられる．このとき，子ノードを $C_1 = (\mathbb{X}_1, \mathbb{Y}_1) \in \mathbb{E}_{\mathrm{other}(\_)}, ..., C_l = (\mathbb{X}_l, \mathbb{Y}_l) \in \mathbb{E}_{\mathrm{other}(\_)}$ とすると，その親ノードは以下のコミットメント C として定義される．

$$C = \langle \vec{\mathbb{X}}, \vec{G}^{\mathsf{x}}_{(\_)} \rangle + \langle \vec{\mathbb{Y}}, \vec{G}^{\mathsf{y}}_{(\_)} \rangle$$

ただし，楕円曲線は各層で入れ替わる．

次に，この Curve Tree において set membership 証明を実現する select-and-rerandomize 方式を以下に示す．この方式は以下の 6 つのアルゴリズムからなる．

- $\mathsf{SelRerand.Setup}(1^\lambda) \to \mathsf{pp}$
  セキュリティパラメータ λ を入力として，スキームの

公開パラメータを返す．そのパラメータは transparent であり，信頼される第三者を必要としない．

- $\mathsf{SelRerand.Comm}(\mathsf{pp}, v_{\text{leaf}}, o) \to C$
  入力 v_{leaf} に対して乱数 o を用いて v_{leaf} のコミットメントを得る．
- $\mathsf{SelRerand.Rerand}(\mathsf{pp}, C, r) \to \hat{C}$
  コミットメント C に対して乱数 r を用いて rerandomize を行い $\hat{C}$ を出力する．
- $\mathsf{SelRerand.Accum}(\mathsf{pp}, S) \to \mathsf{rt}$
  コミットメントの集合 S を入力として圧縮し，木の根ノード rt を出力する．このとき，集合の要素の順序は一貫性を持つ．
- $\mathsf{SelRerand.Prove}(\mathsf{pp}, S, C, r) \to \pi$
  コミットメント C に対して乱数 r を適用し rerandomize されたコミットメントを用いて，$C \in S$ を示す証明 π を返す．
- $\mathsf{SelRerand.Vfy}(\mathsf{pp}, \mathsf{rt}, \hat{C}, \pi) \to \{0,1\}$
  rt を根ノードとする木の葉ノードの集合に対して $\hat{C}$ がその集合の中の要素の rerandomize であることを示す証明 π を検証し，正当なら 1 を，そうでなければ 0 を出力する．

2.5.2 Curve Tree の構成の概要

Curve Tree では membership 証明のために，次のアプローチをとりながら，根ノードから一層ずつ対象の葉ノードまで証明を行う．各深さ D において $(\mathsf{D}, l, \mathbb{E}_{(\_)}, \mathbb{E}_{\text{other}(\_)})-\mathsf{CurveTree}$ の rerandomize されたコミットメントに対して，葉ノードへの子ノードを選ぶ．その選んだ子ノードを rerandomize したコミットメントを出力する．Curve Tree の一つの層において証明する関係を $\mathcal{R}^{(\mathsf{single-level},(\mathrm{evn}))}$（もしくは $\mathcal{R}^{(\mathsf{single-level},(\mathrm{odd}))}$）と表記する．関係 $\mathcal{R}^{(\mathsf{single-level},(\mathrm{evn}))}$（もしくは $\mathcal{R}^{(\mathsf{single-level},(\mathrm{odd}))}$）の入力は以下である．

公開情報: rerandomize された子ノードのコミットメント $\hat{C} \in \mathbb{E}_{(\mathrm{odd})}$（もしくは，$\mathbb{E}_{(\mathrm{evn})}$），その親ノードのコミットメント $C \in \mathbb{E}_{(\mathrm{evn})}$（もしくは，$\mathbb{E}_{(\mathrm{odd})}$）

秘密証拠情報: $\hat{C}$ が C の i 番目の子の rerandomize である場合のインデックス i，Pedersen commitment における opening スカラー $r, \delta, \vec{\mathbb{X}}, \vec{\mathbb{Y}}$

この関係は偶数，奇数それぞれの層において親ノードが $\mathbb{E}_{(\_)}$ 上にあるとすると，以下のように定義される．

$$\mathcal{R}^{(\mathsf{single-level},\mathbb{E}_{(\_)})} := \left\{ \begin{pmatrix} i, r, \delta \\ \vec{X}, \vec{Y} \end{pmatrix} : \begin{array}{c} C = \langle [\vec{\mathbb{X}}], \vec{G}^{\mathsf{x}}_{(\_)} \rangle \\ \quad + \langle [\vec{\mathbb{Y}}], \vec{G}^{\mathsf{y}}_{(\_)} \rangle \\ \quad + [r] \cdot H_{(\_)} \in \mathbb{E}(\_) \\ \hat{C} = (\mathbb{X}_i, \mathbb{Y}_i) + [\delta] \cdot H_{\text{other}(\_)} \in \mathbb{E}_{\text{other}(\_)} \end{array} \right\}$$

