社団法人 情報処理学会 研究報告
IPSJ SIG Technical Reports

2007－ＣＳＥＣ－38 (51)
2007／7／20

DDH問題に基づいた高速グループ署名方式

吉田 琢也† 岡田 光司†

† 東芝ソリューション株式会社
〒 183-8512 東京都府中市片町 3-22
E-mail: †{Yoshida.Takuya,Okada.Koji}@toshiba-sol.co.jp

あらまし DDH問題に基づいた高速グループ署名方式を提案する．グループ署名方式は個人情報保護やプライバシ保護に応用可能な技術として注目されているが，従来方式の署名生成計算量はRSA署名の約8倍から200倍以上もの大きな計算量が必要であり，実用化への大きな課題とされていた．本提案では署名生成計算量がRSA署名の3倍程度の高速なグループ署名方式を提案する．また，提案方式の署名用秘密鍵はRSA署名や従来のグループ署名のものに比べて非常に短い．さらに提案方式はRSA modulusを必要としないため楕円曲線上で効率よく実装可能である．
キーワード グループ署名，DDH問題

Efficient Group Signature Scheme based on the DDH Problem

Takuya YOSHIDA† and Koji OKADA†

† Toshiba Solutions Corporation
3-22, Katamachi, Fuchu-Shi, Tokyo 183-8512, Japan
E-mail: †{Yoshida.Takuya,Okada.Koji}@toshiba-sol.co.jp

Abstract We propose an efficient group signature scheme, which is based on the DDH problem. A group signature scheme can be used as a useful tool to protect personal information and privacy. However, its computational efficiency is far less efficient than ordinary signature schemes and thus it is often considered to be impractical. The computational cost of signature generation of our scheme is only 3 times of RSA signature whereas those of the previous schemes are about 8 to more than 200 times and the length of a member secret key of the proposed schemes is much shorter than RSA signature and previous group signature schemes. Moreover, the proposed scheme can be efficiently implemented on elliptic curves.
Key words group signature, DDH problem

1. はじめに

従来の研究

グループ署名は匿名性を持った電子署名としてChaumらによって提案された[8]．通常の電子署名では署名検証のための公開鍵と署名生成のための秘密鍵が1対1で対応するため署名生成者の匿名性が保てない．これに対してグループ署名では署名検証のためのグループ公開鍵と署名生成のためのメンバ秘密鍵が1対多であり，署名検証の際にはどのメンバが署名を生成したかを特定できないという性質によって匿名性を実現している．また，特権者であるグループ管理者だけは署名者を特定できるという性質も持っている．

しかし当初の方式は，署名長や署名生成計算量がメンバ数に比例するためメンバ数が多いグループでの運用に適さないなど，非常に効率が悪かった．これを大幅に改善したのがCamenischらによって提案された方式[6]であり，効率がメンバ数に依存しないグループ署名が実現された．この方式では，メンバ秘密鍵に対するグループ管理者の署名をメンバ証明書（Membership Certificate）として利用する．グループ署名にはグループ管理者の公開鍵で暗号化されたメンバ証明書（またはその一部）と，メンバ証明書が正しく暗号化されていることとメンバ秘密鍵とメンバ証明書を持っていることの非対話的知識証明が含まれる．署名検証では非対話的知識証明を検証することでグループ管理者に認められたメンバによる署名であることを検証でき，グループ管理者はメンバ証明書を復号することで署名者を特定できる．メンバ証明書ベースとも言えるこの概念はその後のグループ署名方式の基礎となっている重要なものである．

しかし[6]はメンバ数に依存しない方式であるとはいえ非常に効率が悪く，とても実用的な方式とは言い難かった．初めて効率的（Practical）という言葉が使われて広く認められたグルー

プ署名方式は Ateniese らによって提案された方式（[ACJT00] 方式）[1] である．この方式はそれ以前の方式と比べて大幅な効率化に成功し，机上の理論であったグループ署名方式を実用化への可能性を検討できるだけの技術とした．

