Curve Treeを用いた非中央集権型匿名属性認証における検証時間の削減

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

概要：匿名属性認証は、ユーザが匿名性を保ったまま自身の属性を示す認証である。従来の方式では特定の発行元に依存していたのに対して、コミットメントにより暗号化された属性情報をブロックチェーン上に登録し、それを用いたゼロ知識証明により認証を行う非中央集権型匿名属性認証が提案されている。DualRing-EC と呼ばれるリング署名に基づいた方式が提案されているが、コミットメント数 N に対し検証時間が線形で増加するため、ユーザ数が増えると問題となる。そこで本研究では、Curve Tree を用いた効率的な非中央集権型匿名属性認証を提案する。Curve Tree を用いることで、信頼できるセットアップ不要な公開パラメータを用いつつ、検証時間を DualRing-EC よりも削減する。さらに、提案方式を PC 上で実装し処理時間を測定することで、その有効性・実用性を確認する。

キーワード：非中央集権型匿名属性認証, Curve Tree, ゼロ知識証明, ブロックチェーン

Reducing Verification Time in Decentralized Anonymous Attribute-Based Credential System Using Curve Tree

Shoma Watanabe[1,a)]　Toru Nakanishi[1,b)]　Teruaki Kitasuka[1,c)]　Zhuotao Lian[1,d)]

Abstract: Anonymous attribute-based credential systems allow users to prove their attributes while maintaining anonymity. While the conventional systems depends on the trust of specific issuers, a decentralized anonymous attribute-based credential system was proposed, where attributes encrypted by commitments are registered on blockchain and zero-knowledge proofs are used for authentication. Subsequently, a system using a signature scheme called DualRing-EC was proposed, but it suffers from a linear increase in the verification time for the number of commitments N, which causes a problem in case of lots of users. Therefore, in this paper, we propose a decentralized anonymous attribute-based credential system using Curve Trees. By employing Curve Trees, public parameters do not require a trusted setup, and the verification time is reduced, compared to the DualRing-EC-based system. Furthermore, we confirm the effectiveness and practicality of the proposed system by implementing it on a PC and measuring its processing times.

Keywords: Decentralized anonymous attribute-based credential system, Curve tree, Zero-knowledge proof, Blockchain

1. はじめに

現在利用されている認証ではユーザ ID を用いるが, その ID をたどることでサーバは特定のユーザの行動を知ることができてしまう. ここにおける ID とは, ユーザを特定する情報であり, 身近なものではメールアドレスや学籍番号などがこれに当てはまる. そこで, ID をサーバに公開せ

[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)] m253663@hiroshima-u.ac.jp
[b)] t-nakanishi@hiroshima-u.ac.jp
[c)] kitasuka@hiroshima-u.ac.jp
[d)] lian-zhuotao@hiroshima-u.ac.jp

ずにプライバシーを保護しつつ認証できる匿名認証が研究されている.

匿名属性認証とは, 匿名認証の一種であり, ユーザがサービス提供者に対し匿名性を保ったまま自身の属性を示すことができる. ユーザは, 認証局に自身の属性証明書の発行を要求し, 認証局がユーザの属性証明書, すなわちクレデンシャルを発行する. しかし, この手法ではユーザと認証局が結託することで不正なクレデンシャルを作成し, ユーザが特定の属性を持っていないにもかかわらず, サービス提供者を騙し不正に認証を受けることができてしまう. それを避けるため, 発行元である認証局には信頼性が必要となる.

これに対し, 信頼できる発行元を必要としない非中央集権型匿名属性認証 [1] が提案されている. [1] では, ブロックチェーンを用いることで発行元なしで匿名属性認証を実現している. 各ユーザは自身の属性や秘密鍵をコミットメントとして一つの値に圧縮し, それをクレデンシャルとしてブロックチェーン上に登録する. その後, それを用いてゼロ知識証明を行い匿名属性認証を行う. しかし, この方式では RSA ベースの認証を行っているためにパラメータの設定に信頼できる第三者もしくは, それに代わる大規模な計算量 (セキュアマルチパーティ計算, MPC) が必要となってしまう.

この問題に対して, DualRing-EC を用いた非中央集権型匿名属性認証 [2] が提案されており, 楕円曲線群の生成元のみの公開パラメータを用いて方式が構築されているため, 信頼できる第三者を必要としない. DualRing-EC とは, リング署名の一種であり, 公開鍵が N 個ある場合に, その中に自身の所持する秘密鍵に対応した公開鍵が含まれていることを, 検証者にその公開鍵がどれであるかを知らせることなく示すことができる. これを [2] ではコミットメントに拡張することにより, 非中央集権型の匿名属性認証を実現している.

しかし, DualRing-EC の処理時間はコミットメント数を N とすると $O(N)$ で増加する. それにより, [2] の検証時間も $O(N)$ で増加してしまう. ユーザの匿名性は, ユーザのコミットメントを秘匿するブロックチェーン上のコミットメント集合の大きさ N に依存して強くなる. このため N に対する処理時間の増加を抑える必要がある. ブロックチェーンの利用を想定した場合, ブロックチェーンノードによる検証時間の削減は重要である.

そこで本研究では, Curve Tree を用いた非中央集権型匿名属性認証を提案する. Curve Tree [3] では, 木の葉ノードや内部ノードが楕円曲線上の点に対応しており, 楕円曲線を二つ用いて奇数層, 偶数層にそれぞれ対応させることにより, ある層のコミットメント集合を圧縮してもう一つの楕円曲線上の点に対応したコミットメントを計算することができる. こうしてマークル木と呼ばれる木構造を構成して, 多数のコミットメントを 1 つのルートのコミットメントに圧縮しつつ, 自分のコミットメントが木のコミットメント集合に含まれていることを効率的にゼロ知識証明できる. これを [2] の方式の DualRing-EC の代わりに用いることにより, 信頼できるセットアップを必要としない公開パラメータを用いつつ, コミットメント数 N に対する検証時間を DualRing-EC を用いた非中央集権型匿名属性認証よりも削減する. さらに, 提案方式を PC 上で実装して処理時間を測定することによりその有効性, 実用性を確認する.