上記の関係は奇数層と偶数層において以下のようにまとめることができ，Prove で Bulletproofs により証明される．

$$\mathcal{R}^{(\mathsf{evn-levels})} := \left\{ \bigwedge_{j \in \{0,2,\ldots,\mathsf{D}-2\}} \mathcal{R}^{(\mathsf{single-level},(\mathrm{evn}))} \right\}$$

$$\mathcal{R}^{(\mathsf{odd-levels})} := \left\{ \bigwedge_{j \in \{1,3,\ldots,\mathsf{D}-1\}} \mathcal{R}^{(\mathsf{single-level},(\mathrm{odd}))} \right\}$$

なお，提案方式では Curve Tree を改良した Curve Forest [6] を使用している．違いとして，敵対者が生成した悪意のある accumlator によって矛盾する証明が出来ない点や，加法性を活かして最適化している点がある．これにより，安全性の向上やバッチ処理時のコストを効率化をすることができる．

2.5.3 Curve Tree の安全性

select-and-rerandomize の安全性は以下の binding，hiding，zero-knowledge により定義される．

binding: どんな PPT 攻撃者 $\mathcal{A}$ に対しても，以下の確率が無視可能であるとき binding を持つ．

$$\Pr\left[\begin{array}{c} \mathsf{pp} \leftarrow \mathsf{SelRerand.Setup}(1^\lambda); \\ (\vec{v}, \vec{o}, \hat{v}, \hat{o}, \pi) \leftarrow \mathcal{A}(\mathsf{pp}); \\ \forall i \in [n] : C_i \leftarrow \mathsf{SelRerand.Comm}(\mathsf{pp}, v_i, o_i); \\ \hat{C} \leftarrow \mathsf{SelRerand.Comm}(\mathsf{pp}, \hat{v}, \hat{o}); \\ \mathsf{rt} \leftarrow \mathsf{SelRerand.Accum}(\mathsf{pp}, \{C_1, \ldots, C_n\}) : \\ \hat{v} \notin \vec{v} \wedge \mathsf{SelRerand.Vfy}(\mathsf{pp}, \mathsf{rt}, \hat{C}, \pi) = 1 \end{array} \right]$$

Curve Tree が離散対数仮定の下で binding をもつことが証明されている．

zero-knowledge: 任意の $\lambda \in \mathbb{N}$，任意の stateful な敵対者 $\mathcal{A}$，任意の $j^* \in [|S|]$ に対して，以下において $p_0 = p_1$ であるような効率的なシミュレーター $\mathcal{S}$ が存在するとき，zero-knowledge であるという．

$$p_b := \Pr\left[\begin{array}{c} \mathsf{pp} \leftarrow \mathsf{SelRerand.Setup}(1^\lambda); \\ (v_1, \ldots, v_n, o_1, \ldots o_n) \leftarrow \mathcal{A}(\mathsf{pp}); \\ S := \{C_i = \mathsf{SelRerand.Comm}(\mathsf{pp}, v_i, o_i)\}_{i \in [n]}; \\ r \xleftarrow{\$} \mathbb{F}; \hat{C} = \mathsf{SelRerand.Comm}(\mathsf{pp}, C_{j^*}, r); \\ \pi \leftarrow X_b(\mathsf{pp}, S, C, \hat{C}, r) : \\ \mathcal{A}(\mathsf{pp}, \hat{C}, \pi) = 1 \end{array} \right]$$

ただし，$X_0(\mathsf{pp}, S, C, \hat{C}, r) := \mathcal{S}(\mathsf{pp}, S, \hat{C})$ かつ $X_1(\mathsf{pp}, S, C, \hat{C}, r) := \mathsf{SelRerand.Prove}(\mathsf{pp}, S, C, r)$ である．[3] において Curve Tree の zero-knowledge も示されている．

3. リング署名

DualRing [2] と同様のリング署名のセキュリティモデルを以下に示す．リング署名は以下の 4 つの PPT アルゴリズムから構成される．

- $\mathsf{Setup}(\lambda) \to \mathsf{param}$
  セキュリティパラメータ λ を入力として，スキームの公開パラメータを返す．そのパラメータは transparent であり，信頼される第三者を必要としない．
- $\mathsf{KeyGen}(\mathsf{param}) \to (\mathsf{pk}, \mathsf{sk})$

入力 param から鍵ペア pk, sk を出力する．

- $\mathsf{Sign}(\mathsf{param}, M, \mathbf{pk}, \mathsf{sk}) \to \sigma$
  署名されるメッセージ M，リング $\mathbf{pk}$，秘密鍵 sk を入力として，リング署名 σ を出力する．
- $\mathsf{Verify}(\mathsf{param}, M, \mathbf{pk}, \sigma) \to 1/0$
  メッセージ M，リング $\mathbf{pk}$，M に対するリング署名 σ を入力として，リング署名の検証結果（0,1）を出力する．

