Vol.2018-CSEC-83 No.16
2018/12/14

線型準同型署名を用いた管理者に対して秘匿性を持つ評価システムの改善

上野山 大貴[1,a)]　中西 透[1,b)]

概要：ユーザの信頼度を測るために通販やネットオークションなどで使われる評価システムには、ユーザのプライバシー保護のため評価の管理者に対して秘匿性が求められる。従来、線型準同型性を持つ署名方式の用いた管理者に対して秘匿性を持つ評価システムが提案されている。本研究ではこの方式に対して、評価ユーザの増加による公開鍵データサイズの増大を低減する手法を導入する。また、ゼロ知識証明を用いることにより評価ユーザの不正評価を防止できるように改善する。そして PC 上での実装を行い、公開鍵データサイズ及び処理時間を計測し従来法と比較することでその有用性を示す。

キーワード：評価システム、準同型暗号、線型準同型署名、ゼロ知識証明

Improvements of Reputation System with Confidentiality to Administrator Using Linear Homomorphic Signatures

Daiki Uenoyama[1,a)]　Toru Nakanishi[1,b)]

1. はじめに

ネットオークションなどのユーザ間サービスではプライバシー保護のために匿名で行われることが多く、不正ユーザの判別がつかずトラブルが生じてしまう。その対策として、匿名ユーザの信頼度を判別するために評価システムが利用されている。評価システムでは、サービスを利用するユーザが互いに他のユーザと評価点を付け合う。そして、その評価点をシステム管理者が集計し、各ユーザの総評価点を算出する。その総評価点をユーザの ID と共に公開することで、これからサービスを利用するユーザは相手となるユーザの信頼度を検討することができる。

しかし、現在の評価システムではシステム管理者が一括してユーザの評価点を管理しているため、あるユーザが別のユーザにどのような評価点をつけたのかという情報は全てシステム管理者に把握されてしまう。

[1] 広島大学
Hiroshima Uniersity
[a)] m175529@hiroshima-u.ac.jp
[b)] t-nakanishi@hiroshima-u.ac.jp

このようなプライバシー問題の解決策として、システム管理者に対して秘匿性を持つ評価システムが提案されている [6]。この従来方式では、評価ユーザが決定した評価点を被評価ユーザのみが復号可能な形で暗号化し、その暗号文を使った各ユーザの総評価点の算出を被評価ユーザ自身によって行うことで秘匿性を満たしている。このことに伴って、被評価ユーザが自身で算出した総評価点を偽造する恐れがあるが、管理者が暗号文を保証した証明書を発行し、被評価ユーザが算出した総評価点に対し証明書を使ってゼロ知識証明することにより、自身の総評価点の正しさを他のユーザに示すことで総評価点の偽造を防止している。さらに、準同型暗号と線型準同型署名を用いることにより、評価数に依存しない効率的な証明を実現している。

しかし、この従来法では、暗号文を復号できないシステム管理者がその中身である評価点に関する正当性を吟味することなく証明書を発行しなければならないことが問題となる。また、総評価点の計算を保証するために必要となる検査ベクトルの要素数が評価ユーザの数に比例して増大することに依存して、証明書の署名に必要な公開鍵のデータ

サイズも同様に増大する。このことから、評価ユーザが莫大に増加するシステムでは実用性を損ねる恐れがあるとい問題もある。

そこで本研究では、前者の問題に対しては、システム管理者が想定される全ての評価点に対して予め証明書を発行し、評価ユーザの作成した暗号文に対してその評価点の証明書を使ったゼロ知識証明を行うことで、評価点の正当性を検証することを可能にする。後者の問題に対しては、新しい検査ベクトルの一つの要素が従来の検査ベクトル複数の要素の働きをするように改良を行うことで検査ベクトルの要素数の増大を抑え、これによって公開鍵のデータサイズの増大も低減させる。

第 2 章では提案方式に必要な数学的準備を示す。第 3 章では評価システムのモデルという観点から従来方式と提案方式を比較し、提案方式が満たすべき安全性について述べる。第 4 章で提案方式のアルゴリズムを示す。第 5 章は実装した提案方式を用いた実験結果を示し、第 6 章でまとめとともに今後の課題について述べる。

2. 数学的準備

2.1 双線形写像

本研究では、双線形写像が構成する楕円曲線上の群を利用している。共に位数が素数 p である 2 つの巡回群 $\mathbb{G}$,$\mathbb{G}_T$ に対して写像 $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ は以下の性質を満たすとき、双線形写像と呼ぶ。

- 双線形性:$\forall u, u_1, u_2, v, v_1, v_2 \in \mathbb{G}$ について、

$$e(u, v_1 \cdot v_2) = e(u, v_1) \cdot e(u, v_2)$$
$$e(u_1 \cdot u_2, v) = e(u_1, v) \cdot e(u_2, v)$$

- 非退化性:$\mathbb{G}$ の生成元 $g, \mathbb{G}_T$ の単位元 $1_{\mathbb{G}_T}$ について、

$$e(g, g) \neq 1_{\mathbb{G}_T}$$

また、双線形性から $\forall u, v \in \mathbb{G}, \forall a, b \in \mathbb{Z}$ について、以下の式も成立する。

$$e(u^a, v^b) = e(u, v)^{ab}$$

2.2 Desion Linear(DLIN) 仮定

[1] において、DDH 仮定を発展させた DLIN 仮定が提唱されている。DLIN 仮定は 2.4 節で示す BBS 暗号方式の解読不可能性を保証するものである。以下にこれを示す。

