Vol.2018-CSEC-83 No.15
2018/12/14

アキュームレータを用いた 評価値ベースのブラックリスト型匿名認証の拡張

金谷 健士[a)]　中西 透[b)]

概要：第 3 者機関を必要せず不正ユーザを匿名のまま排除するブラックリスト型匿名認証において、従来、アキュームレータにより高速化した方式が提案されている。さらにこの方式は、評価値ベースとなっており、各ユーザは自身のアクティビティに対してサービス提供者か他ユーザからセッションの終了時に評価を受け、その合計値に基づいてブラックリスト化できる。しかし、この従来方式では、ユーザの評価値の合計が公開されてしまうため、サービス提供者はその点数からユーザを追跡することができてしまうという問題がある。そこで、本研究では、評価値を範囲証明することでその具体値を秘匿した方式に拡張する。また、一度認証に成功したアクティブなユーザが高速に認証を行える方式への拡張も示す。

キーワード：匿名認証, ブラックリスト, アキュームレータ

Extension of Blacklistable Anonymous Credential System with Reputation Using Accumulator

Takeshi Kanatani[a)]　Toru Nakanishi[b)]

1. はじめに

現在利用されている ID ベースの認証では、SP（サービス提供者）がユーザを ID とパスワードなどの認証情報を用いて認証を行う。しかし、この手法では ID から利用履歴情報が SP に漏れるため、プライバシーに問題がある。

そこで、ID を秘匿しながら自身が正規ユーザであることだけを証明できる匿名認証が提案されている。しかし従来のグループ署名 [7] による匿名認証では、不正ユーザ対策のため必要とされる第 3 者機関 (TTP) が正規ユーザを特定し得る問題があったため、TTP なしで匿名認証可能なブラックリスト型匿名認証 [1, 4, 5] が提案されている。この方式では、ユーザのセッション ID がブラックリストに含まれていないことを確認することで検証を行う。認証方式 [1] ではブラックリストサイズに比例した認証時間を要するという問題があったが、[4, 5] ではアキュームレータを用いた効率的な認証方式が提案されている。アキュームレータではブラックリストが圧縮されるため、認証データサイズがブラックリストサイズに依存しない。しかし、これらの方式 [1, 4, 5] では、正規ユーザか不正ユーザかという単純な評価しかできない。そこで、先行研究 [8] では、評価値ベースで認証可能なブラックリスト型匿名認証を提案している。各ユーザのセッションに他ユーザ・SP が評価した評価値を持たせ、認証されているユーザのこれまでの評価値の合計を確認できる。これにより、ユーザの過去のアクティビティの評価に応じてブラックリスト化できる。

しかし、この方式では評価値の合計が公開されてしまっているため、SP はその点数からユーザを区別し、追跡できてしまう問題があった。そこで本研究では、この評価値の和に対し範囲証明することで具体的な評価値を秘匿してプライバシーを強化する。また、従来方式 [8] では、認証時間が認証されるユーザの過去の認証回数に依存する問題がある。そこでアクティブユーザに高速な認証を提供する

†1 広島大学
Hiroshima University
a) m173832@hiroshima-u.ac.jp
b) t-nakanishi@hiroshima-u.ac.jp

ExpressLane 型 [9] への拡張も示す。これにより、一度認証に成功すると、過去の時間区分での認証を省略することにより高速な認証を行うことができる。

2. 数学的準備

2.1 双線形写像

本研究では以下の双線形群を利用する。

(1) G_1, G_2, G_T を位数 p の巡回群とする。

(2) G_1, G_2 の生成元をそれぞれ g, h とする。

(3) 双線形写像 $e : G_1 \times G_2 \to G_T$ は以下の双線形性と非退化性を満たす。

- **双線形性:** 任意の $u, u' \in G_1, v, v' \in G_2$ について、$e(uu', v) = e(u, v)e(u', v)$ 及び $e(u, vv') = e(u, v)e(u, v')$。
- **非退化性:** $e(g, h) \neq 1$。

上記の双線型写像は楕円曲線上のペアリングにより実現できる。

2.2 知識の署名

知識の署名は SPK: Signature based on Proof of knowledge と記述される。SPK は知識の証明を変換することで得られる。

知識の零知識証明とは証明者 P と検証者 V の対話型プロトコルで、ある関係を満たす秘密情報を知っていることを秘密情報を漏らすことなく証明する。離散対数の秘密情報 x を知ることを示すメッセージ m における SPK は以下のように記述される。

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

(1) リプレゼンテーションの SPK

$y = g^x$ となる離散対数の関係は、複数の底 g_1, g_2 を導入することで、秘密情報 x_1, x_2 に対し、$y = g_1^{x_1} g_2^{x_2}$ の用に拡張できる。この $SPK\{(x_1, x_2) : y = g_1^{x_1} g_2^{x_2}\}$ を以下に示す。

秘密鍵: $x_1, x_2 \in Z_p^*$

公開鍵: $y = g_1^{x_1} g_2^{x_2}$

メッセージ m に対する署名生成: $r_1, r_2 \in Z_p^*$ を選び、$t = g_1^{r_1} g_2^{r_2}$ を計算する。 $c = H(g_1||g_2||y||t||m)$ とし、$s_1 = r_1 - cx_1 \pmod p$, $s_2 = r_2 - cx_2 \pmod p$ を計算する。 署名は (c, s_1, s_2) とする。

検証: $t' = g_1^{s_1} g_2^{s_2} y^c$ を計算し、$c = H(g_1||g_2||y||t'||m)$ となるか検証する。

(2) 複数の検証式における SPK

2 つ以上の検証式における SPK も構成可能である。以下に $SPK\{(x_1, x_2) : y = g_1^{x_1} \wedge z = g_1^{x_2} g_2^{x_1}\}(m)$ を示す。