2. 数学的準備

2.1 楕円曲線とその基本的な定義

$\mathbb{E}[\mathbb{F}_q] \subseteq \mathbb{F}_q \times \mathbb{F}_q$ は，有限体 $\mathbb{F}_q$ 上の楕円曲線 $\mathbb{E}$ に属する点 (X, Y) の集合を表す. 文脈から明らかである場合，$\mathbb{F}_q$ を省略し，単に $\mathbb{E}$ と記述する. 楕円曲線上の点はアーベル群 $(\mathbb{E}[\mathbb{F}_q], +)$ を生成し, 点同士の演算を加法記法により記述する. 楕円曲線上の群の位数 $p = |\mathbb{E}[\mathbb{F}_q]|$ は素数であり, この群は巡回群となる.

要素 $G \in \mathbb{E}[\mathbb{F}_q]$ に対する $s \in \mathbb{F}_p$ のスカラー倍は $[s] \cdot G$ により表される. スカラーのベクトル $\boldsymbol{s} \in \mathbb{F}_q^n$ と楕円曲線上の点のリスト $\boldsymbol{G} \in \mathbb{E}[\mathbb{F}_q]^n$ の内積は $\langle \boldsymbol{s}, \boldsymbol{G} \rangle = \sum_{i \in [n]} [s_i] \cdot G_i$ で表される.

2.2 安全性仮定

本研究では, Curve Tree の安全性のために以下の一般化 DL 仮定を用いる.

一般化 DL 仮定：$\mathcal{G}(1^\lambda)$ をセキュリティパラメータ λ に対して楕円曲線のパラメータ $(\mathbb{E}, \mathbb{F}_q, \mathbb{F}_p)$ を出力するアルゴリズムとする. このとき, 全ての確率的多項式時間 (PPT) アルゴリズム $\mathcal{A}$ において, 以下の確率は無視できる.

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

2.3 コミットメント

コミットメントは送信者が選択した値を受信者に対し秘匿しながらコミット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性質を満たす.

- **隠蔽性 (Hiding)**: コミットメントの入力に用いられた値を受信者は開示される前に知ることができない.
- **束縛性 (Binding)**: 送信者は送信したコミットメントの入力に用いた値を後から変更することができない. 開示する際, その値は入力に用いられた値ただその一つに定まる.

本研究においては, Pedersen Commitment [6] を使用し, ベクトル v のコミットメント c は以下のように定義される.

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

ここで, $\mathbb{F}_p = |\mathbb{E}[\mathbb{F}_q]|, G_1, ..., G_l, H \in \mathbb{E}[\mathbb{F}_q], \boldsymbol{v} \in \mathbb{F}_p^l$ であり, $r \in \mathbb{F}_p$ は乱数である.
また, Pedersen Commitment は再ランダム化可能であるという性質を持つ. すなわち, 新たな乱数 $\delta \xleftarrow{\$} \mathbb{F}_p$ を用いて $C^* \leftarrow C + [\delta] \cdot H$ を計算することで, 同じ $\boldsymbol{v}$ と乱数 $r + \delta$ に対するコミットメント C^* が得られる.

2.4 ゼロ知識証明

ゼロ知識証明とは, ある人が自分の持つある命題が真であることを, それ以外の知識を何も伝えずに証明する手法である. 本研究では, 知識の署名と Curve Tree で用いられている Bulletproof と呼ばれる汎用ゼロ知識証明 (zk-SNARK) を用いる. 知識の署名 (SPK: Signature based on Proof of Knowledge) は, 知識の証明 (PK: Proof of Knowledge) を Fiat-Shamire heuristic により変換することで得られる署名である. ディジタル署名では, 署名者から検証者への一方向性の通信 (非対話の通信) となるため, Non-Interactive Zero-Knowledge (NIZK) arguments と呼ばれる場合もある. 本研究では, 離散対数である秘密情報 x を知ることを示すメッセージ m に対する SPK を利用し, 以下のように記述する.

$$\mathsf{SPK}\{(x) : y = g^x\}(m)$$

Bulletproof [5] とは, R1CS と呼ばれる方式で算術回路を制約として表し, 任意の回路に基づいた NP 関係をゼロ知識証明として効率的に証明するものであり, セットアップに信頼できる第三者を必要としない. 本研究では, 楕円曲線 $\mathbb{E}$ 上における Bulletproof を用い $\mathsf{zk\text{-}BP}[\mathbb{E}]$ と表記する. $\mathsf{zk\text{-}BP}[\mathbb{E}]$ は $\mathbb{E}[\mathbb{F}_q]$ 上の R1CS 関係に対して効率的な証明システムであり, 一般化 DL 仮定の下, ランダム・オラクル・モデルにおいて計算量的健全性とゼロ知識性を持つ.

2.5 Curve Tree

Curve Tree [3] とは, 葉ノードおよび内部ノードが楕円曲線上の点から成る, 木構造を用いた効率的なメンバーシップ証明である. 葉ノードには各ユーザのコミットメントが対応しており, 楕円曲線を二つ用い, 各層で交互に曲線を切り替えることでそれらの圧縮を可能にし, 匿名性を保ったまま効率的にメンバーシップ証明を行うことができる. 木の深さ $D \in \mathbb{N}$, 木の分岐係数 $l \in \mathbb{N}$ である Curve Tree のアルゴリズムを以下に示す.