$\mathbf{pk}$ は公開鍵のベクトル（$\mathsf{pk}_1, ...\mathsf{pk}_n$）であり，簡単のため、本論文の残りでは、Setup 以外のアルゴリズムに対するシステムパラメータ param の入力は省略する．このとき，リング署名の安全性は以下となる．

偽造不可能性 偽造不可能性とは，攻撃者 $\mathcal{A}$ が一部の正直なユーザを適応的に corrupt し彼らの秘密鍵を取得できたとしても，corrupt していないユーザとしての有効な署名を生成することができないことを意味する．以下で定式化を行う．

任意の多項式時間攻撃者 $\mathcal{A}$ に対して，ある整数 q_k（セキュリティパラメータ λ に対して多項式な値）が存在し，以下の確率が無視可能であるとき**偽造不可能性**をもつ．

$$\Pr\left[\begin{array}{c} \mathsf{param} \leftarrow \mathsf{Setup}(\lambda) \text{ for } i \in [1, q_k]; \\ (\hat{\mathsf{pk}_i}, \hat{\mathsf{sk}_i}) \leftarrow \mathsf{KeyGen}(); S := \{\hat{\mathsf{pk}_i}\}_{i=1}^{q_k}; \\ (M^*, \mathbf{pk}^*, \sigma^*) \leftarrow \mathcal{A}^{\mathcal{CO}, \mathcal{SO}}(\mathsf{param}, S) : \\ 1 \leftarrow \mathsf{Verify}(M^*, \mathbf{pk}^*, \sigma^*) \wedge \\ \mathbf{pk}^* \subseteq S \setminus C, (M^*, \mathbf{pk}^*, \cdot) \text{ was not the input of } \mathcal{SO} \end{array}\right]$$

ただし，攻撃者 $\mathcal{A}$ に与えられるオラクルは以下のように定義される:

$\mathcal{CO}(i) : \hat{\mathsf{sk}_i}$ を出力し，集合 C その公開鍵 $\hat{\mathsf{pk}_i}$ を追加する．

$\mathcal{SO}(M, \mathbf{pk}, j)$：入力はメッセージ M，公開鍵のベクトル $\mathbf{pk}$，署名者のインデックス j である．出力は $\hat{\mathsf{pk}_j} \notin \mathbf{pk}$ の場合は $\perp$ であり，その他の場合は $\sigma \leftarrow \mathsf{Sign}(M, \mathbf{pk}, \hat{\mathsf{sk}_j})$ である．

匿名性 強匿名性モデルとは，攻撃者 $\mathcal{A}$ に全ユーザの秘密鍵を生成するための乱数が与えられても，任意の 2 ユーザのいずれの署名者かわからないことを意味する．以下で定式化を行う

任意の多項式時間攻撃者 $\mathcal{A}_1, \mathcal{A}_2$, に対して，ある整数 q_k（λ に対して多項式な値）が存在し，以下の確率が無視可能であるとき**匿名性**をもつ．

$$\left|\Pr\left[\begin{array}{c} \mathsf{param} \leftarrow \mathsf{Setup}(\lambda) \text{ for } i \in [1, q_k]; \\ (\hat{\mathsf{pk}_i}, \hat{\mathsf{sk}_i}) \leftarrow \mathsf{KeyGen}(\mathsf{param}; \omega_i); S := \{\hat{\mathsf{pk}_i}\}_{i=1}^{q_k}; \\ (M^*, \mathbf{pk}^*, i_0, i_1, St) \leftarrow \mathcal{A}_1^{\mathcal{SO}}(\mathsf{param}, S); \\ b \leftarrow_s \{0,1\}; \sigma^* \leftarrow \mathsf{Sign}(M^*, \mathbf{pk}^*, \mathsf{sk}_{i_b}); \\ b' \leftarrow \mathcal{A}_2(\sigma^*, \{\omega_i\}_{i=1}^{q_k}, St) : \\ b = b' \wedge \hat{\mathsf{pk}_{i_0}}, \hat{\mathsf{pk}_{i_1}} \in S \cap \mathbf{pk}^* \end{array}\right] - \frac{1}{2}\right|$$

ここでオラクル $\mathcal{SO}$ は偽造不可能性と同じであり，$\mathcal{A}_1$ によって選択されたリング $\mathbf{pk}^*$ には，攻撃者が生成した公開鍵を含めることができる．