秘密鍵: $x_1, x_2 \in Z_p^*$

公開鍵: $y = g_1^{x_1}, z = g_1^{x_2} g_2^{x_1}$

メッセージ m に対する署名生成: $r_1, r_2 \in Z_p^*$ を選び、$t_y = g_1^{r_1}, t_z = g_1^{r_2} g_2^{r_1}$ を計算する。 $c = H(g_1||g_2||y||z||t_y||t_z||m)$ とし、$s_1 = r_1 - cx_1 \pmod p$, $s_2 = r_2 - cx_2 \pmod p$ を計算する。 署名は (c, s_1, s_2) とする。

検証: $t_y = g_1^{s_1} y^c, t'_z = g_1^{s_2} g_2^{s_1} z^c$ を計算し $t = t'$ となるか検証する。 また $c = H(g_1||g_2||y||z||t'_y||t'_z||m)$ となるか検証する。

2.3 *AHO* 署名

AHO 署名は複数の群要素のメッセージに署名できる方式である。また署名検証のペアリングの関係式を零知識証明できる。L 個の要素に対する AHO 署名の適用は下記の 3 つのプロトコルで構成される。

(1) **AHOKeyGen:** AHO 署名の秘密鍵を出力する。素数の位数 p である双線形群 G_1, G_2 を選び、その線形写像を e とする。$G_r, H_r \in G_1$, $h \in G_2$, $\mu_z, v_z, \mu_1, ..., \mu_L, v_1, ..., v_L, \alpha_a, \alpha_b \in Z_p$ をそれぞれランダムに選ぶ。$G_z = G_r^{\mu_z}, H_z = H_r^{v_z}, G_1 = G_r^{\mu_1}, ..., G_L = G_r^{\mu_L}, H_1 = H_r^{v_1}, ..., H_L = H_r^{v_L}, A = e(G_r, h^{\alpha_a}), B = e(H_r, h^{\alpha_b})$ を計算する。

出力:*AHO* 署名の公開鍵

$$pk_{AHO} = (G, p, e, g, G_r, H_r, G_z, H_z, G_1, ..., G_L, H_1, ..., H_L)$$

AHO 署名の秘密鍵

$$sk_{AHO} = (\alpha_a, \alpha_b, \mu_z, v_z, \mu_1, ..., \mu_L, v_1, ..., v_L)$$

(2) **AHOSign:** L 個のメッセージ $(M_1, ..., M_L)$ と、上記の秘密鍵 sk_{AHO} から、署名を作成する。 まず $\beta, \epsilon, \eta, \iota, \kappa \in Z_p$ をランダムに選び、以下のように $\theta_1, \sim \theta_7$ を計算する。

$\theta_1 = h^{\beta}$, $\theta_2 = h^{\epsilon - \mu_z \beta} \prod_{i=1}^{L} M_i^{-\mu_i}$, $\theta_3 = h^{\eta}$, $\theta_4 = h^{(\alpha_a - \epsilon)/\eta}$, $\theta_5 = h^{\iota - v_z \beta} \prod_{i=1}^{L} M_i^{-v_i}$, $\theta_6 = h^{\kappa}$, $\theta_7 = h^{(\alpha_b - \iota)/\kappa}$

出力: 署名 $\sigma = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7)$

(3) **AHOverify:** 上記のメッセージ $(M_1, ..., M_L)$ と署名 σ に対し、検証を行う。 以下の検証式を満たせば検証成功となる。

$A = e(G_z, \theta_1) \cdot e(G_r, \theta_2) \cdot e(\theta_3, \theta_4) \cdot \prod_{i=1}^{L} e(G_i, M_i)$

$B = e(H_z, \theta_1) \cdot e(H_r, \theta_5) \cdot e(\theta_6, \theta_7) \cdot \prod_{i=1}^{L} e(H_i, M_i)$

2.4 評価値ベース型アキュームレータ

アキュームレータにより、複数の要素を 1 つのデータにまとめ、要素間の関係を検証できる。[4, 5] では、2 つの集合において包含関係を示すアキュームレータを利用している。具体的には、2 つの集合 U, V を $\{1, ..., n\}$ の部分集合

とすると、$U \subset V$ であるかを検証できる。このアキュームレータをブラックリスト型匿名認証に使用することにより認証時間を軽減している。アキュームレータの検証では集合 U と V 間のマッチングを行っているため、$|U \cap V|$ の個数の正しさが検証される。さらに [8] では、評価値ベースのアキュームレータに拡張されており、U と V のマッチングの個数ではなくマッチングした要素に対する評価値の和が得られる。本研究では、[8] と同様に評価値ベースのアキュームレータを用いる。このアキュームレータは、以下のアルゴリズムから構成される。

AccSetup: これは公開情報を出力するアルゴリズムである。素数の位数 p である双線形群 G_1, G_2 を選び、その双線形写像を e とする。$g \in G_1, h \in G_2, \gamma \in Z_p$ をランダムに選び、以下の計算を行う。

$g_1 = g^{\gamma^1}$, ..., $g_n = g^{\gamma^n}$, $g_{n+2} = g^{\gamma^{n+2}}$, ..., $g_{2n} = g^{\gamma^{2n}}$, $h_1 = h^{\gamma^1}$, ..., $h_n = h^{\gamma^n}$, $h_{n+2} = h^{\gamma^{n+2}}$, ..., $h_{2n} = h^{\gamma^{2n}}$