- $\mathsf{Setup}(1^\lambda) \to \mathsf{pp}$
  セキュリティパラメータ λ に対し, 公開パラメータ pp を生成する. これらのパラメータは透過的であり, 信頼できる第三者を必要としない.
- $\mathsf{Comm}(\mathsf{pp}, v_{leaf} \in \mathbb{F}_{|\mathbb{E}_{(\mathsf{evn})}|}, o \in \mathbb{F}_{|\mathbb{E}_{(\mathsf{evn})}|}) \to C$
  入力 v_{leaf} の乱数 o によるコミットメント C を出力する.
- $\mathsf{Rerand}(\mathsf{pp}, C \in \mathbb{E}_{(\mathsf{evn})}, r \in \mathbb{F}_{|\mathbb{E}_{(\mathsf{evn})}|}) \to \hat{C}$
  コミットメント C を乱数 r により再ランダム化し, $\hat{C}$ として出力する.
- $\mathsf{Accum}(\mathsf{pp}, S' = \{C_1, ..., C_n\}) \to \mathsf{rt}$
  コミットメント集合 S' から木を生成し, 木の根である rt を出力する.
  ここで, 木の深さは D, 木の分岐係数は l であり, 2-cycles の楕円曲線 $(\mathbb{E}_{(\mathsf{evn})}, \mathbb{E}_{(\mathsf{odd})}, \mathbb{F}_p, \mathbb{F}_q)$ で構成される. $\mathbb{E}^l_{(\mathsf{evn})}, \mathbb{E}^l_{(\mathsf{odd})}$ にはそれぞれ l 個の点の X, Y 座標の $2l$ 個の値が存在し, それぞれ l 個ずつ $\boldsymbol{G}^x_{(\mathsf{evn})}, \boldsymbol{G}^y_{(\mathsf{evn})}, \boldsymbol{G}^x_{(\mathsf{odd})}, \boldsymbol{G}^y_{(\mathsf{odd})}$ に分けられる. また, $\mathbb{E}_{(\mathsf{evn})}, \mathbb{E}_{(\mathsf{odd})}$ のアルゴリズム中における表記について, $\mathbb{E}_{(\_)}$ は現在の層に対応している楕円曲線を表し, $\mathbb{E}_{\mathsf{other}(\_)}$ はもう一方の楕円曲線を表す. 例えば, $\mathbb{E}_{(\_)} = \mathbb{E}_{(\mathsf{evn})}$ であれば $\mathbb{E}_{\mathsf{other}(\_)} = \mathbb{E}_{(\mathsf{odd})}$ である.
  葉ノード $(0, l, \mathbb{E}_{(\_)}, \mathbb{E}_{\mathsf{other}(\_)})$ に対する楕円曲線上の点はコミットメント $C \in \mathbb{E}_{(\_)}$ によって定まり, 親ノード $(D, l, \mathbb{E}_{(\_)}, \mathbb{E}_{\mathsf{other}(\_)})$ に対する楕円曲線上の点 C' は以下の計算により求められる. ここで, l 個の子ノード $(D-1, l, \mathbb{E}_{\mathsf{other}(\_)}, \mathbb{E}_{(\_)})$ のコミットメントを, $C_1 = (X_1, Y_1) \in \mathbb{E}_{\mathsf{other}(\_)}, ..., C_l = (X_l, Y_l) \in \mathbb{E}_{\mathsf{other}(\_)}$ とし, $\boldsymbol{X} = (X_1, ..., X_l), \boldsymbol{Y} = (Y_1, ..., Y_l)$ とする.

$$C' = \left\langle \boldsymbol{X}, \boldsymbol{G}^x_{(\_)} \right\rangle + \left\langle \boldsymbol{Y}, \boldsymbol{G}^y_{(\_)} \right\rangle$$

- $\mathsf{Prove}(\mathsf{pp}, S, C_{leaf}, r) \to \pi$
  $C_{leaf} \in S$ である証明を Bulletproof を用いて偶数層, 奇数層ごとに生成し, それらの集合 π を返す. 具体的な処理は 5.2 節の提案方式において示す.
- $\mathsf{Vfy}(\mathsf{pp}, \mathsf{rt}, \hat{C}_{leaf}, \pi) \to 0/1$
  π に対して, $\hat{C}_{leaf}$ が C_i を再ランダム化したコミットメントであり, C_i が rt を根とする木の葉ノードであることを偶数層, 奇数層ごとに検証し, すべて成功した場合 1 を出力し, そうでなければ 0 を出力する. 具体的な処理は 5.2 節の提案方式において示す.

Curve Tree の安全性を以下に示す.

- 束縛性 (Binding)
  任意の確率的多項式時間 (PPT) 攻撃者 $\mathcal{A}$ に対して, 以下の確率が無視可能である.

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

- ゼロ知識性
  あるシミュレータ $\mathcal{S}$ が存在し, 任意のセキュリティパラメータ $\lambda \in \mathbb{N}$, 任意の PPT 攻撃者 $\mathcal{A}$, 任意のインデックス $j^* \in [n]$ に対し、$p_0 = p_1$ が成り立つ。 ここで、p_0 および p_1 は次のように表される.

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

  ここで, 関数 X_b は b の値に応じて, それぞれ $X_0(\mathsf{pp}, S, C, \hat{C}, r) := \mathcal{S}(\mathsf{pp}, S, \hat{C})$ はシミュレータが出力する疑似的な証明, $X_1(\mathsf{pp}, S, C, \hat{C}, r) := \mathsf{Prove}(\mathsf{pp}, S, C, r)$ は実際の Prove アルゴリズムによる証明を生成する.