- 巡回群 $\mathbb{G}$ において、与えられた 2 つのベクトル $(g^a, g^b, g^{ac}, g^{bd}g^{c+d}), (g^a, g^b, g^{ac}, g^{bd}, g^z)$ は計算量的に識別困難である。

2.3 Simultaneous Double Paring (SDP) 仮定

本研究で用いる準同型署名では、その安全性が SDP 仮定 [3] に基づく。以下にこれを示す。

- 双線形写像 $(\mathbb{G}, \mathbb{G}_T)$ において、任意の要素の組 (g_z, g_r, h_z, h_u) に対して、次の 2 つの式を満たす (z, r, u) を求めるのが困難である。ただし、z, r, u はいずれも $1_{\mathbb{G}}$ ではない。

$$e(g_z, z) \cdot e(g_r, r) = 1_{\mathbb{G}_T}$$
$$e(z_z, z) \cdot e(h_u, u) = 1_{\mathbb{G}_T}$$

2.4 Boneh-Boyen-Shacham(BBS) 暗号方式

[1] において、双線形写像での BBS 暗号方式が提案されている。この暗号方式は与えられた平文 m_i を v^{m_j} に変換して暗号化した暗号文 $c(m_i)$ は次の準同型性と呼ばれる性質をを持つ。

$$c(m_1) \cdot c(m_2) = c(m_1 + m_2)$$

この性質から $c(m_1)$ と $c(m_2)$ を復号して m_1 と m_2 を求めることなく平文の和の暗号文 $c(m_1 + m_2)$ が得られることがわかる。本研究ではこのことを利用する。以下にこの方式を示す。

(1) **BBS-Keygen($\mathbb{G}$)**: 秘密鍵 sk_{BBS} と公開鍵 pk_{BBS} を以下のように生成する。

$$sk_{BBS} = (x, y) \xleftarrow{R} \mathbb{Z}_p^2$$
$$pk_{BBS} = (u, v, f, h)$$
$$\text{ただし、} u, v \xleftarrow{R} \mathbb{G}, f = u^x, h = u^y$$

(2) **Encrypt(m, pk_{BBS})**: 平文 $m \in \mathbb{Z}_p$ に対し以下のように暗号文 c を作成する。

$$c = (f^r, h^t, v^m \cdot u^{(r+t)})$$
$$(r, t \xleftarrow{R} \mathbb{Z}_p)$$

(3) **Decrypt(sk_{BBS}, c)**: c に対し sk_{BBS} を用いて以下のように計算する。

$$\frac{v^m \cdot u^{(r+t)}}{(f^r)^{\frac{1}{x}} \cdot (h^t)^{\frac{1}{y}}} = \frac{v^m \cdot u^{(r+t)}}{u^r \cdot u^t} = \frac{v^m \cdot u^{(r+t)}}{u^{(r+t)}} = v^m$$

この求められた v^m に対して離散対数問題を解くことで m を求める。

2.5 AHO 署名方式

[2] において、双線形写像で複数の群要素のメッセージ $\{M_i\}_{i=1}^n$ に対して署名可能な方式が提案されている。また、本研究ではメッセージは 1 個で固定され、そのメッセージに関する署名検証がゼロ知識証明可能であることを利用する。以下に署名検証に至るまでのこの方式を示す。

(1) **AHO-Keygen**($\mathbb{G}, n(=1)$): 秘密鍵 sk_{AHO} と公開鍵 pk_{AHO} を $n(=1)$ 個のメッセージベクトルを想定して生成する。

- $G_r, H_r \xleftarrow{R} \mathbb{G}$ と $\gamma_z, \delta_z \xleftarrow{R} \mathbb{Z}_p$ および $\gamma, \delta \xleftarrow{R} \mathbb{Z}_p$ を用意し、以下のように定義する。

$$G_z = G_r^{\gamma_z}, H_z = H_r^{\delta_z}, G = G_r^{\gamma}, H = H_r^{\delta}$$

- $\alpha, \beta \xleftarrow{R} \mathbb{Z}_p$ を用意し、以下のように定義する。

$$A = e(G_r, g^{\alpha}), B = e(H_r, g^{\beta})$$

- 以下のように秘密鍵 sk_{AHO} と公開鍵 pk_{AHO} を定義する。

$$sk_{AHO} = (\alpha, \beta, \gamma_z, \delta_z, \gamma, \delta)$$
$$pk_{AHO} = (G_r, H_r, G_z, H_z, G, H, A, B)$$

(2) **AHO-Sign**(sk_{AHO}, M): メッセージ M に対して署名 $\vec{\sigma}$ を作成する。

- $\zeta, \rho, \tau, \varphi, \omega \xleftarrow{R} \mathbb{Z}_p$ を用意し、$\theta_1, \cdots, \theta_7$ を以下のように計算する。

$$\theta_1 = g^{\zeta}$$
$$\theta_2 = g^{\rho - \gamma_z \zeta} \cdot (\tilde{g}^{s_i})^{-\gamma}$$
$$\theta_3 = G_r^{\tau}$$
$$\theta_4 = g^{(\alpha - \rho)/\tau}$$
$$\theta_5 = g^{\varphi - \delta_z \zeta} \cdot (\tilde{g}^{s_i})^{-\delta}$$
$$\theta_6 = H_r^{\omega}$$
$$\theta_7 = g^{(\beta - \varphi)/\omega}$$