$(p, G, e,\ g_1 = g^{\gamma^1},\ ...,\ g_n = g^{\gamma^n},\ g_{n+2} = g^{\gamma^{n+2}},\ ...,\ g_{2n} = g^{\gamma^{2n}},\ h_1 = h^{\gamma^1},\ ...,\ h_n = h^{\gamma^n},\ h_{n+2} = h^{\gamma^{n+2}},\ ...,\ h_{2n} = h^{\gamma^{2n}})$ を公開情報とする。

AccGen: これは公開情報と集合 V, V の各要素 i に対するスコア s_i を用いてアキュムレータを計算するアルゴリズムである。 各セッション ID $i \in V$ の評価値 $s_i \in Z$ に対して $acc_V = \prod_{i \in V} g_{n+1-i}^{s_i}$ として計算される。

AccWitGen: これは公開情報, V, s_i 集合 $U \subset V$ を用いてアキュームレータの検証に必要な補助情報 W を計算するアルゴリズムである。各セッション ID i の評価値を s_i に対して、$W = \prod_{i \in U} \prod_{j \in V}^{i \neq j} g_{n+1-j+i}^{s_j}$ として計算される。

AccVerify: このアルゴリズムは、公開情報, 集合 U, V, アキュームレータ acc_V, 補助情報 W を用いて、$U \subset V$ の評価値の合計値 $\sum_{i \in U} s_i$ を検証するアルゴリズムである。まず、$acc_U = \prod_{i \in U} h_i$ を計算する。そして以下の検証式を用いてアキュームレータの検証を行う。

$$\frac{e(acc_V, acc_U)}{e(g, W)} = e(g_1, h_n)^{\sum_{i \in U} s_i}$$

提案認証方式では集合 V に全ユーザの使用済 (評価値のついた) セッション ID の集合を用い、 あるユーザが持つセッション ID の集合 U に対する評価値の和を得るようにしている。

3. モデルと安全性

[8] の従来方式では、ユーザの評価値の合計が公開されてしまうため、匿名性が弱くなっており、SP や他のユーザは公開された合計点からユーザを追跡できてしまう恐れがある。そこで、本研究では評価の合計点の範囲のみを認証するよう拡張する。例えば、60 点以下のユーザを許可するシステムの場合、従来方式ではユーザの評価点の合計 56 点が公開された場合、1 回の評価点が 5 点以下であるなら、同時期の 0 点のユーザとは異なり、55 点のユーザと同一である可能性が高い。一方提案方式では 0-60 点の範囲にあることのみを示すことにより、具体値を秘匿したまま認証ポリシーを満たしていることを証明できるようになる。

ブラックリスト型匿名認証ではユーザと SP が各手続きを行う。SP は認証中のユーザが誰かを特定することができない。SP はセッション毎にそのセッションでの ID を発行しリスト L_S そのセッションでの評価値 s_i とともに追加する。各手続きは以下の通りである。

セットアップ: SP が自身の公開鍵、 秘密鍵を生成する。

登録: ユーザが選んだ秘密鍵に対する初期証明書をユーザに発行する。

認証: SP がユーザにセッション ID のリスト L_s と対応する評価値のリストを送付する。ユーザはこれらと証明書を用いて SP と匿名認証を行う。このとき、ユーザの行った全セッションの評価点の合計値の範囲が明らかになる。匿名範囲認証が成功した場合 (被認証ユーザの評価の範囲が認証ポリシーを満たしている場合)、SP はユーザへ今回のセッション ID と新しい証明書を発行し、この ID をリスト L_s へ今回のセッションの評価値とともに追加する。

3.1 安全性

ブラックリスト型匿名範囲認証は、以下の安全性を満たす。

匿名性: 認証時、SP はどのユーザと手続きをしているか特定することができない。また、任意の 2 つの認証においてその 2 ユーザが同一のユーザであるか知ることができない。

偽造不能性: 未登録である不当なユーザは認証に成功できない。また、ユーザは評価値の合計が範囲内のときのみ認証に成功する。

4. 提案方式

4.1 概要

[8] で提案した評価値ベースのブラックリスト型匿名範囲認証方式に基づく。 この方式は [4, 5] のアキュムレータを用いたブラックリスト型匿名認証方式において、[4, 5] で示したアキュームレータにより、ユーザの評価値の合計値を検証できるように変更したものである。しかし、この方式では評価値の合計値が公開されるため、ユーザ追跡が可能となり、匿名性に問題がある。

そこで、評価値の合計値が属する範囲のみが公開される範囲認証に拡張する。範囲認証のために、範囲内の全ての値 v に対して署名 $sig(v)$ を公開鍵として発行しておく。認証時にユーザは、自身の評価点の合計値に対する署名を零

知識証明する。これにより、具体的な値を示すことなく範囲に入っていることを検証できる。

4.2 アルゴリズム

セットアップ: SP は以下の処理により公開鍵と秘密鍵を生成する。

(1) SP: 双線形群 G_1, G_2 と、$g, \hat{g} \in G_1$, $h, \hat{h} \in G_2$, $\gamma \in Z_p^*$ をランダムに選ぶ。 アキュムレータの公開鍵 $pk_{acc} = (g_1 = g^{\gamma^1}, ..., g_n = g^{\gamma^n}, g_{n+2} = g^{\gamma^{n+2}}, ..., g_{2n} = g^{\gamma^{2n}}, h_1 = h^{\gamma^1}, ..., h_n = h^{\gamma^n}, h_{n+2} = h^{\gamma^{n+2}}, ..., h_{2n} = h^{\gamma^{2n}})$ を生成する。