4. 提案方式

4.1 提案方式の概要

Curve Tree [3] のシステムと 2.4.2 小節で述べた SPK を組み合わせることによって，効率的なリング署名を構築する．Curve Tree では，楕円曲線 $\mathbb{E}_{(\_)}$ のスカラー体の要素に対するコミットメント C_i を各葉ノードに割り当てている．本研究では，楕円曲線 $\mathbb{E}_{(\_)}$ のスカラー体のランダムな要素を秘密鍵とし，そのコミットメントを乱数を 0 として計算することで離散対数型の公開鍵を生成し，それを Curve Tree の各葉に割り当てる．これにより，公開鍵の集合の中に証明者の公開鍵が含まれていることを Curve Tree の select-and-rerandomize を用いることによって，効率的に証明することができる．また，Curve Tree の証明で使われた，署名者の rerandomize された公開鍵に対してその秘密鍵の知識の SPK を生成することでその公開鍵の所有者であることを証明でき，リング署名が構成できる．このとき，Curve Tree の証明サイズが $O(\log n)$，木構築なしの証明生成・検証時間が $O(\sqrt[\mathsf{D}]{n})$ であることから，構成したリング署名の署名長は $O(\log n)$ となり，公開鍵集合が変化せず木構築をしない場合の署名生成・検証時間も $O(\sqrt[\mathsf{D}]{n})$ となる．ただし，リングのサイズを n，Curve Tree の深さを D とする．

4.2 提案方式のアルゴリズム

提案方式のアルゴリズムは以下の通りである．Sign, Verify のサブアルゴリズムとして，Rerand，Accum，Schnorr.Sign，Schnorr.Verify を用いる．Curve Tree の葉ノードは楕円曲線 $\mathbb{E}_{(\mathrm{evn})}$ 上の点であるとする．

- $\mathsf{Setup}(1^\lambda) \to \mathsf{param}$
  セキュリティパラメータ λ に対して，2 サイクルの楕円曲線とそれらの生成元を生成し，公開パラメータ param を出力する．
  1) λ に対する 2 サイクルの楕円曲線 $(\mathbb{E}_{(\mathrm{evn})}, \mathbb{E}_{(\mathrm{odd})})$ を生成する．
  2) 1) の楕円曲線上の生成元 $\vec{G}^{(\mathrm{evn})} \in \mathbb{E}^n_{(\mathrm{evn})}, H^{(\mathrm{evn})} \in \mathbb{E}_{(\mathrm{evn})}, \vec{G}^{(\mathrm{odd})} \in \mathbb{E}^{n'}_{(\mathrm{odd})}, H^{(\mathrm{odd})} \in \mathbb{E}_{(\mathrm{odd})}$ を生成する．この n と n' はそれぞれの楕円曲線における constraints の数に依存する．
  3) $\mathsf{param} = (\vec{G}^{(\mathrm{evn})} \in \mathbb{E}^n_{(\mathrm{evn})}, H^{(\mathrm{evn})} \in \mathbb{E}_{(\mathrm{evn})}, \vec{G}^{(\mathrm{odd})} \in \mathbb{E}^{n'}_{(\mathrm{odd})}, H^{(\mathrm{odd})} \in \mathbb{E}_{(\mathrm{odd})})$ を出力する．
- $\mathsf{Keygen}(\mathsf{param}) \to (\mathsf{pk}, \mathsf{sk})$
  param を入力とし，一組の公開鍵 pk と秘密鍵 sk を生成する．
  1) 秘密鍵 $\mathsf{sk} \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{evn})}|}$ を生成し，その秘密鍵を用いて公開鍵 $\mathsf{pk} \leftarrow [\mathsf{sk}] \cdot G_1^{(\mathrm{evn})} \in \mathbb{E}_{(\mathrm{evn})}$ を計算する．