[ACJT00] 方式の署名生成には RSA 署名生成の 200 倍程度の計算量が必要であるためその後も継続的に改良方式の検討が行われており，現在広く知られている高速なグループ署名方式は Camenisch らによって提案された方式（[CG04] 方式）[5] である．この方式の署名生成計算量は RSA 署名生成の 8 倍程度にまで低減されている．

一方，RSA 問題や離散対数問題などの標準的な計算量的仮定に基づいたこれらのグループ署名方式とは異なるアプローチとして，近年は署名長の削減，計算量の削減，効率のよいリボケーションなどを実現するために，双線形写像を利用し双線形群の計算量的仮定に基づいたグループ署名も提案されている．代表的な研究としては [3], [4], [10] などが挙げられる．

本研究の成果

本研究では，いくつかの制約はあるが極めて高速なグループ署名方式を提案する．

提案方式の制約の 1 つはリボケーションに対応していないことである．リボケーションはメンバのグループからの脱退や不正者の排除を行う必要があるモデルにおいては必要な機能であるが，提案方式ではこれに対応していない．

結託に弱いことも提案方式の制約の 1 つである．従来のグループ署名方式ではユーザ同士の結託やグループ管理者も含めた結託に対する安全性が半ば当然のように安全性要件として議論されてきた．しかし，場合によっては結託に対する安全性はシステムの運用やソフトウェアやハードウェアの耐タンパ化技術によって担保できることもあり，結託耐性を低くても高速であることが重視される場合も少なくない．例えば，PC は処理が高速である反面，アプリケーションの不正解析が比較的容易であり結託の危険性が高い．それに対し，携帯電話や IC カードでは処理能力が限られてはいるが，不正解析がより困難であり結託の危険性は低い．このように結託が困難な場合にはこの安全性要件は特に重視されない．

また，提案方式は JOIN と呼ばれるプロトコルを持たない．JOIN プロトコルはメンバ秘密鍵をグループ管理者にも知られずにメンバだけの秘密として生成するためのプロトコルである．しかし，グループ管理者を信頼できるモデルにおいてはグループ管理者がメンバ秘密鍵を生成して配布すればよいため JOIN プロトコルは不要である．

これらの制約の一方で，提案方式は極めて効率がよいことが最大の特長である．高速にべき乗剰余演算を行う手法である Simultaneous Multiple Exponentiation を利用した場合，上述の [CG04] 方式が RSA 署名生成の 8 倍以上の計算量であるのに対し，提案方式では RSA 署名生成のわずか 3 倍程度の計算量で署名生成を行える．また，Simultaneous Multiple Exponentiation では底の値によってテーブルの事前計算を行う必要があるが，提案方式ではべき乗剰余演算の底が常に固定であるため毎回テーブルの事前計算を行う必要がなくテーブルを持っておくことで計算量をさらに若干減らすことができる．

さらに，提案方式は署名生成に利用するメンバ秘密鍵が非常に短く，そのビット長は [CG04] 方式の 1/10，RSA の 1/9 でしかない．

[ACJT00] 方式や [CG04] 方式の安全性が強 RSA 問題に基づいているのに対し，提案方式は DDH 問題に基づいていることも特筆すべき点である．すなわち提案方式は楕円曲線上でも効率よく実装でき，署名長や鍵長の大幅な短縮やある程度の高速化が可能となる．提案方式は DDH 問題のみに基づく効率的なグループ署名として最初の方式である．

さらに，提案方式は単純な演算の組み合わせで実装可能であるため幅広いプラットフォーム上での応用が期待できる．

2. グループ署名

本章ではグループ署名の機能と安全性について定義する．提案方式はリボケーション機能を持たない方式であるため，同様の方式について定義した [2] を拡張して定義する．