(2) SP: AHO 署名の公開鍵と秘密鍵を計算する。その準備として、 $G_\gamma, H_\gamma \in G_1, \mu_z, v_z, \mu_1, \mu_2, \mu_3, \mu_4, \mu_5, v_1, v_2, v_3, v_4, v_5, \alpha_a, \alpha_b \in Z_p$ をランダムに選ぶ。 これらを使って、以下を計算する。
$G_z = G_\gamma^{\mu_z}, H_z = H_\gamma^{v_z}, G_1 = G_\gamma^{\mu_1}, ..., G_5 = G_\gamma^{\mu_5}$, $H_1 = H_\gamma^{v_1}, ..., H_5 = H_\gamma^{v_5}, A = e(G_\gamma, h^{\alpha_a}), B = e(H_\gamma, h^{\alpha_b})$
AHO 署名の公開鍵は、
$pk_{AHO} = (G_\gamma, H_\gamma, G_z, H_z, G_1, ..., G_5, H_1, ..., H_5)$,
AHO 署名の秘密鍵は、
$sk_{AHO} = (\alpha_a, \alpha_b, \mu_z, v_z, \mu_1, ..., \mu_5, v_1, ..., v_5)$ となる。

(3) SP: (2) と同様に、評価値の範囲に関する AHO 署名の公開鍵と秘密鍵を計算する。AHO 署名の公開鍵は、
$\hat{pk}_{AHO} = (\hat{G}_\gamma, \hat{H}_\gamma, \hat{G}_z, \hat{H}_z, \hat{G}_1, \hat{H}_1)$,
AHO 署名の秘密鍵は、
$sk_{AHO} = (\hat{\alpha}_a, \hat{\alpha}_b, \hat{\mu}_z, \hat{v}_z, \hat{\mu}_1, \hat{v}_1)$ となる。

(4) SP: 範囲内の全ての値 $j \in \{l_1, l_1+1, ..., l_2\}$ に対して、$\hat{sk}_{AHO}$ を用いて AHO 署名 $\rho_j = \Theta_{j1}, ..., \Theta_{j7}$ を計算する。

(5) SP: 全ユーザの使用済みセッション ID の集合を L_S とする。 また、セッション ID i での評価値を s_i とする。 このセットアップの段階では $L_S = \phi$ とする。

登録: ユーザは以下のプロトコルにより SP から初期証明書とその秘密鍵を取得する。

(1) ユーザ: 各ユーザは以下のパラメータをランダムに選ぶ。
・ユーザの秘密情報 x
・証明書のタグ T_0
・$r_{x,0}, r_{T_0} \in Z_p^*$ (x と T_0 の SPK に使用。)

(2) ユーザ: ユーザはコミットメントとして $C_{x,0} = h^x \hat{h}^{r_{x,0}}, C_{T_0} = h^{T_0} \hat{h}^{r_{T_0}}$ を計算し、SP へ $(C_{x,0}, C_{T_0})$ を送付する。 また、$SPK\{(x, T_0, r_{x,0}, r_{T_0}) : C_{x,0} = h^x \hat{h}^{r_{x,0}}, C_{T_0} = h^{T_0} \hat{h}^{r_{T_0}}\}$ を計算し送付する。

(3) SP: ユーザの使用済みセッション ID の集合を $L_U = \phi$ とする。 また $N_0 = 1, P_0 = h_1, R_0 = 0, r_{N_0} = r_{R_0} = 0$ とする。 コミットメントとして $C_{N_0} = h^{N_0} \hat{h}^{r_{N_0}}, C_{P_0} = P_0 \hat{h}^{R_0}, C_{R_0} = h^{R_0} \hat{h}^{r_{R_0}}$ を計算し、メッセージ $(C_{x,0}, C_{T_0}, C_{N_0}, C_{P_0}, C_{R_0})$ の AHO 署名 $\sigma_0 = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ をユーザに送付する。

(4) ユーザ: ユーザの使用済みセッション ID の集合を $L_U = \phi$ とする。 また $N_0 = 1, P_0 = h_1, R_0 = 0, r_{N_0} = r_{R_0} = 0$ とし、$C_{N_0} = h^{N_0} \hat{h}^{r_{N_0}}, C_{P_0} = P_0 \hat{h}^{R_0}, C_{R_0} = h^{R_0} \hat{h}^{r_{R_0}}$ を計算する。
ユーザの証明書を
$cert_0 = (L_U, T_0, N_0, P_0, R_0, C_{x,0} C_{T_0}, C_{N_0}, C_{P_0}, C_{R_0}, r_{x,0}, r_{T_0}, r_{N_0}, r_{R_0}, \sigma_0)$ とする。また、その秘密鍵を $sec = x$ とする。