- $\vec{\sigma} = (\theta_1, \cdots, \theta_7)$ とする。

(3) AHO-Verify($pk_{AHO}, \vec{\sigma}, M$): 以下により $\vec{\sigma}$ が M の署名であることが検証される。

- 以下の 2 つの検証式の成立を確認する。

$$A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot e(G, M)$$
$$B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot e(H, M)$$

2.6 Linear Homomorophic(LH) 署名方式

[3] において、双線形写像での準同型 (LH) 署名方式が提案されている。この署名方式は与えられたメッセージ M_i に対して作成した署名 $\sigma(M_i)$ は次の準同型性と呼ばれる性質を持つ。

$$\sigma(M_1) \cdot \sigma(M_2) = \sigma(M_1 + M_2)$$
$$a \in \mathbb{Z}_p \text{に対して、} a \cdot \sigma(M) = \sigma(a \cdot M)$$

本研究では、この性質を利用して n 個のメッセージベクトル $\{\vec{M}_i\}_{i=1}^n$ の署名 $\{\vec{\sigma_i}\}_{i=1}^n$ を乗算することで、$\sum_{i=1}^n \vec{M}_i (= \vec{M}$ とする) の署名 $\vec{\sigma}$ を作成する。以下にこの方式を示す。

(1) **LH-Keygen**($\mathbb{G}, n, \tau$): 秘密鍵 sk_{LH} と公開鍵 pk_{LH} を、ユーザ数 n と Lbit の ID である τ に基づいて生成する。

- $h \xleftarrow{R} \mathbb{G}$ と $\alpha_z, \alpha_r, \beta_z \xleftarrow{R} \mathbb{Z}_p$ を用意し、以下のように定義する。

$$g_z = h^{\alpha_z}, g_r = h^{\alpha_r}, h_z = h^{\beta_z}$$

- $i = 1, \cdots, n$ に対し、$\chi_i, \gamma_i, \delta_i \xleftarrow{R} \mathbb{Z}_p$ を用意し、以下のように計算する。

$$g_i = g_z^{\chi_i} g_r^{\gamma_i}, h_i = h_z^{\chi_i} h^{\delta_i}$$

- $\overline{\mathbf{w}} = (w_0, w_1, \cdots, w_L) \xleftarrow{R} \mathbb{G}^{L+1}$ を用意し、$H_{\mathbb{G}}(\tau) = w_0 \cdot \prod_{k=1}^{L} w_k^{\tau[k]}$ とすることでハッシュ関数 $H_{\mathbb{G}} : \{0,1\} \to \mathbb{G}$ を定義する。(ただし、$\tau[k]$ は Lbit の τ の左から k ビット目の値、すなわち 0 か 1)
- 以下のように秘密鍵 sk_{LH}, 公開鍵 pk_{LH} を定義する。

$$sk_{LH} = (h_z^{\alpha_r}, \{\chi_i, \gamma_i, \delta_i\}_{i=1}^n)$$
$$pk_{LH} = (g_z, g_r, h_z, h, \{g_i, h_i\}_{i=1}^n, \overline{\mathbf{w}})$$

(2) **LH-Sign**($sk_{LH}, \tau, \vec{M_i}$): メッセージベクトル $\vec{M_i}$ に対して署名 $\vec{\sigma_i}$ を作成する。ただし、$\vec{M_i}$ の j 番目の要素を m_{ij} とする。

- $\theta, \rho \xleftarrow{R} \mathbb{Z}_p$ を用意し、以下を計算する。

$$z_i = g_r^{\theta} \cdot \prod_{j=1}^{n} {m_{ij}}^{-\chi_i}$$
$$q_i = g_z^{-\theta} \cdot \prod_{j=1}^{n} {m_{ij}}^{-\gamma_i}$$
$$u_i = (h_z^{\alpha_r})^{-\theta} \cdot \prod_{j=1}^{n} {m_{ij}}^{-\delta_i} \cdot H_{\mathbb{G}}(\tau)^{-\rho}$$
$$v_i = h^{\rho}$$

- $\vec{\sigma_i} = (z_i, q_i, u_i, v_i)$ とする。

(3) **LH-SignDerive**($pk_{LH}, \tau, \{\vec{\sigma_i}\}_{i=1}^n$): n 個の署名 $\vec{\sigma_i}$ に対して、以下により $\vec{\sigma}$ を生成する。

- $\rho' \xleftarrow{R} \mathbb{Z}_p$ を用意し以下を計算する。

$$z = \prod_{j=1}^{n} z_j$$
$$q = \prod_{j=1}^{n} r_j$$
$$u = \prod_{j=1}^{n} u_j \cdot H_{\mathbb{G}}(\tau)^{-\rho'}$$
$$v = \prod_{j=1}^{n} v_j \cdot h^{\rho'}$$

- $\vec{\sigma} = (z, q, u, v)$ とする。

(4) **LH-Verify**$(pk_{LH}, \vec{\sigma}, \vec{M})$: 以下により $\vec{\sigma}$ が $\vec{M}$ の署名であることが検証される。ただし、$\vec{M}$ の j 番目の要素を m_j とする。