2.6 Curve Forests

Curve Forests [4] とは, Curve Tree [3] に対しバッチ処理を適用することで [3] と比較して証明時間, 検証時間をそれぞれ 2 倍, 3 倍高速化し, データサイズを 60% 削減している. [3] が単一のコミットメントのメンバーシップ証明を効率よく行えるのに対し, [4] は複数のコミットメントに対する証明に対応する. 本研究では高速化された Curve Forests の実装を利用する.

この方式では, 同じ要素集合に対して異なる生成元を用いて複数の Curve Tree を構築する. 証明においては, 複数の木にあるパスの各レベルにおけるノードを合算し, 一つの仮想的なノードとして扱うことで, 再ランダマイズに関する制約数を大幅に削減する. さらに, Curve Tree に存在した permissible point (許容点) に関する制約, すなわち, 特定の点のみを許容する処理を排除し, より単純かつ安全性を向上させたモデルを構築している.

許容点とは, あるユニバーサルハッシュ関数 $h(x, y)$ が特定の条件を満たすような点のことである. Curve Forests [4] では, x 座標を親にコミットする前に共通の既知の要素 Δ を足すことによって子をシフトし, 許容点のためのシフト操作を不要にしている.

3. 非中央集権型匿名属性認証の定義

3.1 アルゴリズムの定義

以下に非中央集権型匿名属性認証システム [1] の各アルゴリズムを示す.

- $\mathsf{Setup}(1^\lambda, n, l) \to (params)$
  セキュリティパラメータ λ とユーザの属性数 n, クレデンシャルの属性数 l に対し, 公開パラメータ $params$ を生成する.
- $\mathsf{KeyGen}(params) \to sk$
  ユーザの長期秘密鍵 sk を生成する.
- $\mathsf{FormNym}(params, sk) \to (Nym, sk_{Nym})$
  長期秘密鍵 sk のユーザの仮名 Nym, 仮名に対する秘密鍵 sk_{Nym} を生成する.
- $\mathsf{MintCred}(params, sk, Nym, sk_{Nym}, attrs, aux) \to (c, sk_c, \pi_M)$
  sk, Nym, sk_{Nym} とユーザの属性集合 $attrs = \{a_i\}_{i\in[n]}$, クレデンシャル発行の正当性を示す補助情報 aux に対し, クレデンシャル c, クレデンシャルに対する秘密鍵 sk_c, クレデンシャルの正しさを示す証明 π_M を生成する.
- $\mathsf{MintVerify}(params, c, attrs, Nym, aux, \pi_M) \to \{0, 1\}$
  クレデンシャル c の正当性を示す証明 π_M を検証し, 成功した場合に限り, クレデンシャルを台帳に載せることを許可する.
- $\mathsf{Show}(params, sk, Nym, sk_{Nym}, attrs, \boldsymbol{open}, c, sk_c, \boldsymbol{C}) \to \pi_S$
  sk, Nym, sk_{Nym}, と開示する属性のインデックス集合 $\boldsymbol{open} \subset [n]$ とユーザのクレデンシャル c, 台帳のクレデンシャルの集合 $\boldsymbol{C}$ に対し, クレデンシャル c に開示する属性集合 $\{a_i\}_{i\in\boldsymbol{open}}$ が含まれ, かつクレデンシャル c がクレデンシャル集合 $\boldsymbol{C}$ に含まれることを示す証明 π_1, クレデンシャル c と仮名 Nym がともに長期秘密鍵 sk のユーザのものであることを示す証明 π_2 からなる証明 π_S を生成する.
- $\mathsf{ShowVerify}(params, Nym, \pi_S, \{a_i\}_{i\in\boldsymbol{open}}, \boldsymbol{open}, \boldsymbol{C}) \to \{0, 1\}$
  $Nym, \{a_i\}_{i\in\boldsymbol{open}}, \boldsymbol{open}, \boldsymbol{C}$ に対する証明 π_S の正当性を検証する. 正しければ受理, そうでなければ拒否する.

3.2 システムモデルとプロトコル

本システムは, ユーザが属性を含むクレデンシャルを登録するプロトコルと, それを用いて匿名のまま認証を行うプロトコルから構成される. 事前に Setup により $params$ が生成されているとする.

(1) 登録フェーズ：ユーザは, KeyGen, FormNym, MintCred を実行し, 自身のクレデンシャルとそれに対する秘密鍵および仮名を生成し, ブロックチェーン上に仮名, クレデンシャルを, 属性およびクレデンシャルの正当性を示す補助情報とともに登録する. ブロックチェーン上の各ノードは MintVerify を実行し, 検証が成功すれば登録を許可し, そうでなければ登録を拒否する.

(2) 認証フェーズ：ユーザはサービス提供者に対する仮名を FormNym により生成する. そして Show アルゴリズムを実行し, 証明したい属性の集合を含むクレデンシャ

ルがブロックチェーン上に記録されており, かつ仮名を構成するユーザの長期秘密鍵, 仮名に対する秘密鍵, クレデンシャルに対する秘密鍵を知っていることを示す証明を生成し, サービス提供者に送信する. サービス提供者はブロックチェーンからクレデンシャル集合を取得し, ユーザが送信した証明を用いて ShowVerify を実行する. 検証が成功すればサービス提供者は認証を成功とし, そうでなければ失敗とする.