認証: 以下のプロトコルにより、ユーザは SP に自身のスコアの合計値の範囲を示す。

(1) SP: SP はユーザに L_S と全ての $i \in L_S$ に対する s_i を送付する。

(2) ユーザ： 証明書のコミットメント $(C_{x,t-1}, C_{T_{t-1}}, C_{N_{t-1}}, C_{P_{t-1}}, C_{R_{t-1}})$ を以下のようにランダム化する。乱数 $r'_{x,t-1}, r'_{T_{t-1}}, R'_{-1}, r'_{N_{t-1}}, r'_{R_{t-1}}$ を選ぶ。
$r''_{x,t-1} = r_{x,t-1} + r'_{x,t-1}$,
$r''_{T_{t-1}} = r_{T_{t-1}} + r'_{T_{t-1}}$,
$r''_{N_{t-1}} = r_{N_{t-1}} + r'_{N_{t-1}}$,
$R''_{t-1} = R_{t-1} + R'_{t-1}$,
$r''_{R_{t-1}} = r_{R_{t-1}} + r'_{R_{t-1}}$ とする。
以下の計算を行う。
$C'_{x,t-1} = C_{x,t-1} \hat{h}^{r'_{x,t-1}} = h^x \hat{h}^{r''_{x,t-1}}$,
$C'_{T_{t-1}} = C_{T_{t-1}} \hat{h}^{r'_{T_{t-1}}} = h^{T_{t-1}} \hat{h}^{r''_{T_{t-1}}}$
$C'_{N_{t-1}} = C_{N_{t-1}} \hat{h}^{r'_{N_{t-1}}} = h^{N_{t-1}} \hat{h}^{r''_{N_{t-1}}}$
$C'_{P_{t-1}} = C_{P_{t-1}} \hat{h}^{R'_{t-1}} = P_{t-1} \hat{h}^{R''_{t-1}}$
$C'_{R_{t-1}} = C_{R_{t-1}} \hat{h}^{r'_{R_{t-1}}} = h^{R_{t-1}} \hat{h}^{r''_{R_{t-1}}}$
$com_1 = (C'_{x,t-1}, C'_{T_{t-1}}, C'_{N_{t-1}}, C'_{P_{t-1}}, C'_{R_{t-1}})$ とする。

(3) ユーザ: ユーザが所持している AHO 署名 を [5] の手法によりランダム化して $\sigma_{t-1} = (\theta'_1, ..., \theta'_7)$ を得て、コミットメント $\{C_{\theta'_i}\}_{i \in (1,2,5)} = \theta'_i \hat{h}^{r_{\theta'_i}}$ を計算する。$com_{AHO} = (\{\theta'_i\}_{i \in (3,5,6,7)}, \{C_{\theta'_i}\}_{i \in (1,2,5)})$ とする。

(4) ユーザ: ユーザのスコアの合計値 $S = \sum_{i \in U} s_i$ を計算する。S に対する範囲の AHO 署名 $\rho_S = (\Theta_{S1}, ..., \Theta_{S7})$ を公開鍵から取得する。(3) と同様にランダム化して $(\Theta'_{S1}, ..., \Theta'_{S7})$ を得る。

(5) SP： SP は L_S を用いて $acc_{L_S} = \prod_{i \in L_S} g_{n+1-i}^{s_i}$ を計算し、ユーザに送付する。

(6) ユーザ: ユーザは SP から送付された L_S, s_i を使用して、$W = \prod_{j \in L_U} \prod_{i \in L_S}^{i \neq j} g_{n+1-i+j}^{s_i}$ を計算する。 実際には、以前使用した補助情報 W_{t-1}、リスト L_S の差分 δ_{L_S} に対して、$W_t = W_{t-1} \cdot \prod_{j \in L_U} \prod_{i \in \delta_{L_S}} h_{n+1-i+j}^{s_i}$ を計算すればよい。 Z_p からランダムに選び、r_W としてコミットメント $C_W = W \hat{h}^{rw}$ を計算する。
$com_{acc} = C_W$ とする。

(7) ユーザ: $R_t = R''_{t-1} = R_{t-1} + R'_{t-1}$ とする。 $r_{R_t} \in Z_p$

をランダムに選ぶ。$C_{R_t} = h^{R_t}\hat{h}^{r_{R_t}}$ を計算する。同様に T_t, r_{T_t} をランダムに選び、 $C_{T_t} = h^{T_t}\hat{h}^{r_{T_t}}$ を計算する。$com_2 = (C_{R_t}, C_{T_t})$ とする。

(8) ユーザ: SP へ $(com_1, com_{AHO}, com_{acc}, com_2), T_{t-1}$ を送付する。

(9) ユーザ: SP へ以下の知識の署名 SPK を計算して送付する。

$SPK\{(x, T_{t-1}, N_{t-1}, R_{t-1}, r''_{x,t-1}, r''_{T_{t-1}}, r''_{N_{t-1}}, r''_{R_{t-1}}, r_R, r'_{x,t-1}, r'_{t_{t-1}}, r'_{N_{t-1}}, r'_{R_{t-1}}, R'_{t-1}, r'_{R_{t-1}}, r_{\theta'_1}, r_{\theta'_2}, r_{\theta'_3}, \hat{r}_N, r_W)$:

$C'_{x,t-1} = h^x\hat{h}^{r''_{x,t-1}} \wedge C'_{T_{t-1}}h^{-T_{t-1}}\hat{h}^{r''_{T_{t-1}}} \wedge C'_{N_{t-1}} = h^{N_{t-1}}\hat{h}^{r''_{N_{t-1}}} \wedge C'_{R_{t-1}} = h^{R_{t-1}}\hat{h}^{r''_{R_{t-1}}} \wedge C_{R_t} = h^{R_{t-1}+R'_{t-1}}\hat{h}^{r_{R_t}}$

$$\begin{aligned}&\wedge A^{-1}\ e(G_z, C_{\theta'_1})e(G_r, C_{\theta'_2})e(\theta'_3, \theta'_4)e(G_1, C'_{x,t-1})\\ &e(G_2, C'_{T_{t-1}})e(G_3, C'_{N_{t-1}})e(G_4, C'_{P_{t-1}})e(G_5, C'_{R_{t-1}})\\ &= e(G_z, \hat{h})^{r_{\theta'_1}}e(G_r, \hat{h})^{r_{\theta'_2}}e(G_1, \hat{h})^{r'_{x,t-1}}e(G_2, \hat{h})^{r'_{T_{t-1}}}\\ &\quad e(G_3, \hat{h})^{r'_{N_{t-1}}}e(G_4, \hat{h})^{R'_{t-1}}e(G_5, \hat{h})^{r'_{R_{t-1}}}\end{aligned} \tag{1}$$