- 次の 2 つの検証式の成立を確認する。

$$1_{\mathbb{G}_T} = e(g_z, z) \cdot e(g_q, q) \cdot \prod_{j=1}^{n} e(g_j, m_j)$$

$$1_{\mathbb{G}_T} = e(h_z, z) \cdot e(h, u) \cdot e(H_{\mathbb{G}}(\tau), v) \cdot \prod_{j=1}^{n} e(h_j, m_j)$$

2.7 Groth-Sahai(GS) 証明

[4] において双線形写像における検証式を非対話でゼロ知識証明する方式が提案されている。GS 証明は秘密情報のコミット、証明生成、検証の 3 つの段階から構成されている。秘密情報のコミットでは、秘密情報の数に比例した計算時間を要し、証明生成では検証式に必要な証明の数に比例した計算時間を要する。また、検証では検証式の数に比例した計算時間を要する。

3. 評価システムのモデルと安全性の定義

3.1 評価システムのモデル

評価システムでは、評価ユーザ, 被評価ユーザの二者に加えて評価システムの管理サーバが参加する。評価ユーザは n 人いるものとして $U_i (i = 1, \cdots, n)$ と表し、被評価ユーザは R、管理サーバは M と表す。U_i と R は、M を介して評価を行う。また、三者は次のような振る舞いをするものとする。

- U_i は必ずしも正常な評価点を設定するとは限らない。
- M は評価点の具体的な値を知りたがっているが、評価点そのものに改ざんなどの不正は行わない。
- R は自身の総評価点の算出の際に不正を行いたい。

3.2 従来方式の問題点と提案方式の指針

まず、従来方式における評価システムの各手続きは以下である。

Setup R と M のアルゴリズムである。R は BBS 暗号に必要な公開鍵, 秘密鍵を生成する。また、M は LH 署名に必要な公開鍵, 秘密鍵を生成する。さらに、いずれの秘密鍵も作成者のみが所持し、公開鍵は他ユーザ全てに公開されるものとする。

Vote U_i から M へ、M から R へのプロトコルである。U_i は R の評価点を設定して暗号化し、その暗号文を M へ送信する。M は各 U_i に対応する検査ベクトルを暗号文に組み込んだ後、それらの証明書を発行する。最後に M は暗号文及び証明書を R へ送信する。

Calculate R のアルゴリズムである。R は受け取った複数の暗号文から総評価点の暗号文を作成し、総評価点を復号する。次に R は複数の証明書から総評価点に対応する証明書を作成する。最後に自身の総評価点およびそれに対応する証明書の非対話型ゼロ知識証明を他ユーザに公開する。

Show R の総評価点を確認したい任意のユーザが行えるアルゴリズムである。任意のユーザは、R の公開した総評価点に対するゼロ知識証明を用いて、総評価点が各評価点の加算になっていること、および評価点の重複が含まれていないことを確認する。

前節で U_i の振る舞いとして挙げたように、U_i の設定した評価点は常に正しい値であるとは限らないことから、**Vote** において暗号文を復号することなく評価点に関する正当性を吟味するアルゴリズムを加える必要がある。本研究では、この手続きを AHO 署名方式とゼロ知識証明を用いることで、評価点に関する正当性を M が検証できるように改善する。

このことにより提案方式では **Setup** および **Vote** の手続きを以下のように定義する。

Setup R と M のアルゴリズムである。R は BBS 暗号に必要な公開鍵, 秘密鍵を生成する。また、M は AHO 署名および LH 署名に必要な公開鍵, 秘密鍵を生成する。さらに、いずれの秘密鍵も作成者のみが所持し、公開鍵は他ユーザ全てに公開されるものとする。加えて、M は AHO 署名によって想定しうる全評価値に対する証明書を発行し公開する。

Vote U_i から M へ、M から R へのプロトコルである。U_i は R の評価点を設定して暗号化し、その暗号文を M へ送信する。また U_i は暗号文に対するゼロ知識証明も送信し、M がその正当性を検証する。M は各 U_i に対応する検査ベクトルを暗号文に組み込んだ後、それらの証明書を発行する。最後に M は暗号文及び証明書を R へ送信する。

3.3 安全性の定義

提案する評価システムが満たす安全性は以下の通りである。

- **正当性**: **Vote** において U_i の設定した評価点が正常な値である。
- **匿名性**: **Show** から R を特定できない。また、2 つの **Show** の履歴が同じ R のものであることも特定できない。
- **秘匿性**: 各 V_i の評価が R 以外の者には分からない。また、積算された評価点も R 以外の者には分からない。
- **偽造不能性**: **Show** において行われるゼロ知識証明においては、R は総評価値の導出の際に不正がないことを確認できる。

4. 提案方式

4.1 提案方式の概要

Setup において、R は、BBS 暗号に必要な秘密鍵と公開鍵を **BBS-Keygen** によって生成する。M は、AHO 署名と LH 署名に必要な秘密鍵と公開鍵をそれぞれ **LH-Keygen** と **AHO-Keygen** によって生成する。また、**AHO-Sign** によって想定される全ての評価値に対して証明書を発行し、それらも公開する。