3.3 安全性

- **偽造不能性**
  クレデンシャル集合 $\boldsymbol{C}$ 中のいずれかのクレデンシャルに対する秘密鍵を知らないユーザ, もしくは自分のクレデンシャルに指定された属性を含まないユーザは ShowVerify に成功する証明を作ることができない.
- **匿名性**
  検証者は, 証明者がクレデンシャル集合中 $\boldsymbol{C}$ 中のどのクレデンシャルに対する秘密鍵に対応したユーザであるかわからない. また, 2 つの証明が同一の証明者によるものであるかもわからない.

4. 従来方式

4.1 従来方式の概要

[1] で提案されている非中央集権型匿名属性認証では, ブロックチェーン上に暗号化された属性情報のコミットメントを登録し, それを用いることで特定の認証局に依存せず匿名属性認証を行う. [1] の提案方式では, 各ユーザはブロックチェーン上に自身の属性情報のコミットメントを登録し, ゼロ知識証明によってブロックチェーン上のコミットメント集合中に自身のコミットメントが含まれていること, およびそのコミットメントが開示している属性情報を含んでいることをそれ以外の情報を秘匿して証明することで匿名属性認証を行う. しかし, この方式では, RSA ベースの認証を行っており, パラメータの設定に信頼できる第三者が必要となってしまう.

この問題を解決するために DualRing-EC を用いた非中央集権型匿名属性認証 [2] が提案されており, 楕円曲線群の生成元のみの公開パラメータを用いることで, 信頼できる第三者を必要としない. DualRing-EC は, リング署名の一種であり, 公開鍵が N 個ある場合に, それらを鍵束（リング）にまとめたものの中に自身の公開鍵が含まれていることを, 検証者にその公開鍵がどれであるかを知らせることなく $O(\log N)$ のデータサイズで示すことができる. [2] の方式では, コミットメントを公開鍵に変換し, 自身の公開鍵がリングに含まれることを証明している. これにより, コミットメントの包含関係をコミットメント数 N に対して $O(\log N)$ 証明サイズで証明している.

4.2 従来方式の問題点

DualRing-EC を用いた非中央集権型匿名属性認証 [2] では, DualRing-EC の処理時間がコミットメント数に比例するため, 検証時間が線形に増加してしまう. ブロックチェーンでの利用を考えた場合, コミットメント数, すなわちユーザ数の増加を見越したシステムにおいてはブロックチェーンノードの検証時間の増加を抑える必要がある.

5. 提案方式

5.1 提案方式の概要

提案方式では, DualRing の代わりに Curve Tree を用いて証明しているユーザのコミットメントがブロックチェーンに登録されたコミットメント集合に含まれていることを証明する. これにより, コミットメント数 N に対して証明データサイズを $O(\log N)$ に抑えつつ, 証明, 検証時間を $O(\sqrt[D]{N})$ に軽減する. （D は Curve Tree の深さ）. このため, 従来方式 [2] から Setup, Show, ShowVerify を変更する. 提案方式の Setup では, Curve Tree の 2-cycles の楕円曲線のパラメータを生成する. Show, ShowVerify では, [1] での DualRing による証明を Curve Tree [3] のアルゴリズム Accum と CurveTree.Prove および CurveTree.Vfy に変更し, 効率的なゼロ知識メンバーシップ証明により, 自身のコミットメントがブロックチェーン上のコミットメント集合に含まれていることを証明する. 以下に提案方式のアルゴリズムを示す.

5.2 アルゴリズム

- $\mathsf{Setup}(1^\lambda, n) \to (params)$
  (1) セキュリティパラメータ λ に対し, $p = |\mathbb{E}_{\mathrm{evn}}[\mathbb{F}_q]|$ and $q = |\mathbb{E}_{\mathrm{evn}}[\mathbb{F}_p]|$ となる 2-cycles の $\mathbb{E}_{\mathrm{evn}}, \mathbb{E}_{\mathrm{odd}}$ とその素体 $\mathbb{F}_p, \mathbb{F}_q$ を構成する.
  (2)$\mathbb{E}_{\mathrm{evn}}, \mathbb{E}_{\mathrm{odd}}$ をもとに,
  $G_0^{\mathrm{evn}}, G_1^{\mathrm{evn}}, \ldots, G_n^{\mathrm{evn}}, H_{\mathrm{evn}} \in \mathbb{E}_{\mathrm{evn}}$
  $G_0^{\mathrm{odd}}, G_1^{\mathrm{odd}}, \ldots, G_n^{\mathrm{odd}}, H_{\mathrm{odd}} \in \mathbb{E}_{\mathrm{odd}}$
  をランダムに選び, 公開パラメータ
  $params = (\boldsymbol{G}_{\mathrm{evn}} = (G_0^{\mathrm{evn}}, \ldots, G_n^{\mathrm{evn}}), \boldsymbol{G}_{\mathrm{odd}} = (G_0^{\mathrm{odd}}, \ldots, G_n^{\mathrm{odd}}), H_{\mathrm{evn}}, H_{\mathrm{odd}}, p, q)$ を出力する.

- $\mathsf{KeyGen}(params) \to sk$
  ユーザは自身の長期秘密鍵 $sk \in \mathbb{Z}_p$ をランダムに選び出力する.

- $\mathsf{FormNym}(params, sk) \to (Nym, sk_{Nym})$
  ユーザは仮名を生成する. まず, $r \in \mathbb{Z}_p$ をランダムに選び, 仮名に対する秘密鍵を $sk_{Nym} = r$ とし, 仮名 $Nym = [sk] \cdot G_0^{\mathrm{evn}} + [r] \cdot H^{\mathrm{evn}}$ を計算して (Nym, sk_{Nym}) を出力する.