$$\begin{aligned}&\wedge B^{-1}e(H_z, C_{\theta'_1})e(H_r, C_{\theta'_5})e(\theta'_6, \theta'_7)e(H_1, C'_{x,t-1})\\ &e(H_2, C'_{T_{t-1}})e(H_3, C'_{N_{t-1}})e(H_4, C'_{P_{t-1}})e(H_5, C'_{R_{t-1}})\\ &= e(H_z, \hat{h})^{r_{\theta'_1}}e(H_r, \hat{h})^{r_{\theta'_5}}e(H_1, \hat{h})^{r'_{x,t-1}}e(H_2, \hat{h})^{r'_{T_{t-1}}}\\ &\quad e(H_3, \hat{h})^{r'_{N_{t-1}}}e(H_4, \hat{h})^{R'_{t-1}}e(H_5, \hat{h})^{r'_{R_{t-1}}}\end{aligned} \tag{2}$$

$$\begin{aligned}&\wedge e(acc, C'_{P_{t-1}})e(g, C_W)^{-1}\\ &= e(acc, \hat{h})^{R_{t-1}+R'_{t-1}}e(g, \hat{h})^{-r_W}e(g_1, g_n)^S\end{aligned} \tag{3}$$

$$\begin{aligned}&\wedge\hat{A}^{-1}\ e(\hat{G}_z, C_{\Theta'_1})e(\hat{G}_r, C_{\Theta'_2})\ e(\Theta'_3, \Theta'_4)\\ &\quad = e(\hat{G}_z, \hat{h})^{r_{\Theta'_1}}e(\hat{G}_r, \hat{h})^{r_{\Theta'_2}}e(\hat{G}_1, g)^{-S}\end{aligned} \tag{4}$$

$$\begin{aligned}&\wedge\hat{B}^{-1}e(\hat{H}_z, C_{\Theta'_1})e(\hat{H}_r, C_{\Theta'_5})e(\Theta'_6, \Theta'_7)\\ &= e(\hat{H}_z, \hat{h})^{r_{\Theta'_1}}e(\hat{H}_r, \hat{h})^{r_{\Theta'_5}}e(\hat{H}_1, g)^{-S}\}(m)\end{aligned} \tag{5}$$

(10) SP: 送られた T_{t-1} が過去に使用されていたものだった場合は認証失敗とする。 使用されていなければ過去に SP へ送られた T_{t-1} の集合に加える。

(11) SP: このユーザが使用するセッション ID i を選択する。このユーザのアキュームレータにセッション ID を加えるために $C_{P_t} = C_{P_{t-1}}h_i$ を計算する。これは $P_t = P_{t-1}h_i$ の計算に必要である。 また、$N_t = N_{t-1} + 1$ の計算のために $C_{N_t} = C'_{N_{t-1}}g$ を計算する。

(12) SP: $C_{x,t} = C'_{x,t-1}$ としてメッセージ $(C_{x,t}, C_{T_t}, C_{N_t}, C_{P_t}, C_{R_t})$ に対して AHO 署名 σ_t を作成する。その後、(σ_t, i) をユーザに送付する。

(13) ユーザ: L_U に i を加える。$P_t = P_{t-1}h_i$ を計算する。この計算は $\prod_{i\in L_U} h_i$ と同じである。

(14) ユーザ: ユーザはセッション IDi と以下の新しい証明書を得る。

$cert_t =$

$$(L_U, T_t, N_t, P_t, R_t, C_{x,t}, C_{T_t}, C_{N_t}, C_{P_t}, C_{R_t}, r_{x,t}, r_{T_t}, r_{N_t}, r_{R_t})$$

本方式では、認証回数 N_{t-1} を秘匿しているが、S と同様に、N_{t-1} の範囲証明を行うように拡張可能である。

4.3 安全性

(1) 式を整理すると以下となる。

$$\begin{aligned}A = {} & e(G_z, C_{\theta'_1}\hat{h}^{-r_{\theta_1}})e(G_r, C_{\theta'_2}\hat{h}^{-r_{\theta_2}})e(\theta'_3, \theta'_4)\\ & e(G_1, C'_{x,t-1}\hat{h}^{-r'_{x,t-1}})e(G_2, C'_{T_{t-1}}\hat{h}^{-r'_{T_{t-1}}})\\ & e(G_3, C'_{N_{t-1}}\hat{h}^{-r'_{N_{t-1}}})e(G_4, C'_{P_{t-1}}\hat{h}^{-R'_{t-1}})\\ & e(G_5, C'_{R_{t-1}}\hat{h}^{-r'_{R_{t-1}}})\end{aligned}$$

これを変形して、

$$\begin{aligned}A = {} & e(G_z, \theta'_1)e(G_r, \theta'_2)e(\theta'_3, \theta'_4)e(G_1, C_{x,t-1})e(G_2, C_{T_{t-1}})\\ & e(G_3, C_{N_{t-1}})e(G_4, C_{P_{t-1}})e(G_5, C_{R_{t-1}})\end{aligned}$$

となり、コミットメント $(C_{x,t-1}, C_{T_{t-1}}, C_{N_{t-1}}, C_{P_{t-1}}, C_{R_{t-1}})$ に対する AHO 署名の検証式が得られる。(2) 式も同様である。さらに、(3) 式から、以下となる。

$$\begin{aligned}&e(acc, C_{P_{t-1}}\hat{h}^{-(R_{t-1}+R'_{t-1})})e(g, C_W\hat{h}^{-r_w})^{-1}\\ &= e(g_1, g_n)^S\end{aligned}$$