Vote において、U_i は R の評価値を決定した後、BBS 暗号の **Encrypt** によって、評価値を暗号化し M に送信する。また暗号文に対して、M が公開した証明書の中から暗号文に対応したものを選んで GS 証明を行い、それも M へ送信する。M はこれを用いて評価値の正当性を検討する。また、U_i ごとに対応する検査ベクトル $\vec{E_i}$ を暗号文に付加する。続いて M は **LH-Sign** によって、検査ベクトルを付加した暗号文に対する証明書を発行し R に送信する。

Calculate において、R は各暗号文を BBS 暗号の準同型性を用いて総評価値の暗号文へ圧縮する。この際に暗号文に含まれる検査ベクトルも圧縮され $\vec{E}$ が得られる。このとき、n 個のベクトルを重複なく乗算すると、ベクトル $\vec{E_{cor}}$ が得られる。これを用いて後に評価の重複を検出することができる。次に R は BBS 暗号の **Decrypt** によって総評価値の暗号文を復号し総評価値を得る。最後に R は LH 署名の **SignDerive** によって証明書の圧縮を行い、GS 証明を用いてその正当性のゼロ知識証明を作成し公開する。

Show において、任意のユーザは R の公開した GS 証明の検証を行う。GS 証明では総評価値および圧縮された LH 署名が秘密情報となるため、任意のユーザはこれに関する情報を得ることはできない。

4.2 検査ベクトルの改善

従来方式と提案方式における検査ベクトルの改善点について述べる。まず以下に [3] で示されている従来方式での検査ベクトルの振る舞いを示す。

- **Vote** において n 個の暗号文にそれぞれ検査ベクトル $\vec{E_i} = (1, \cdots, g, 1, \cdots, 1)$($i$ 番目の要素を g、その他の要素を全て 1 とした n 次元ベクトル) が付加される。

$$\begin{aligned}\vec{E_1} &= (g, 1, \cdots, 1, 1)\\ \vec{E_2} &= (1, g, \cdots, 1, 1)\\ &\vdots\\ \vec{E_{n-1}} &= (1, 1, \cdots, g, 1)\\ \vec{E_n} &= (1, 1, \cdots, 1, g)\end{aligned}$$

- **Calculate** において暗号文の圧縮の際に検査ベクトルも圧縮され $\vec{E}$ が得られる。このとき圧縮では、検査ベクトルのそれぞれ i 番目の要素ごとに積算がなされ、n 個の暗号文が重複なく圧縮されているときは、それぞれの要素の積がすべて g となっている $\vec{Ecor} = (g, g, \cdots, g)$ が得られる。また、i 番目の暗号文が重複しているときは、$\vec{E}$ の i 番目の要素の g の次数が 2 以上となる。つまり $\vec{E}$ と $\vec{E_{cor}}$ を比較することで何番目の暗号文が重複しているかを検出することができる。

しかし、検査ベクトルが n 個の要素を持つベクトルであることから、**LH-sign** において署名される検査ベクトルを付加した暗号文の要素数は $n+3$ 個となる。LH 署名では公開鍵の要素数が、署名されるメッセージベクトルの要素数に比例するため、評価ユーザ数 n が莫大に増加した場合、公開鍵のデータサイズも莫大なものになる恐れがある。

そこで提案方式では、検査ベクトルに使われる g を $n'(\simeq \sqrt{n})$ 個に増やすことによってベクトルの要素数を $\sqrt{n}$ に近い値に抑えることで公開鍵データサイズに関する問題を解決する。以下に提案方式での検査ベクトルの振る舞いを示す。

- **Vote** において n 個の暗号文にそれぞれ検査ベクトル $\vec{E_i} = (1, \cdots, g_l, 1, \cdots, 1)$($i'$ 番目の要素を g_l、その他の要素を全て 1 とした n' 次元ベクトル、ただし $i = (i'-1) \times n' + l$) が付加される。

$$\begin{aligned}\vec{E_1} &= (g_1, 1, \cdots, 1, 1)\\ \vec{E_2} &= (g_2, 1, \cdots, 1, 1)\\ &\vdots\\ \vec{E_{n'}} &= (g_{n'}, 1, \cdots, 1, 1)\\ \vec{E_{n'+1}} &= (1, g_1, \cdots, 1, 1)\\ &\vdots\\ \vec{E_{n-1}} &= (1, 1, \cdots, 1, g_{n'-1})\\ \vec{E_n} &= (1, 1, \cdots, 1, g_{n'})\end{aligned}$$

- **Calculate** において暗号文の圧縮の際に検査ベクトルも圧縮され $\vec{E}$ が得られる。このとき圧縮では、検査ベクトルのそれぞれ i 番目の要素ごとに積算がなされ、n 個の暗号文が重複なく圧縮されているときは、それぞれの要素の積がすべて $g_{pro} = g_1 \times g_2 \times \cdots \times g_{n'}$ となっている $\vec{Ecor} = (g_{pro}, g_{pro}, \cdots, g_{pro})$ が得られる。また、i 番目の暗号文が重複しているときは、$\vec{E}$ の i' 番目の要素が $g_{pro} \times g_l$ となる。つまり $\vec{E}$ と $\vec{E_{cor}}$ を比較することで何番目の暗号文が重複しているかを検出することができる。

4.3 アルゴリズム

Setup

(1) M はセキュリティパラメータ λ を選び、$p > 2^\lambda$ を満たす素数 p を位数とした双線形写像 $(\mathbb{G}, \mathbb{G}_T)$ を作る。