2.1 グループ署名の機能

効率のよい既存方式のほとんどはメンバ秘密鍵に対するグループ管理者の署名をメンバ証明書（Membership Certificate）として利用する．提案方式ではグループ管理者の署名を利用しないため，従来方式のメンバ証明書と区別するために署名者特定情報（Tracing Information）という言葉を使う．その値が暗号化してグループ署名に含まれ，署名者特定情報が正しく暗号化されていることとメンバ秘密鍵と署名者特定情報を持っていることの非対話的知識証明が含まれることはメンバ証明書と同様である．

グループ署名方式 GS は以下の 5 つの多項式時間アルゴリズムから成る．

鍵生成

公開パラメータとグループのメンバ数 n を入力とし，グループ公開鍵 gpk，グループ秘密鍵 gmsk，メンバ秘密鍵の集合 $\mathbf{gsk} = (\mathbf{gsk}[1], \ldots, \mathbf{gsk}[n])$，それに対応する署名者特定情報 $\mathbf{T} = (T_1, \ldots, T_n)$ を生成して出力する確率的多項式時間アルゴリズム GKg.

署名生成

グループ公開鍵 gpk，メンバ秘密鍵 $\mathbf{gsk}[i]$，署名者特定情報 T_i，メッセージ msg に対し，グループ署名 σ を生成する確率的多項式時間アルゴリズム GSig.

署名検証

グループ公開鍵 gpk，メッセージ msg，グループ署名 σ を入力とし，署名が正しければ valid を，正しくなければ invalid を出力する確定的多項式時間アルゴリズム GVf.

署名者特定

グループ公開鍵 gpk，グループ秘密鍵 gmsk，メッセージ msg，グループ署名 σ を入力とし，署名が正しければその署名を生成したユーザの ID を，正しくなければ invalid を出力する確定的多項式時間アルゴリズム Open.

2.2 グループ署名の安全性

グループ署名の安全性について当初は数多くの要件が定義さ

れていたが，Static Group，すなわちメンバの追加・削除機能を持たず一度グループが生成されるとメンバの変更がないグループのグループ署名について Bellare らがその要件をまとめている [2]．ただし，[2] の要件は非常に厳しいものであり，グループメンバ全員の結託に対する安全性が考慮されている．そのため，その要件を弱めて安全性を定義することが一般的に行われている．例としては [4] などが挙げられる．ここではその要件に基づき，グループ管理者やメンバの結託がない場合の安全性として再定義する．結託以外の定義については [2], [4] の定義と同様である．

以下の Correctness，Anonymity，Traceability の 3 つの性質を持つとき，グループ署名方式 $\mathcal{GS}$ は安全であるという．

Correctness

$\mathsf{GVf}(\mathsf{gpk}, \mathrm{msg}, \mathsf{GSig}(\mathbf{gsk}[i], \mathrm{msg})) = \mathrm{valid}$ かつ $\mathsf{Open}(\mathsf{gmsk}, \mathrm{msg}, \mathsf{GSig}(\mathbf{gsk}[i], \mathrm{msg})) = i$ すなわち，正しく生成された署名は検証に成功し，署名者を特定できる．

Anonymity

以下のゲームを考える．

（1） Setup: $\mathsf{GKg}(n)$ を実行して gpk, gmsk, $\mathbf{gsk}$, $\mathbf{T}$ を生成し，敵 $\mathcal{A}$ に gpk を与える．

（2） Queries: 敵 $\mathcal{A}$ は以下の 2 種類の質問を行える．ただし Corruption Query は 1 回のみに限られる．

（a） Signing: ユーザ i とメッセージ msg を指定し，$\sigma = \mathsf{GSig}(\mathsf{gpk}, \mathbf{gsk}[i], \mathrm{msg})$ を得る．

（b） Corruption: ユーザ $u (1 \leqq u \leqq n)$ を指定し，$\mathbf{gsk}[u]$ を得る．

（3） Challenge: 敵 $\mathcal{A}$ はメッセージ msg とユーザ ID i_0, i_1 を出力する．このとき $u = i_0$ または $u = i_1$ であってはならない．Challenger は $b \leftarrow \{0, 1\}$ をランダムに選び，$\sigma^* \leftarrow \mathsf{GSig}(\mathsf{gpk}, \mathbf{gsk}[i_b], \mathrm{msg})$ を計算して敵 $\mathcal{A}$ に返す．