これを変形して、

$$e(acc, P_{t-1})e(g, W)^{-1} = e(g_1, g_n)^S$$

となり、アキュムレータの検証式が得られる。

次に (4) 式を整理すると以下となる。

$$\hat{A} = e(\hat{G}_z, C_{\Theta'_1}\hat{h}^{-r_{\Theta_1}})e(\hat{G}_r, C_{\Theta'_2}\hat{h}^{-r_{\Theta_2}})e(\Theta'_3, \Theta'_4)e(\hat{G}_1, g^S)$$

$\Theta'_1 = C_{\Theta'_1}\hat{h}^{-r_{\Theta_1}}, \Theta'_2 = C_{\Theta'_2}\hat{h}^{-r_{\Theta_2}}$ とすると

$$\hat{A} = e(\hat{G}_z, \Theta'_1)e(\hat{G}_r, \Theta'_2)e(\Theta'_3, \Theta'_4)e(\hat{G}_1, g^S)$$

となり、g^S に対する AHO 署名の検証式が得られる。(5) 式も同様である。

匿名性: 各情報はコミットメント、SPK によって秘匿された形で送信されるので、ユーザを特定することができない。また、同様に同じユーザの認証かどうかも分からない。さらに、評価点の合計値も範囲しか分からないため、それによる追跡もできない。

偽造不能性: **登録**と**認証**では、AHO 署名である証明書の保持が証明される。 AHO 署名の偽造不能性から未登録

ユーザは認証に失敗する。 またアキュームレータの特性より、そのユーザの総評価点 $S = \sum_{i \in L_U} s_i$ が保証される。そして g^S の AHO 署名の保持も証明され、範囲内の値 j に対し g^j の AHO 署名は発行されてないことから範囲も保証される。

5. 実験結果

5.1 実行環境

以下の性能の PC において提案方式を実装し認証時間を測定した。 この測定結果から提案方式の認証時間の有用性を示す。

- **メモリ:** 8GB
- **CPU:** Intel®Core™i5-6400 CPU@ 2.70GHz × 4
- **OS:** ubuntu 14.04 LTS 32bit
- **ペアリングライブラリ:** PBC -0.5.14

5.2 測定結果

5.2.1 認証時間

サーバのリストサイズに対する従来方式と提案方式の認証時間を図 1 に示す。

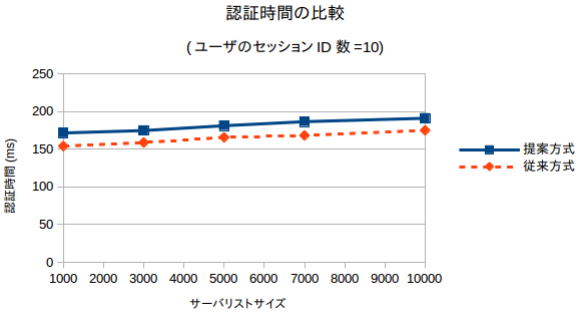

図 1 リストサイズに対する認証時間の変化

範囲証明による認証時間のオーバヘッドは 15ms 程度に収まり、十分実用的である。

5.2.2 鍵サイズの比較

提案方式では、範囲証明のための署名情報が公開鍵ととして必要であるため、公開鍵のサイズが増大してしまう。範囲証明での範囲のサイズに対する範囲証明のための公開鍵サイズの変化を図 2 に示す。

公開鍵サイズは範囲のサイズに依存して増加するが、100 程度の範囲であれば 250KB に収まる。

6. ExpressLane 型への拡張

ExpressLane は”アクティブな”ユーザのみに高速な認証を提供するものであり、[9] により提案されている。本研究の方式にこの手法を適用して $|L_U|$ に依存する補助情報 W の計算時間を軽減することを考える。時間を一定の時間間隔で区切る。サーバのリスト L_s を時間間隔毎に管

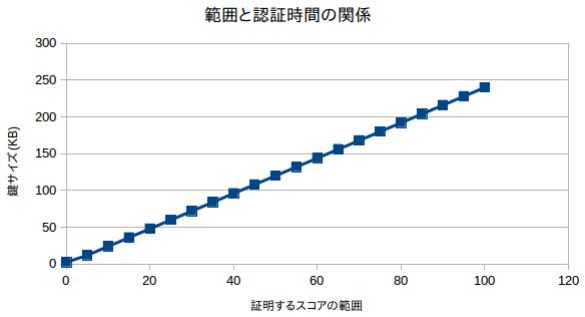

図 2 範囲証明のための公開鍵サイズ

理し、時間間隔 T_t 開始までのリストを L_t とする。時間間隔 T_t 中の現在までのリストを $\delta_t{}^*$ とする。

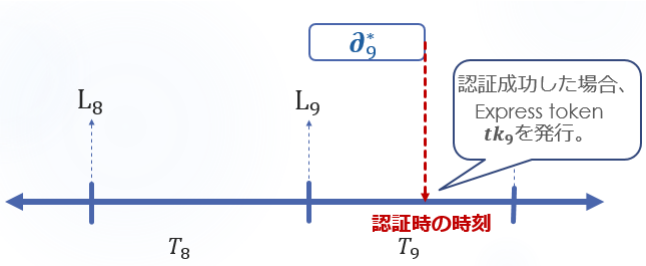

図 3 最初の認証 (Normal Lane)