- $\mathsf{MintCred}(params, sk, Nym, sk_{Nym}, attrs, aux) \rightarrow (c, sk_c, \pi_M)$
(1) ユーザの属性集合 $attrs$ に対するクレデンシャルとして Pedersen Commitment を以下のように生成する．まず，$r \in \mathbb{Z}_p$ をランダムに選び，$c = [sk] \cdot G_0^{\text{evn}} + [r] \cdot H_{\text{evn}} + \sum_{i \in [n]} [a_i] \cdot G_i^{\text{evn}}$ を計算する．クレデンシャルに対する秘密鍵を $sk_c = r$ とする．
(2) クレデンシャル c と仮名 Nym が同じ秘密鍵 sk を有するユーザにより生成されたことを示す証明

$$\pi_M = \mathsf{SPK}\{(sk, sk_{Nym}, sk_c) : \\ c = [sk] \cdot G_0^{\text{evn}} + [sk_c] \cdot H_{\text{evn}} + \textstyle\sum_{i \in [n]} [a_i] \cdot G_i^{\text{evn}} \\ \wedge Nym = [sk] \cdot H_{\text{evn}} + [sk_{Nym}] \cdot G_0^{\text{evn}}\}(aux)$$

を生成し，(c, sk_c, π_M) を出力する．

- $\mathsf{MintVerify}(params, c, attrs, Nym, aux, \pi_M) \rightarrow \{0, 1\}$
SPK π_M の正当性を検証する．検証に成功した場合 1 を，そうでなければ 0 を出力する．検証ノードは，このアルゴリズムが 1 を出力した場合に限り，クレデンシャル c を台帳に乗せることを許可する．

- $\mathsf{Show}(params, sk, Nym, sk_{Nym}, attrs, \boldsymbol{open}, c, sk_c, \boldsymbol{C}) \rightarrow \pi_S$
(1) クレデンシャル集合 $\boldsymbol{C}$ から Curve Tree の木を Accum を用いて構成し，根ノードを rt と表す．また，(sk, sk_{Nym}, sk_c) から生成された自身のクレデンシャルのコミットメントを c_{leaf} として，$c_0, \ldots, c_D$ を rt から c_{leaf} へのパスとする．ここで，c_0 は rt に対応し，c_D は c_{leaf} に対応する．
(2)$r_0 = 0$ とし，$i = 2, 4, \ldots$，$i' = 1, 3, \ldots$ として $r_i \in \mathbb{Z}_p, r'_i \in \mathbb{Z}_q$ をランダムに選び，$c_i, c_{i'}$ の再ランダム化されたコミットメント
$c'_i = c_i + [r_i] \cdot H_i$
$c'_{i'} = c_{i'} + [r_{i'}] \cdot H_{i'}$
を計算する．
(3)$c'_i, c'_{i'}$ がそれぞれ偶数の各層，奇数の各層の $c_i, c_{i'}$ の再ランダム化されたコミットメントであり，$c_i(c'_i)$ が $c_{i+1}(c_{i'+1})$ の親ノードであることを示す Curve Tree の証明
$\pi_{\text{evn}} \leftarrow \mathsf{zk\text{-}BP}[\mathbb{E}_{\text{evn}}].\mathsf{Prove}\big(\mathsf{pp}, \mathcal{R}^{\text{evn-levels}}, \mathsf{x}_{\text{evn}}, \mathsf{w}_{\text{evn}}\big)$
$\pi_{\text{odd}} \leftarrow \mathsf{zk\text{-}BP}[\mathbb{E}_{\text{odd}}].\mathsf{Prove}\big(\mathsf{pp}, \mathcal{R}^{\text{odd-levels}}, \mathsf{x}_{\text{odd}}, \mathsf{w}_{\text{odd}}\big)$
を生成し，$\pi_1 := (c'_1, \ldots, c'_D, \pi_{\text{evn}}, \pi_{\text{odd}})$ をセットする．
(4) ランダムな $nonce$ に対して，仮名 Nym が長期秘密鍵 sk を有するユーザに発行されたものであり，同一の sk と開示属性 $\{a_i\}_{i \in \boldsymbol{open}}$ を含むコミットメントの再ランダム化されたものが c'_D であることを示す証明 $\pi_2 := \mathsf{SPK}\{(sk, r', sk_{Nym}, \{a_i\}_{i \notin \boldsymbol{open}}) : c'_D - \sum_{i \in \boldsymbol{open}} [a_i] \cdot G_i^{\text{evn}} = [sk] \cdot G_0^{\text{evn}} + [r'] \cdot H_{\text{evn}} + \sum_{i \notin \boldsymbol{open}} [a_i] \cdot G_{i+1}^{\text{evn}} \wedge Nym = [sk_{Nym}] \cdot G_0^{\text{evn}} + [sk] \cdot G_1^{\text{evn}}\}$ を生成し，証明 $\pi_S := (\pi_1, \pi_2)$ を出力する．ここで $r' = r + r_{\text{evn}}$ である．