（4） Restricted Queries:

（a） Signing: 上記と同様．

（b） Corruption: 上記と同様．ただし既に 1 回行われていた場合は質問できない．また，このとき $u = i_0$ または $u = i_1$ であってはならない．

（5） Output: b' を出力する．

$b' = b$ のとき，敵 $\mathcal{A}$ は攻撃に成功したといい，この成功確率が無視できるときグループ署名方式は Anonymity を持つという．

Traceability

以下のゲームを考える．

（1） Setup: $\mathsf{GKg}(n)$ を実行して gpk, gmsk, $\mathbf{gsk}$, $\mathbf{T}$ を生成し，敵 $\mathcal{A}$ に gpk を与える．

（2） Queries: 敵 $\mathcal{A}$ は以下の 2 種類の質問を行える．ただし Corruption Query は 1 回のみに限られる．

（a） Signing: ユーザ i とメッセージ msg を指定し，$\sigma = \mathsf{GSig}(\mathsf{gpk}, \mathbf{gsk}[i], \mathrm{msg})$ を得る．

（b） Corruption: ユーザ $u (1 \leqq u \leqq n)$ を指定し，$\mathbf{gsk}[u]$ を得る．

（3） Response: 敵 $\mathcal{A}$ はメッセージ msg^* と署名 σ^* を出力する．

$\mathsf{Open}(\mathsf{gmsk}, \mathrm{msg}^*, \sigma^*) = i \neq u$ であり，i, msg^* が Signing Query で質問されていないとき，敵 $\mathcal{A}$ は攻撃に成功したといい，この成功確率が無視できるときグループ署名方式は Traceability を持つという．

3. 準　　備

本章では，提案方式を理解するうえで重要な DDH 問題，representation，Cramer-Shoup 暗号について説明する．

3.1 DDH 問題

G を素数位数 q の乗法巡回群とする．$\mathbb{R}$ はランダムな 4 つ組 $(g_1, g_2, u_1, u_2) \in G^4$ の分布，$\mathbb{D}$ は $g_1, g_2 \in G$ と $r \in \mathbb{Z}_q$ をランダムに選び $u_1 = g_1^r$, $u_2 = g_2^r$ とした 4 つ組 $(g_1, g_2, u_1, u_2) \in G^4$ の分布としたとき，これらの分布を見分ける問題を DDH 問題と呼ぶ．

提案方式の安全性は DDH 問題に帰着される．

3.2 representation

乗法巡回群における演算において $h = g_1^{e_1} g_2^{e_2} \cdots g_l^{e_l}$ を満たす $(e_1, e_2, \ldots, e_l)$ を $g_1, g_2, \ldots, g_l$ を底とした h の representation と呼ぶ．

representation は暗号理論の分野でも古くは [7] で Relaxed Discrete Log (RDL) として利用され，その後もしばしば利用されている．[6] では Schnorr 署名 [12] を応用した representation の非対話的知識証明が利用されている．提案方式ではメンバ秘密鍵に representation を利用し，グループ署名にはメンバ秘密鍵を持っていることを示すための representation の非対話的知識証明が含まれる．

3.3 Cramer-Shoup 暗号

提案方式では，署名者特定情報の暗号化に Cramer-Shoup 暗号を利用する．以下に Cramer-Shoup 暗号を説明する．

公開鍵・秘密鍵ペア生成

公開パラメータとして素数位数 q の乗法巡回群 G とその生成元 g_1 と universal one-way hash family を入力とし，以下の処理を行う．

（1） $g_1, g_2 \in G$ をランダムに選択する．

（2） $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_q$ をランダムに選択する．

（3） $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^z$ を計算する．

（4） $\mathcal{H}$ を universal one-way hash family から選択する．

（5） 公開鍵 $\mathrm{pk} = (g_1, g_2, c, d, h, \mathcal{H})$, 秘密鍵 $\mathrm{sk} = (x_1, x_2, y_1, y_2, z)$ を出力する．