現在の時間間隔 T_{pd} 内での最初の認証 (図 3) では、T_{pd} 開始時点でのリスト L_{pd} に対しアキュームレータを用いて $S_{pd} = \sum_{i \in L_U} s_i$ を計算する。S_{pd} の範囲を証明し、認証に成功した場合、サーバは Exspress token tk_{pd} をそのユーザに与える。Express token には S_{pd} のコミットが格納される。また、T_{pd} のみの $L_U(\tilde{L}_U$ とする) に対する $C_{p_t}(C_{\tilde{p}_t}$ とする) も格納する。

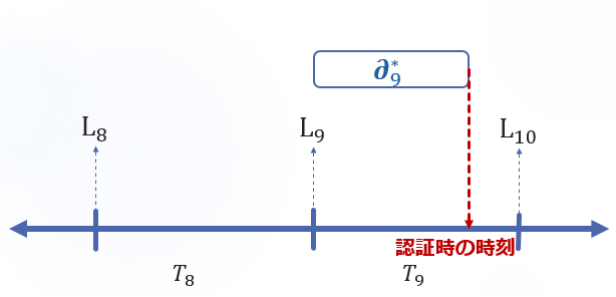

図 4 Express pass 発行後の認証 (Express Lane)

2 回目以降 ExpressLane で認証するユーザ (図 4) は、tk_{pd} と δ_{pd}^* を用いて、認証を行う。このとき、L_{pd} に対するアキュームレータ検証は省き、tk_{pd} 中の S_{pd} を利用する。δ_{pd}^* のアキュームレータ検証には $\tilde{L}_U$ を用いて効率化する。

具体的なアルゴリズムは以下のとおりである。

6.1 アルゴリズム

提案方式での証明書 $cert_t$ に現在の時間 T_{pd} を加えて、署名されるメッセージに g^{pd} も加える。零知識証明では g^{pd} は明らかにして、現時間 T_{pd} で最初の認証であるかを検証する。

6.1.1 Normal Lane

T_{pd} で最初の認証の時、元のプロトコルに加えて、ユーザは以下の計算を行う。

$\tilde{P}_t = h_i$

$C_{\tilde{P}_t} = h_i h^r$

$C_S = h^S h^{r'}$

express token tk_{pd} として $g^{pd}, C_{\tilde{P}_t}, C_S$ の AHO 署名 $\tilde{\sigma}_t$ を発行する。

6.1.2 Express Lane

元プロトコルで、L_S は T_{pd} 中のみのセッション ID をもつ δ^*_{pd} に変更し、C_{P_t} には $C_{\tilde{P}_t}$ を使用する。アキュームレータで検証したスコア $\tilde{S}$ に対して、T_{pd} 以前のスコアである C_S 中の S を加えてから範囲証明を行う。さらに $\tilde{P}_t = \tilde{P}_{t-1} h_i$, $C_{\tilde{P}_t} = C_{\tilde{P}_{t-1}} h_i$ とし、express token tk_{pd} である $(g^{pd}, C_{\tilde{P}_t}, C_S)$ の AHO 署名を発行する。

7. まとめ

本研究では、範囲証明を導入することにより、従来の評価値ベースブラックリスト型匿名認証 [8] でのプライバシー問題を解決した。ユーザはサービス提供者に評価値の合計について具体値を知らせず、評価値の合計がポリシーの範囲内にあることだけを証明できる。範囲証明の追加により認証時間は多少のオーバーヘッドが発生するものの問題ないことを確認した。

さらに、ExpressLane の導入について検討した。一度認証に成功したユーザには ExpressPass を発行し、以前の時間間隔までのスコアに対するアキュームレータ検証を省略することで認証を効率的に行える。

8. 今後の課題

提案方式では 1 種類のスコアのリストしか証明することができない。これを拡張し複数のリストに対して証明可能にすることが課題として挙げられる。

参考文献

[1] P. P. Tsang, M. H. Au, A.Kapadia, S. W. Smith, “Blacklistable Anonymous Credentials: Blocking Misbehaving Users without TTPs”, ACM-CCS2007, pp. 72– 81, 2007.

[2] A. Sudarsono, T. Nakanishi, N. Funabiki, “A Pairing-Based Anonymous Credential System with Efficient Attribute Proofs”, Journal of Information Processing, Vol.20, No.3, pp. 774–784, 2012.

[3] P. P. Tsang, M. H. Au, A. Kapadia, S. W. Smith, “PEREA: Towards Practical TTP-Free Revocation in Anonymous Authentication”, ACM-CCS2008, pp. 333–3334, 2008.

[4] 愛甲 悠, 中西 透, “アキュームレータを用いたブラックリスト型匿名認証システムの認証時間の軽減”, 信学技報, vol. 115, no. 293, ISEC2015-47, pp. 75-80, 2015.

[5] 愛甲 悠, 中西 透, “アキュームレータを用いたブラックリスト型匿名認証システムの改良”, 信学技報, ISEC2016-70, pp. 1–7, 2016.

[6] M. Abe, G. Fuchsbauer, J. Groth, K. Haralambiev, M. Ohkubo, “Structure-preserving signatures and commitments to group elements”, CRYPTO 2010, LNCS 6223, pp. 209–236, 2010.

[7] D. Boneh, X. Boyen, H. Shacham, “Short Group Signatures”, CRYPTO2004, pp. 41–55, 2004.

[8] 金谷 健士, 中西 透, “アキュームレータを用いた評価値ベースのブラックリスト型匿名認証”, 信学技報, vol. 117, no. 369, ISEC2017-81, pp. 59-65, 2017.

[9] Man Ho Au, Apu Kapadia, Willy Susilo, “BLACR: TTP-Free Blacklistable Anonymous Credentials with Reputation”, NDSS Symposium 2012, pp. 1-17, 2012.