2) $(\mathsf{pk}, \mathsf{sk})$ を出力する．

- $\mathsf{Rerand}(\mathsf{param}, C \in \mathbb{E}_{(\mathrm{evn})}, r \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{evn})}|}) \to \hat{C}$
  あるノードのコミットメント C に対して rerandomize を行う．
  1) C に対する rerandomize として $\hat{C} \leftarrow C + [r] \cdot H^{(\mathrm{evn})}$ を計算する．
  2) $\hat{C}$ を出力する．
- $\mathsf{Accum}(\mathsf{param}, \mathbf{pk}, l) \to \mathsf{rt}$
  リング $\mathbf{pk}$ を入力として，リング内のすべての公開鍵を葉ノードとする木を 2.5.1 に基づき構築し，根ノード rt を出力する．
- $\mathsf{Schnorr.Sign}(\mathsf{param}, M, \hat{C}_{\mathrm{leaf}}, \mathsf{sk}_i, r_{\mathrm{leaf}}) \to (c, s_1, s_2)$
  乱数 r_{leaf} を用いて $\hat{C}_{\mathrm{leaf}}$ に rerandomize された公開鍵に対して，対応する秘密鍵 sk_i によるメッセージ M の SPK（c, s_1, s_2）を生成する．
  1) 二つの乱数 $r_1 \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{evn})}|}, r_2 \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{evn})}|}$ を生成する．
  2) $t \leftarrow [r_1] \cdot G_1^{(\mathrm{evn})} + [r_2] \cdot H^{(\mathrm{evn})}$ を計算する．この t が SPK におけるコミットメントとなる．
  3) $c \leftarrow H(G_1^{(\mathrm{evn})} || H^{(\mathrm{evn})} || \hat{C}_{\mathrm{leaf}} || t || M)$ を計算する．H はハッシュ関数であり，入力を結合してチャレンジ c を計算している．
  4) レスポンスとして $s_1 \leftarrow r_1 - c \cdot \mathsf{sk}_i, s_2 \leftarrow r_2 - c \cdot r_{\mathrm{leaf}}$ を計算する．
  5) 署名として（c, s_1, s_2）を出力する．
- $\mathsf{Schnorr.Verify}(\mathsf{param}, M, \hat{C}_{\mathrm{leaf}}, c, s_1, s_2) \to \{0,1\}$
  $\hat{C}_{\mathrm{leaf}}, M$ に対する SPK（c, s_1, s_2）の検証を行う．
  1) $t' \leftarrow [s_1] \cdot G_1^{(\mathrm{evn})} + [s_2] \cdot H^{(\mathrm{evn})} + [c] \cdot \hat{C}_{\mathrm{leaf}}$ を計算する．正しくチャレンジとレスポンスが計算されている場合，$t = t'$ となる．
  2) $c' \leftarrow H(G_1^{(\mathrm{evn})} || H^{(\mathrm{evn})} || \hat{C}_{\mathrm{leaf}} || t' || M)$ を計算する．
  3) c と c' が異なる場合は 0 を，正しい場合は 1 を出力する．
- $\mathsf{Sign}(\mathsf{param}, M, \mathbf{pk}, \mathsf{pk}_i, \mathsf{sk}_i) \to \sigma$
  署名されるメッセージ M，リング $\mathbf{pk}$，自身の公開鍵 pk_i，秘密鍵 sk_i を入力として，リング署名 σ を出力する．$n = |\mathbf{pk}|$ に対して構築する最適な木の深さを D，子ノードの数を l とする（Verify も同様）．
  1) $\mathsf{Accum}(\mathsf{param}, \mathbf{pk}, l)$ を実行することで木を構築し，根ノードを rt として受け取る．また，根ノードから葉ノードの一つである pk_i までのパスを辿り，その辿ったノードを $C^{(0)}, ..., C^{(\mathsf{D})}$ とする．ただし，$C^{(0)}$ は rt に対応しており，$C^{(\mathsf{D})}$ は pk_i とする．
  2) $\hat{C}^{(0)}$ に rt，$r^{(0)}$ に 0 を代入する．根ノードは rerandomize する必要がないためである．
  3) D が偶数の時，以下の処理を行う．
     k が 1 から $\frac{\mathsf{D}}{2}$ の間, 次の処理をループすることで，木の上層から順番に葉ノードまでの rerandomize を行う．
     I) $j \leftarrow 2k - 1, j' \leftarrow 2k$ を計算する．
     II) $r^{(j)} \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{odd})}|}, r^{(j')} \in \mathbb{F}_{|\mathbb{E}_{(\mathrm{evn})}|}$ を生成する．
     III) $\mathsf{Rerand}(\mathsf{param}, C^{(j)}, r^{(j)})$ を実行し，返り値を $\hat{C}^{(j)}$ に代入．
     IV) $\mathsf{Rerand}(\mathsf{param}, C^{(j')}, r^{(j')})$ を実行し，返り値を $\hat{C}^{(j')}$ に代入．
  4) D が奇数の時，3) と同様の処理を行う．
  5) 偶数層，奇数層それぞれに対して，Curve Tree の SelRerand.Prove を用いて証明 $\pi_{(\mathrm{evn})}, \pi_{(\mathrm{odd})}$ を生成する．
  6) $\mathsf{Schnorr.Sign}(\mathsf{param}, M, \hat{C}^{(\mathsf{D})}, \mathsf{sk}_i, r^{(\mathsf{D})})$ を用いて SPK (c, s_1, s_2) を生成する．
  7) $\sigma := (\hat{C}^{(1)}, ..., \hat{C}^{(\mathsf{D})}, \pi_{(\mathrm{evn})}, \pi_{(\mathrm{odd})}, c, s_1, s_2)$ を出力する．
- $\mathsf{Verify}(\mathsf{param}, M, \mathbf{pk}, \sigma) \to \{0,1\}$
  メッセージ M，リング $\mathbf{pk}$，M に対するリング署名 σ を入力として，リング署名の検証結果（0,1）を出力する．
  1) $\mathsf{Accum}(\mathsf{param}, \mathbf{pk}, l)$ を実行し，根ノードを rt として受け取る．ただし，今までに同じ $\mathbf{pk}$ において木を構築したことがある場合は省略できる．
  2) rt を $\hat{C}^{(0)}$ に代入する．
  3) 偶数層, 奇数層それぞれに対する証明 $\pi_{(\mathrm{evn})}, \pi_{(\mathrm{odd})}$ を Curve Tree の SelRerand.Vfy を用いて検証し，証明が正しい場合は 1，正しくない場合は 0 をそれぞれ $b_{(\mathrm{evn})}, b_{(\mathrm{odd})}$ に代入する．
  4) $\mathsf{Schnorr.Verify}(\mathsf{param}, M, \hat{C}^{(\mathsf{D})}, c, s_1, s_2)$ によって SPK (c, s_1, s_2) を検証し，正しい場合は 1，正しくない場合は 0 を b に代入する．
  5) $b_{(\mathrm{evn})} \wedge b_{(\mathrm{odd})} \wedge b$ を出力する．