暗号化

公開鍵 $\mathrm{pk} = (g_1, g_2, c, d, h, \mathcal{H})$, メッセージ $m \in G$ を入力とし，以下の処理を行う．

（1） $r \in \mathbb{Z}_q$ をランダムに選択する．

（2） $u_1 = g_1^r$, $u_2 = g_2^r$, $e = h^r m$ を計算する．

（3） $\alpha = \mathcal{H}(u_1, u_2, e)$ を計算する．

（4） $v = c^r d^{r\alpha}$ を計算する．

（5） 暗号文 (u_1, u_2, e, v) を出力する．

復号

暗号文 (u_1, u_2, e, v) を入力とし，以下の処理を行う．

（1） $\alpha = \mathcal{H}(u_1, u_2, e)$ を計算する．

（2） $u_1^{x_1+y_1\alpha}u_2^{x_2+y_2\alpha}=v$ が成り立つかどうかを検証し，一致しなければ不正な暗号文として拒絶して処理を終了する．

（3） $m=e/u_1^z$ を計算し，平文として出力する．

4. 提案方式

本章では我々の提案する高速グループ署名方式の詳細を説明する．

4.1 我々のアプローチ

本研究の目的はグループ署名方式をさらに高速化することである．

高速化のための基本的なアプローチとして離散対数ベースの方式を検討した．RSA ベースの方式はべき指数が長く，位数を知らない群における非対話的知識証明の効率が悪いため全体的な効率が悪くなっていた．効率的な方式として知られている [ACJT00] 方式 [1] や [CG04] 方式 [5] もそこに限界があった．[ACJT00] 方式が RSA ベースの方式であったのに対し，[CG04] は一部を離散対数ベースとしたことで大幅な効率改善を実現しているが，RSA ベースの部分も残っている．これに対し，本研究では全て離散対数ベースの方式とすることで高速化を図っている．

我々はさらに representation をメンバ秘密鍵として利用している．離散対数が 1 つの公開鍵に対して 1 つの秘密鍵しか持てないのに対し，representation は 1 つの公開鍵に対して複数の秘密鍵を作れるためメンバが複数存在するモデルでの利用に適している．Kiayias らの方式 [11] でも representation を利用しているが，署名者特定情報として representation そのものを利用しているため効率のよい方式とはなっていない．しかし署名者の特定に必ずしも representation そのものを必要とはしないため，提案方式では representation から一意に計算される値を署名者特定情報として利用している．

一方で，高速化のために機能的に割り切った部分もある．具体的には，提案方式は結託耐性，JOIN プロトコル，リボケーション機能を持たない．1. 章で述べたとおり，全てのアプリケーションにおいてこれらの機能が必要とされるわけではなく，場合によってはこれらの機能よりも高速であることの方が重視される場合も少なくない．そのため，我々は高速化を最重要課題として方式の検討を行った．

4.2 提案方式の詳細

鍵生成

公開パラメータとして素数位数 q の乗法巡回群 G とその生成元 g_1 と universal one-way hash family を，メンバ数として n を入力とし，以下の処理を行う．

（1） 後述のグループ公開鍵・グループ秘密鍵ペア生成を行う．

（2） 後述のメンバ秘密鍵生成を n 回繰り返す．

（3） グループ公開鍵 gpk，グループ秘密鍵 gmsk，メンバ秘密鍵の集合 $(\mathsf{gsk}[1],\ldots,\mathsf{gsk}[n])$，それに対応する署名者特定情報 $(T_1,\ldots,T_n)$ を出力する．

グループ公開鍵・グループ秘密鍵ペア生成

公開パラメータとして素数位数 q の乗法巡回群 G とその生成元 g_1 と universal one-way hash family を入力とし，以下の処理を行う．