(2) R は **BBS-Keygen($\mathbb{G}$)** を行う。
- 秘密鍵 $sk_R = (x, y)$ を $\mathbb{Z}_p^2$ からランダムに選ぶ。
- 公開鍵 $pk_R = (g, \tilde{g}, f, \tilde{f})$ について、$g, \tilde{g}$ をそれぞれランダムに $\mathbb{G}$ から選び、$f, \tilde{f}$ を以下のように計算する。

$$f = g^x$$
$$\tilde{f} = g^y$$

(3) M は $\sqrt{n} \leq n' < \sqrt{n+1}$ となる整数 n' をとる。
(4) M は n' 個の生成元 $g_1, \cdots, g_{n'}$ を生成する。
(5) M は **LH-Keygen($\mathbb{G}, n'+3$)** を行う。
- h をランダムに $\mathbb{G}$ から選び、$\alpha_z, \alpha_r, \beta_z$ をランダムに $\mathbb{Z}_p$ から選ぶ。
- g_z, g_r, h_z を以下のように定める。

$$g_z = h^{\alpha_z}$$
$$g_r = h^{\alpha_r}$$
$$h_z = h^{\beta_z}$$

- $j = 1, \cdots, n'+3$ について $\chi_j, \gamma_j, \delta_j$ をそれぞれランダムに $\mathbb{Z}_p$ から選び、g_j, h_j を以下のように定める。

$$g_j = g_z^{\chi_j} \cdot g_r^{\gamma_j}$$
$$h_j = h_z^{\chi_j} \cdot h^{\delta_j}$$

- $\overline{\mathbf{w}} = \{w_0, w_1, \cdots, w_L\}$ を $\mathbb{G}^{L+1}$ からランダムに選び、ハッシュ関数 H を以下のように定義する。ただし、$\tau[l]$ は τ の左から l ビット目の値 (すなわち 0 か 1) を指すものである。

$$H_{\mathbb{G}}(\tau) = w_0 \cdot \prod_{k=1}^{L} w_k^{\tau[l]}$$

(6) M は **AHO-Keygen($\mathbb{G}$,1)** を行う。
- G_r, H_r をそれぞれランダムに $\mathbb{G}$ から選ぶ。
- γ_z, δ_z をそれぞれランダムに $\mathbb{Z}_p$ から選び、γ, δ をそれぞれランダムに $\mathbb{Z}_p$ から選ぶ。
- $G_z = G_r^{\gamma_z}, H_z = H_r^{\delta_z}$ と定め、$G = G_r^{\gamma}, H = H_r^{\delta}$ と定める。
- α, β をそれぞれランダムに $\mathbb{Z}_p$ から選び、$A = e(G_r, g^{\alpha}), B = e(H_r, g^{\beta})$ とする。

(7) M は全ての評価値 $s_i (= 1, \cdots, K)$ に対して **AHO-Sign(sk_M)** を行う。
- $\zeta, \rho, \tau, \varphi, \omega$ をそれぞれランダムに $\mathbb{Z}_p$ から選ぶ。
- $V_{i1}, \cdots, V_{i7}$ を以下のように計算する。

$$V_{i1} = g^{\zeta}$$
$$V_{i2} = g^{\rho - \gamma_z \zeta} \cdot (\tilde{g}^{s_i})^{-\gamma}$$
$$V_{i3} = G_r^{\tau}$$
$$V_{i4} = g^{(\alpha - \rho)/\tau}$$
$$V_{i5} = g^{\varphi - \delta_z \zeta} \cdot (\tilde{g}^{s_i})^{-\delta}$$
$$V_{i6} = H_r^{\omega}$$
$$V_{i7} = g^{(\beta - \varphi)/\omega}$$

- $\vec{\varsigma_{s_i}} = (V_{i1}, \cdots, V_{i7})$ とし、$List = (\vec{\varsigma_{s_1}}, \cdots, \vec{\varsigma_{s_K}})$ とする。

(8) M の持つ秘密鍵、公開鍵をそれぞれ、以下のように定義する。

$$sk_M = (h_z^{\alpha_r}, \{\chi_j, \gamma_j, \delta_j\}_{j=1}^{n'+3}, \alpha, \beta, \gamma_z, \delta_z, \gamma, \delta)$$
$$pk_M = (g_z, g_r, h_z, h, \{g_j, h_j\}_{j=1}^{k+3}, \overline{\mathbf{w}}$$
$$, G_r, H_r, G_z, H_z, G, H, A, B, List)$$

Vote

(1) U_i は R の評価値 s_i を決定し、それに対応した $\vec{\varsigma_{s_i}}$ を pk_M の $List$ から取得する。
(2) U_i は **Encrypt(pk_R, s_i)** を行う。
- 評価値 s_i に対して、r_i,t_i をそれぞれ $\mathbb{Z}_p$ からランダムに選び、pk_R を用いて $\vec{S_i} = (C_{i1}, C_{i2}, C_{i3})$ を以下のように計算して s_i 暗号化する。

$$C_{i1} = f^{r_i}, C_{i2} = \tilde{f}^{t_i}, C_{i3} = \tilde{g}^{s_i} \cdot g^{r_i + t_i}$$