4.3 提案方式の安全性

提案方式の安全性として，偽造不可能性と匿名性について述べる．

定理 1 提案方式は，ランダムオラクルモデルにおいて，一般化離散対数仮定の下で偽造不可能である．

以下二つの場合に分けて証明できる．

1) Sign で証明に用いた公開鍵（すなわち $\hat{C}^{(\mathsf{D})}$ に randomize された pk_i）がリング $\mathbf{pk}$ に入っている場合：
   この場合, 偽造不可能性の勝利条件が $\mathbf{pk} \subseteq S \backslash C$ であることから，pk_i が敵に corrupt されていないことを意味する．このとき, 離散対数問題として与えられた $[a] \cdot G$ を pk_i として $\mathcal{A}$ を動かし，$\mathcal{A}$ が偽造（$M^*, \mathbf{pk}^*, \sigma^* = (\hat{C}^{(1)^*}, ..., \hat{C}^{(D)^*}, \pi^*_{(evn)}, \pi^*_{(odd)}, c^*, s_1^*, s_2^*)$）を出力すると，SPK の special soundness より，$\mathcal{A}$ をリワインドして秘密鍵 $\mathsf{sk}_i = a$ を抽出し出力することで離散対数仮定を破ることができる．

2) 証明した公開鍵 pk_i がリング $\mathbf{pk}$ に入っていない場合：この場合，$\mathcal{A}$ が $(M^*, \mathbf{pk}^*, \sigma^* = (\hat{C}^{(1)^*}, ..., \hat{C}^{(D)^*}, \pi^*_{(evn)}, \pi^*_{(odd)}, c^*, s^*_1, s^*_2)$ を出力したときに，SPK の special soundness より，$\mathcal{A}$ をリワインドして秘密鍵 sk_i を抽出する．$\mathbf{pk}^*$ に対応した秘密鍵を $\mathbf{sk}^*$ とする．また，$\pi^* = (\hat{C}^{(1)^*}, ..., \hat{C}^{(\mathsf{D})^*}, \pi^*_{(evn)}, \pi^*_{(odd)})$ とする．このとき，$(\mathbf{sk}^*, \mathbf{0}, \mathsf{sk}_i, 0, \pi^*)$ を出力することで，Curve Trees の Select-and-Rerandomize Binding を破ることができる．

定理 2 提案方式は，ランダムオラクルモデルにおいて，匿名である．

元の匿名性定義での攻撃者との試行をゲーム 0 とし，それから以下を変更した試行をゲーム 1 とする．$\sigma^* \leftarrow \mathsf{Sign}(M^*, \mathbf{pk}^*, \mathsf{sk}_{i_b})$ の計算において，ランダムな値としてコミットメント $\hat{C}^1, ..., \hat{C}^{\mathsf{D}-1}$ を生成し，Bulletproofs の statistical zero-knowledge によりそれぞれの関係のシミュレータの出力 $\pi_{(\mathsf{evn})}, \pi_{(\mathsf{odd})}$ を得る．その後，SPK の special honest-verier zero-knowledge により $\hat{C}^{\mathsf{D}}$ に対するシミュレーターの出力 (c, s_1, s_2) を得る．そしてこのように生成した $\sigma^* = (\hat{C}^{(1)}, ..., \hat{C}^{(D)}, \pi_{(evn)}, \pi_{(odd)}, c, s_1, s_2)$ を $\mathcal{A}_2$ に渡す．ゲーム 1 では b と独立したランダムな σ^* であり，$b = b'$ となる確率は $\frac{1}{2}$ である．また，ゲーム 0 とゲーム 1 は，コミットメントの hiding，Bulletproofs および SPK の zero-knowledge 性により，識別することができない．よってゲーム 0 においても $b = b'$ の確率は $\frac{1}{2}$ となり，匿名性が成り立つ．