（1） $a,b,x_1,x_2,y_1,y_2,z\in\mathbb{Z}_q$ をランダムに選択する．

（2） $g_2=g_1^a$, $f=g_1^b$, $c=g_1^{x_1}g_2^{x_2}$, $d=g_1^{y_1}g_2^{y_2}$, $h=g_1^z$ を計算する．

（3） $\mathcal{H}$ を universal one-way hash family から選択する．

（4） グループ公開鍵 $\mathrm{gpk}=(g_1,g_2,f,c,d,h,\mathcal{H})$，グループ秘密鍵 $\mathrm{gmsk}=(a,b,x_1,x_2,y_1,y_2,z)$ を出力する．

メンバ秘密鍵生成

グループ公開鍵 $\mathrm{gpk}=(g_1,g_2,f,c,d,h,\mathcal{H})$，グループ秘密鍵 $\mathrm{gmsk}=(a,b,x_1,x_2,y_1,y_2,z)$ を入力とし，以下の処理を行う．

（1） $k_{i2}\in\mathbb{Z}_q$ をランダムに選択する．

（2） $k_{i2}=k_{j2}$ なるメンバ秘密鍵 $\mathsf{gsk}_j=(k_{j1},k_{j2})$ を持つメンバが存在している場合は k_{i2} を選択しなおす．

（3） $k_{i1}=b-ak_{i2} \bmod q$ を計算する．

（4） 署名者特定情報 $T_i=g_1^{k_{i1}}$ を計算する．

（5） ユーザ i のメンバ秘密鍵 $\mathsf{gsk}[i]=(k_{i1},k_{i2})$ と署名者特定情報 T_i を出力する．

署名生成

グループ公開鍵 $\mathrm{gpk}=(g_1,g_2,f,c,d,h,\mathcal{H})$，メンバ秘密鍵 $\mathsf{gsk}[i]=(k_{i1},k_{i2})$，署名者特定情報 T_i，メッセージ $\mathrm{msg}\in\{0,1\}^*$ を入力とし，以下の処理を行う．

（1） $r\in\mathbb{Z}_q$ をランダムに選択する．

（2） $u_1=g_1^r$, $u_2=g_2^r$, $e=h^rT_i$ を計算する．

（3） $\alpha=\mathcal{H}(u_1,u_2,e)$ を計算する．

（4） $v=c^rd^{r\alpha}$ を計算する．

（5） $r_1,r_2,r_r\in\mathbb{Z}_q{}^*$ をランダムに選択する．

（6） $A=g_1^{r_1}g_2^{r_2}$, $B=g_1^{r_r}$, $C=h^{r_r}g_{r_1}$ を計算する．

（7） $\beta=\mathcal{H}(g_1,g_2,h,u_1,u_2,e,v,A,B,C,\mathrm{msg})$ を計算する．

（8） $s_1=r_1+\beta k_{i1} \bmod q$, $s_2=r_2+\beta k_{i2} \bmod q$, $s_r=r_r+\beta r \bmod q$ を計算する．

（9） グループ署名 $\sigma=(u_1,u_2,e,v,A,B,C,s_1,s_2,s_r)$ を出力する．

署名者特定情報 T_i は毎回メンバ秘密鍵から計算してもよい．この場合，署名者特定情報は署名生成の入力には不要となる．

署名検証

グループ公開鍵 $\mathrm{gpk}=(g_1,g_2,f,c,d,h,\mathcal{H})$，メッセージ $\mathrm{msg}\in\{0,1\}^*$，グループ署名 $\sigma=(u_1,u_2,e,v,A,B,C,s_1,s_2,s_r)$ を入力とし，以下の処理を行う．

（1） $\beta=\mathcal{H}(g_1,g_2,h,u_1,u_2,e,v,A,B,C,\mathrm{msg})$ を計算する．

（2） $A=f^{-\beta}g_1^{s_1}g_2^{s_2}$, $B=u_1^{-\beta}g_1^{s_r}$, $C=e^{-\beta}h^{s_r}g_1^{s_1}$ が成り立つかどうかを検証し，成り立てば valid，成り立たなければ invalid を出力する．