- $\mathsf{ShowVerify}(params, Nym, \pi_S, attrs, \boldsymbol{open}, \boldsymbol{C}) \rightarrow \{0, 1\}$
(1) クレデンシャル集合 $\boldsymbol{C}$ に対して Curve Tree の Accum を用いて木を生成し，根ノードを rt とする．
(2) π_1 について検証する．受け取った $c'_1, \ldots, c'_{D-1}$ に加え，$c'_0 = \mathsf{rt}$ とする．偶数層，奇数層ごとに
$b_{\text{evn}} = \mathsf{zk\text{-}BP}[\mathbb{E}_{\text{evn}}].\mathsf{VerProof}\big(\mathsf{pp}, \mathcal{R}^{\text{evn-levels}}, \mathsf{x}_{\text{evn}}, \pi_{\text{evn}}\big)$
$b_{\text{odd}} = \mathsf{zk\text{-}BP}[\mathbb{E}_{\text{odd}}].\mathsf{VerProof}\big(\mathsf{pp}, \mathcal{R}^{\text{odd-levels}}, \mathsf{x}_{\text{odd}}, \pi_{\text{odd}}\big)$
により各証明 $\pi_{\text{evn}}, \pi_{\text{odd}}$ を検証する．
(3)SPK π_2 について検証し，結果を b_2 とする．
$b_{\text{evn}} \wedge b_{\text{odd}} \wedge b_2 = 1$ であれば検証に成功したとして 1 を出力する．そうでなければ 0 を出力する．

5.3 安全性

安全性に関して，一般化 DL 仮定の下，Curve Trees の束縛性とゼロ知識性および SPK の健全性とゼロ知識性を前提とし，以下のように達成できている．

- **偽造不能性**
π_1 において，Curve Trees の束縛性より，集合 $\boldsymbol{c}$ に属したコミットメントでコミットされた属性集合に含まれない属性 a_i の証明に成功できない．また π_2 においては，SPK により Curve Trees で証明される自身のランダマイズされたコミットメント c'_D および偽名 Nym における秘密鍵 sk の知識を証明しているため，証明されているコミットメント（クレデンシャル）$c_{leaf} = c_D$ の秘密鍵を知らないなら証明に成功しない．よって，偽造不能性を満たす．
- **匿名性**
Curve Trees のゼロ知識性により，π_1 から検証者は追加の情報を得ることはない．また，π_2 においても，SPK のゼロ知識性のため情報を得ることはできない．こうして，匿名性を満たす．

6. 実装・実験結果

本研究では，**表 1** に示す実装環境において提案方式の実装，および処理時間の測定を行った．実装においては，Curve Forest により高速化された Curve Tree のソースコード[*1]を拡張して提案方式を実装した．従来方式 [2] につい

[*1] https://github.com/simonkamp/curve-trees

表 1　実装環境

OS	WSL2(Ubuntu 24.04.1 LTS)
CPU	Intel(R) Core(TM) i7-11700 2.50 GHz
メモリ	16.0GB
プログラミング言語	Rust
楕円曲線	pasta cycle

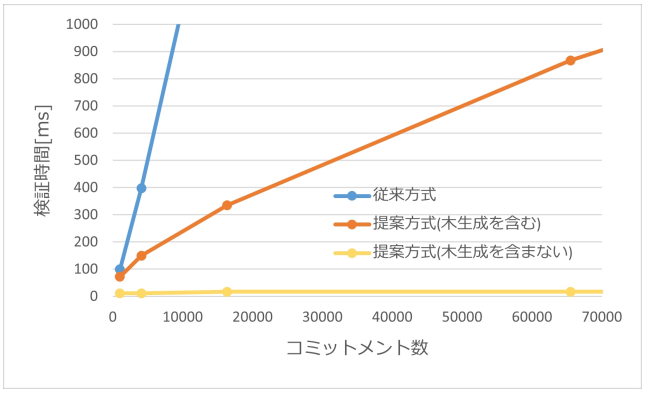

図 1　検証時間の比較

ては C 言語および OpenSSL を用いた実装を同一環境で計測した.

処理時間については, コミットメント数, 属性数を変化させ計測を行った. Curve Tree の木の深さ D は 2, 開示する属性数は 3 で固定している. コミットメント数 N に対して, 木の分岐係数を l とすると $N = l^D$ となる. 計測した範囲の N に対しては Curve Tree の処理時間が最適となるため $D = 2$ としている.

6.1　従来方式との検証時間の比較

図 1 はユーザの保持する属性数を 32 に固定し, コミットメント数を変化させたときの両方式の検証時間 (Curve Tree の木生成を含まない場合, 含む場合) を示す. 図 1 で示されているように, 提案方式の検証時間は $N = 60000$ でも $900ms$ 程度であり, [2] と比べて十分に高速化され, コミットメント数が増えるにつれてその差は大きくなっている. 木生成を含む場合は, 木生成の時間がコミットメント数に対してほぼ線形で増加するため合計時間もそのようになるが, グラフの傾きは [2] と比べて小さくなっており, 木生成を含まない場合と同様に, コミットメント数が増えると [2] との検証時間の差は大きくなる.

Curve Tree の証明, 検証時間について, 木の分岐係数を l, 深さを D とすると, Bulletproof で証明すべき制約の総数は $D \cdot (912 + l - 1)$ となる. D を 2 に固定するとその値は $\sqrt{N}$ に近似できる. 実際には, Bulletproof では再帰的な処理を行うために, $2^{n-1} < D \cdot (912 + l - 1) \leq 2^n$ の場合は一定の計算を行い, . 2^n を超えたとき, 2^{n+1} の制約数に対応した処理時間となる. そのため, 検証時間が線形で増加する [2] と比べて, 提案方式の木生成を含まない検証時間は $O(\sqrt{N})$ で階段状に増加することから, その差は大きくなっている.