(3) U_i は s_i に対応した $\vec{\varsigma_{s_i}}$ の要素 $(V_{i1}, \cdots, V_{i7})$ を用いて GS 証明を行い、s_i が正常な評価値であることを示す。検証式は以下のようになる。

$$A = e(G_z, V_1) \cdot e(G_r, V_2) \cdot e(V_3, V_4) \cdot e(G, \tilde{g}^{s_i})$$
$$B = e(H_z, V_1) \cdot e(H_r, V_5) \cdot e(V_6, V_7) \cdot e(H, \tilde{g}^{s_i})$$
$$C_{i1} = f^{r_i}, C_{i2} = \tilde{f}^{t_i}, C_{i3} = \tilde{g}^{s_i} \cdot g^{r_i + t_i}$$

(4) U_i は $\vec{S_i}$ と上記の GS 証明を M に送信する。
(5) M は $\vec{S'}_i = (\vec{S_i}, \vec{E_i})$ を作成する。ただし $\vec{E_i}$ は i' 番目の要素を g_l とし、他の要素を全て 1 とした n' 次元ベクトル $(i = (i'-1) \times n' + l)$ である。
(6) M は **LH-Sign($sk_M, \tau, \vec{S'}_i$)** を行う。ここで τ は L bit の R の ID とする。
- θ と ρ をランダムに $\mathbb{Z}_p$ から選ぶ。
- $\vec{\sigma_i} = (z_i, q_i, u_i, v_i)$ を以下のように計算する。ただし、$\vec{S'}_i$ の j 番目の要素を m'_{ij} とする。

$$z_i = g_r^{\theta} \cdot \prod_{j=1}^{n'+3} {m'}_{ij}^{-\chi_j}$$
$$q_i = g_z^{-\theta} \cdot \prod_{j=1}^{n'+3} {m'}_{ij}^{-\gamma_j}$$
$$u_i = (h_z^{\alpha_r})^{-\theta} \cdot \prod_{j=1}^{n'+3} {m'}_{ij}^{-\delta_j} \cdot H_{\mathbb{G}}(\tau)^{-\rho}$$
$$v_i = h^{\rho}$$

(7) M は $\{\vec{S'}_i\}_{i=1}^n$ と $\{\vec{\sigma}_i\}_{i=1}^n$ を R に送信する。

Calculate

(1) R は以下のように $\vec{S} = \prod_{i=1}^n \vec{S'_i}$ を計算する。ただし $g_{sum} = g_1 \cdot g_2 \cdot \cdots \cdot g_{n'}$ とする。

$$\begin{aligned}\vec{S} &= \prod_{i=1}^{n} \vec{S'}_i \\ &= \prod_{i=1}^{n} (\vec{S_i}, \vec{E_i}) \\ &= \prod_{i=1}^{n} ((C_{i1}, C_{i2}, C_{i3}), \vec{E_i}) \\ &= (\prod_{i=1}^{n} C_{i1}, \prod_{i=1}^{n} C_{i2}, \prod_{i=1}^{n} C_{i3}, g_{pro}, g_{pro}, \cdots, g_{pro}) \\ &= (f^{\sum_{i=1}^n r_i}, \tilde{f}^{\sum_{i=1}^n t_i}, \tilde{g}^{\sum_{i=1}^n s_i} \cdot g^{\sum_{i=1}^n (r_i+t_i)} \\ &\quad , g_{pro}, g_{pro}, \cdots, g_{pro})\end{aligned}$$

(2) R は $\mathbf{Decrypt}(sk_R, \vec{S})$ を行う。ただし、$\vec{S}$ の j 番目の要素を m_j とする。

- 以下の計算によって $s_R (= \sum_{i=1}^n s_i)$ を得る。

$$\begin{aligned}s_R &= \log_{\tilde{g}} \frac{m_3}{\{m_1\}^{\frac{1}{x}} \cdot \{m_2\}^{\frac{1}{y}}} \\ &= \log_{\tilde{g}} \frac{\tilde{g}^{\sum_{i=1}^n s_i} \cdot g^{\sum_{i=1}^n (r_i+t_i)}}{\{f^{\sum_{i=1}^n r_i}\}^{\frac{1}{x}} \cdot \{\tilde{f}^{\sum_{i=1}^n t_i}\}^{\frac{1}{y}}} \\ &= \log_{\tilde{g}} \frac{\tilde{g}^{\sum_{i=1}^n s_i} \cdot g^{\sum_{i=1}^n (r_i+t_i)}}{g^{\sum_{i=1}^n r_i} \cdot g^{\sum_{i=1}^n t_i}} \\ &= \log_{\tilde{g}} \frac{\tilde{g}^{\sum_{i=1}^n s_i} \cdot g^{\sum_{i=1}^n (r_i+t_i)}}{g^{\sum_{i=1}^n (r_i+t_i)}} \\ &= \log_{\tilde{g}} \tilde{g}^{\sum_{i=1}^n s_i} \\ &= \sum_{i=1}^{n} s_i\end{aligned}$$

(3) R は $\mathbf{SignDerive}(pk_M, \tau, \{\vec{\sigma_i}\}_{i=1}^n)$ を行う。

- ρ' を $\mathbb{Z}_p$ からランダムに選ぶ。
- $\vec{\sigma} = (z, q, u, v)$ を以下のように計算する。

$$z = \prod_{i=1}^{n} z_i$$
$$q = \prod_{i=1}^{n} q_i$$
$$u = \prod_{i=1}^{n} u_i \cdot H_{\mathbb{G}}(\tau)^{-\rho'}$$
$$v = \prod_{i=1}^{n} v_i \cdot h^{\rho'}$$