5. 実装と評価

提案方式の有効性を示すために提案方式を実装し処理時間を計測した．

5.1 実装および実験環境

本研究では，github で公開されている `https://github.com/simonkamp/curve-trees` の Curve Tree の実装を使用し，それを拡張して Rust で提案方式を実装した．この実装では pallas 曲線と vesta 曲線を用いた Pasta サイクルが用いられている．処理時間の測定は Intel（R） Core（TM） i5-10400 （2.90 GHz），16GB のメモリを搭載した Windows 11 Home 上で行っている．また，従来方式である DualRing は，楕円曲線として secp256r1（P256）使用し，提案方式と同様の環境で実装した．

5.2 実験結果と評価

以下の図 1 から図 5 において，実行時間やデータサイズの値がジャンプしているところは，2 サイクルのそれぞれの曲線において Curve Tree で利用している NIZK の bulletproofs で使用される constraints の数が 2 のべき乗を超えた場合である．これは，bulletproofs において再帰的に constraints の数を 2 で割って処理するために，constraints に padding を付けて 2 のべき乗となるようにしているためである．このことから，2^{n-1} から 2^n の範囲内に constraints の数がある場合には署名長や，処理時間はほぼ一定になる．また，Curve Tree を用いた提案方式でのリングのサイズは $n = l^{\mathsf{D}}$ となり，そのときの constraints の数は $\mathsf{D}(910 + l - 1)$ と計算される．以下，図はリングのサイズ n を 1 から 16384 まで増加させた場合の変化を表す．また，$\mathsf{D} = 2$ とした場合，n に関わらず，2 つの曲線の constraints は最小で 911 ずつとなる．このことから，$\mathsf{D} = 1$ のときに，vesta 曲線の constraints が 2048 を超えるときの n の値が 2048 であり，n が 16384 までの範囲では各曲線の constraints が 2048 を超えないため，$n \geq 2048$ では深さ D を 2 にしている．

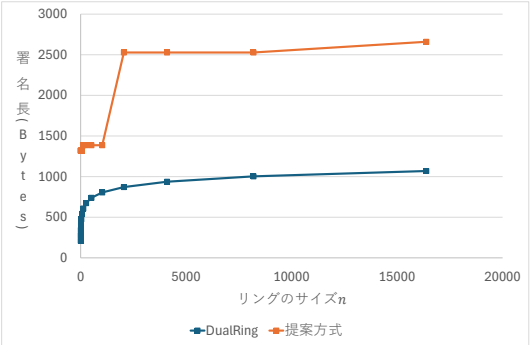

図 1 従来方式と提案方式のリングのサイズに対する署名長の比較

5.2.1 署名長の比較

図 1 は，従来方式と提案方式のリングのサイズ n に対する署名長の比較を示している．提案方式では，Curve Trees の証明データサイズにより $O(\log n)$ となっており，DualRing と同じオーダーである。具体的には DualRing の署名長の倍となっているが，$n > 2000$ で 2500Byte 程度で十分実用的である．

5.2.2 処理時間の比較

図 2 は，n に対する署名生成時間および署名検証時間の変化を示している．Accum では，効率的な multi scalar multiplication（msm）を使用しており，$O(n)$ であるものの高速に処理できる．$n \leq 16384$ では Sign の Accum 以外の処理時間が支配的なため，$O(\sqrt[6]{n})$ のオーダーとなっている．一方，Verify では Accum 以外の処理時間が小さいため、Accum の処理時間が支配的になる．図 3 では，n に対するリングが変化しない（Accum を行わない）場合の署名生成時間，署名検証時間の変化を示している．署名生成時間，署名検証時間ともに署名長と同様の場所でジャンプしており，階段状の形となっている．提案方式では SPK の生成と比較して Curve Tree の証明生成が支配的であることから，署名生成時間が $O(\sqrt[6]{n})$ となっている．こうして，DualRing [2] では $O(n)$ の署名生成時間であるのに対して，