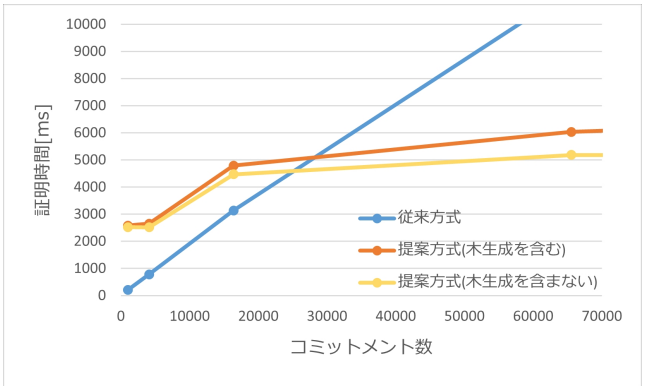

図 2　証明時間の比較

ブロックチェーンでの利用を考えた場合, ブロックチェーン上で検証済みの木を再利用することで検証の際に木を生成する必要がなくなるため, 検証時間の差は図 1 のように大きくなる.

6.2　従来方式との証明時間の比較

図 2 において, 6.1 節と同様に属性数を 32 に固定し, コミットメント数を変化させたときの両方式の証明時間 (木生成を含まない場合, 含む場合) を示す. 木生成を含まない証明時間についても制約数に比例するため, $D = 2$ において $O(\sqrt{N})$ に比例しつつ 2^n ごとに増加する階段状のグラフとなる. また木生成を含む場合は, 木生成の時間がほぼ N に比例しているが, N が 7000 程度までは証明時間が支配的であり, 証明時間のみの場合と同様のグラフとなっている. しかし, Bulletproof の証明コストが大きいため, コミットメント数がおよそ 30000 を超えるまで従来方式の方が速くなっている. 一方, コミットメント数が増えるにつれ証明時間は提案方式の方が速くなり, その差は大きくなっている.

6.3　制約数の増加による検証時間への影響

図 3 は同様の条件下で, ゼロ知識証明の Bulletproof で証明すべき回路における制約の総数とその検証時間への影響を比較したものである. 前述の通り, 制約数を表す青のグラフが 2^n を超えたとき, 検証時間を表すオレンジのグラフが大きくなっていることが読み取れる.

6.4　属性数に対する SPK の処理時間

図 4 において, 木の分岐係数 l を 256, すなわちコミットメント数 N を 256^2 に固定し, 属性数を変化させたときのゼロ知識証明 SPK の処理時間の変化を示している. 証明および検証にかかる時間は属性数に対し線形で大きくなっているが, Curve Tree の証明, 検証時間が 1000ms 単位であるのに対し SPK は 1ms 程度であるため, 属性数について, 現実的な数十の範囲では実行時間に影響はないといえる.

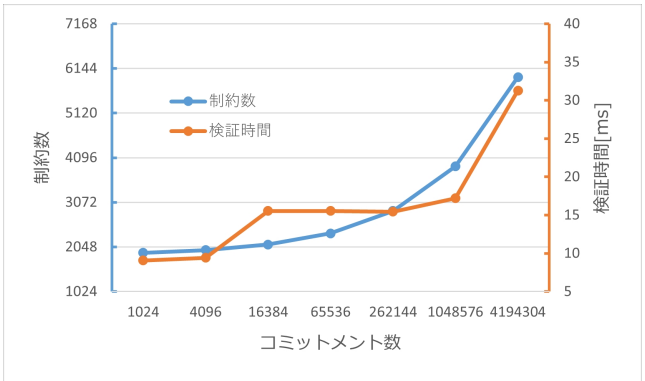

図 3 制約数の増加と検証時間

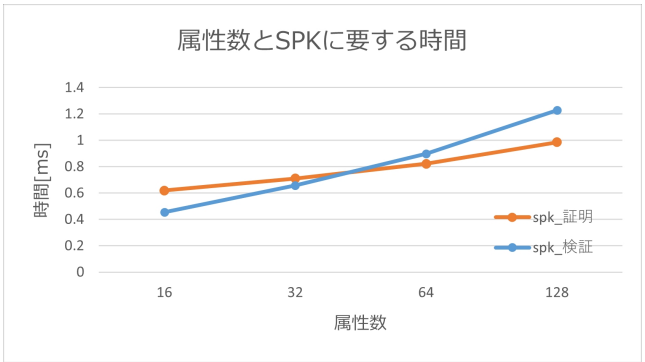

図 4 属性数に対する SPK の処理時間

7. まとめ

本研究では, Curve Tree を用いることにより検証時間を削減した非中央集権型の匿名属性認証方式を提案した. また, 提案した方式を PC 上で実装し, 従来方式 [2] と比べて検証時間が削減されていること, さらにコミットメント数が多ければ証明時間についても [2] と比べて高速化できていることを確認した. 今後の課題としては, 木生成の時間がコミットメント数に対して線形に大きくなり, 実行時間への影響が大きいため, この部分の改善が挙げられる.

参考文献

[1] C. Garman, M. Green, and I. Miers: Decentralized Anonymous Credentials, NDSS 2014 (2014).

[2] T. Nakanishi, M. Tani, and T. Kitasuka: Decentralized Attribute-Based Credentials with Short Attribute Proofs from DualRing, CANDAR 2024, pp. 203-209 (2024).

[3] M. Campanelli, M. Hall-Andersen, and S. H. Kamp: Curve Trees: Practical and Transparent Zero-Knowledge Accumulators, USENIX Security 2023, pp. 4391-4408 (2023).

[4] M. Campanelli, M. Hall-Andersen, and S. H. Kamp: Curve Forests: Transparent Zero-Knowledge Set Membership with Batching and Strong Security, IACR ePrint 2024/1647 (2024) (published in FC 2025).

[5] B. Bünz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille, and G. Maxwell: Bulletproofs: Short Proofs for Confidential Transactions and More, 2018 IEEE Symposium on Security and Privacy, pp. 315-334 (2018).

[6] T. Pedersen: Non-interactive and Information-theoretic Secure Verifiable Secret Sharing: CRYPTO’91, pp.129-140, (1991).