(4) R は次の検証式の GS 証明を生成し公開する。ただし、$\vec{S}$ の j 番目の要素を m_j とする。

$$1_{\mathbb{G}_T} = e(g_z, z) \cdot e(g_r, q) \cdot \prod_{j=1}^{n'+3} e(g_j, m_j)$$
$$1_{\mathbb{G}_T} = e(h_z, z) \cdot e(h, u) \cdot e(H_{\mathbb{G}}(\tau), v) \cdot \prod_{j=1}^{n'+3} e(h_j, m_j)$$
$$m_3 = \tilde{g}^{s_R} \cdot m_1^{\frac{1}{x}} \cdot m_2^{\frac{1}{y}}$$

(5) ここで、1,2 番目の等式は LH 署名の検証を示し、3 番目の等式は積算された暗号文が評価値 s_R から復号されたことを示している。

Show

(1) 任意のユーザは、s_R が積算された暗号文の復号結果であることを確認するために GS 証明を検証する。

5. 実装と結果

5.1 実装環境

表 1 に実装環境を示す。本研究では群演算とペアリング計算を必要とするため、ELiPS ライブラリを使用して C 言語によって実装を行った。

表 1

OS	Ubuntu 14.04
CPU	Intel®Core™ i5-4460(3.2GHz×4)
メモリ	7.8GiB
使用言語	C 言語
多倍長ライブラリ	GMP-6.0.0
ペアリングライブラリ	ELiPS 254-1-03

5.2 実験結果

5.2.1 公開鍵のデータサイズの比較

図 1 に評価数の増加に対する、従来方式および提案方式における公開鍵データサイズの変化を示す。提案方式では LH 署名の公開鍵のデータサイズが評価数 n に比例しているが、提案方式では検査ベクトルの改善により、検査ベクトルの要素数が $n'(\simeq \sqrt{n})$ となったことにより LH 署名の公開鍵のデータサイズも $\sqrt{n}$ に比例しているため効率的である。

また AHO 署名の公開鍵のデータサイズについて、評価数 n には依存しないが、公開鍵に含まれる $List$ の要素数は M の想定した K 個の評価点には比例するため、公開鍵のデータサイズも K に比例する。図 1 では $K = 5$ と固定したデータサイズとなっている。

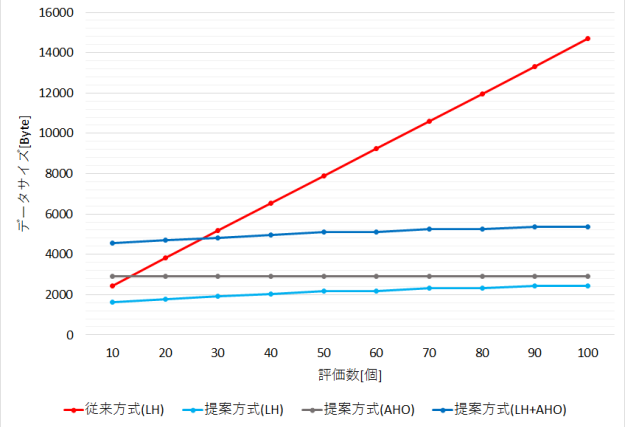

図 1 公開鍵データサイズ

5.2.2 AHO 署名によるオーバヘッド

提案方式では、**Vote** の（3）において AHO 署名によるゼロ知識証明を追加することで不正な評価点の送信を防止している。その追加したゼロ知識証明の処理時間はおよそ 237$[ms]$ であり、それほど実用性は損なわれていない。

6. まとめ

本研究では、本研究では、線型準同型署名を用いた管理者に対して秘匿性を持つ評価システムの改善を提案し、その PC 実装を行った。提案方式では、AHO 署名によるゼロ知識証明を用いることで、評価点の正当性の検証を可能にした。また、検査ベクトルの改良をすることで、公開鍵のデータサイズを削減した。

謝辞 本研究の一部は、JSPS 科研費 (JP16K00187) の助成を受けている。

参考文献

[1] D. Boneh, X. Boyen, and H. Shacham, "Short Group Signatures", Crypto '04, LNCS 3152, pp.41-55, 2004.
[2] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, and M. Ohkubo, Structure-preserving signatures and commitments to group elements", Crypto 2010, LNCS 6223, pp. 209-236, 2010.
[3] B. Libert, T. Peters, M. Joye, and M. Yung, "Linearly Homomorphic Structure-Preserving Signatures and Their Applications", Crypto 2013, LNCS 8043, pp.289-307, 2013.
[4] J. Groth and A. Sahai, "Efficient non-interactive proof sys-tems for bilinear groups", EUROCRYPT 2008, LNCS 4965, pp.415-432, 2008.
[5] 野村 智也, 中西 透, 舩曳 信生, "管理者に対して強固な秘匿性を持つ評価システムの提案", 信学技報, ISEC 2011 － 64, 2011.
[6] 上野山 大貴, 中西 透, "線型準同型署名を用いた管理者に対して秘匿性を持つ評価システム", 信学技報, ISEC 2017 － 83, 2017.