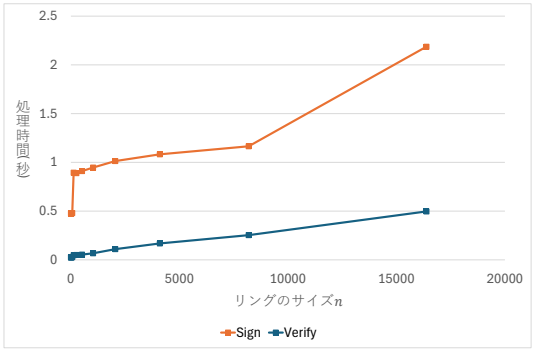

図 2 リングのサイズに対する署名生成時間および署名検証時間の変化

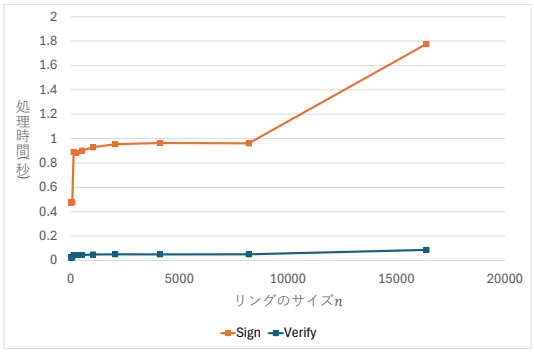

図 3 リングのサイズに対するリングが変化しない場合の署名生成時間，署名検証時間の変化

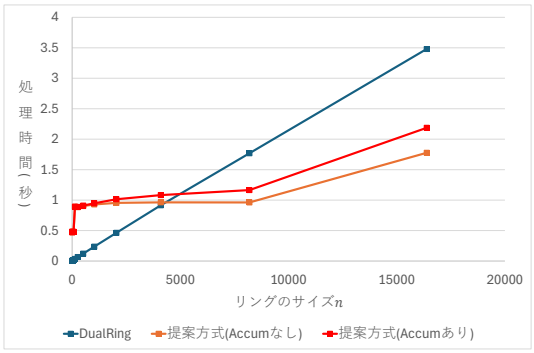

図 4 リングのサイズに対する従来方式と提案方式の署名生成時間の比較

sublinear な処理時間を達成している．また，署名検証時間の方が署名生成時間よりも 10 倍以上小さい．

図 4 は，n に対する従来方式と提案方式の署名生成（Accum あり，なし）時間の比較を示している．n が 4096 以下の場合は，DualRing の方が小さくなるが，それ以上の場合は，Accum あり，なしともに提案方式の方が小さくなっている．これは，DualRing の署名生成時間が $O(n)$ であるのに対して，提案方式では n が 16384 までは $O(\sqrt[p]{n})$ の処理時間であるためである．

図 5 は，n に対する従来方式と提案方式の署名検証（Accum あり，なし）時間の比較を示している．木の構築を含む場合は DualRing よりも大きいか同程度となるが，含まない場合は 2048 以下では DualRing の方が小さいものの，それ以

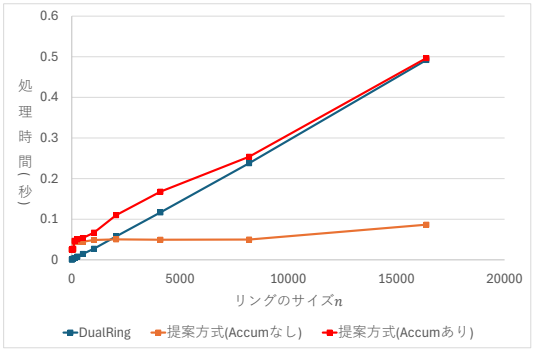

図 5 リングのサイズに対する従来方式と提案方式の署名検証時間の比較

上では提案方式の方が小さくなっている．これは，Accum を除く Verify の処理時間が $O(\sqrt[p]{n})$ であり，さらに Sign よりも高速なためである．

以上の結果から，Accum を含む場合は署名生成は $n > 4096$ で高速、検証は同程度となるものの，Accum を含まない場合は署名・検証ともに sublinear であり，署名生成が $n > 4096$，検証が $n > 2048$ において従来方式より高速である．Accum はリングが変化しない場合は計算する必要がなく，そのようなケースでは，従来方式と比較して十分な高速化が達成できている．

6. まとめ

本研究では，リング署名 [1] について，Curve Tree [3] と SPK を組み合わせた方式を提案した．この提案方式では，署名長を $O(\log n)$ に維持したまま，リングが変化しない場合において，署名生成・検証時間を sublinear 時間まで削減している．今後の課題は，木の更新方法の検討等が考えられる．

参考文献

[1] R. L. Rivest, A. Shamir, and Y. Tauman. How to leak a secret. In *ASIACRYPT 2001*, pages 552–565, 2001.

[2] T. H. Yuen, M. F. Esgin, J. K. Liu, M. H. Au, and Z. Ding. Dualring: Generic construction of ring signatures with efficient instantiations. In *CRYPTO 2021*, pages 251–281, 2021.

[3] M. Campanelli, M. Hall-Andersen, and S. H. Kamp. Curve trees: Practical and transparent zero-knowledge accumulators. In *32nd USENIX Security Symposium, USENIX Security 2023*, pages 4391–4408, 2023.

[4] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *CRYPTO '91*, pages 129–140, 1991.

[5] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *2018 IEEE Symposium on Security and Privacy, SP 2018*, pages 315–334, 2018.

[6] M. Campanelli, M. Hall-Andersen, and S. H. Kamp. Curve forests: Transparent zero-knowledge set membership with batching and strong security. *IACR Cryptol. ePrint Arch.* 1647, 2024 (published in FC 2025).