署名者特定

グループ公開鍵 $\mathrm{gpk}=(g_1,g_2,f,c,d,h,\mathcal{H})$，グループ秘密鍵 $\mathrm{gmsk}=(a,b,x_1,x_2,y_1,y_2,z)$，メッセージ $\mathrm{msg}\in\{0,1\}^*$，グループ署名 $\sigma=(u_1,u_2,e,v,A,B,C,s_1,s_2,s_r)$ を入力とし，以下の処理を行う．

（1） 上述の署名検証を行い，結果が invalid だった場合は不正なグループ署名として拒絶して処理を終了する．

（2） $\alpha = \mathcal{H}(u_1, u_2, e)$ を計算する．

（3） $u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha} = v$ が成り立つかどうかを検証し，一致しなければ不正なグループ署名として拒絶して処理を終了する．

（4） $T = e/u_1^z$ を計算し，$T_{i'} = T$ なる署名者特定情報を持つユーザ i' を署名者として出力する．

4.3 提案方式の安全性

本節では提案方式の安全性を証明する．

[定理 1] 提案グループ署名方式はランダムオラクルモデルにおいて DDH 問題が困難であるという仮定の下で安全である．

[補題 1] 提案方式は Correctness を持つ．

(証明) 提案方式の定義より明らか． ∎

[補題 2] 提案方式はランダムオラクルモデルにおいて DDH 問題が困難であるという仮定の下で Anonymity を持つ．

(証明スケッチ) 提案方式の Anonymity を無視できない確率で破る敵 $\mathcal{A}^{\mathsf{anon}}$ を利用して DDH 問題を無視できない確率で破る解く敵 $\mathcal{A}^{\mathsf{DDH}}$ を構成する．

$\mathcal{A}^{\mathsf{DDH}}$ への入力を (g_1, g_2, u_1, u_2) とする．

GKg を以下のようにシミュレートする．

- $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_q$ をランダムに選択する．
- $i \in \{1, \ldots, n\}$ をランダムに選択する．
- $k_{i1}, k_{i2} \in \mathbb{Z}_q$ をランダムに選択する．
- $f = g_1^{k_{i1}} g_2^{k_{i2}}$ を計算する．
- $T_i = g_1^{k_{i1}}$ とする．
- $j \in \{1, \ldots, n\} \setminus \{i\}$ に対して $T_j \in G$ をランダムに選択する．
- $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^z$ を計算する．
- $\mathcal{H}$ を universal one-way hash family から選択する．
- グループ公開鍵を $\mathrm{gpk} = (g_1, g_2, f, c, d, h, \mathcal{H})$，ユーザ i のメンバ秘密鍵 $\mathbf{gsk}[i] = (k_{i1}, k_{i2})$ とする．

ユーザ j に対する Corruption Query の応答を以下のようにシミュレートする．

- $j = i$ の場合は $\mathbf{gsk}[i] = (k_{i1}, k_{i2})$ を返し，それ以外のユーザが指定された場合はエラーとして終了する．

ユーザ j，メッセージ msg の署名要求に対し，Signing Query に対する応答を以下のようにシミュレートする．

- 署名者特定情報の暗号文の部分は T_j の Cramer-Shoup 暗号化を行う．
- 非対話的知識証明の部分はランダムオラクルを利用したシミュレーションを行う．これは一般的な手法であるため詳細は省略する．

Challenger を以下のようにシミュレートする．

- $b \in \{0, 1\}$ をランダムに選択する．
- 署名者特定情報の暗号文の部分は Cramer-Shoup 暗号の安全性証明と同様にシミュレートする．詳細は [9] を参照．
- 非対話的知識証明の部分はランダムオラクルを利用したシミュレーションを行う．

$\mathcal{A}^{\mathsf{DDH}}$ は $b = b'$ の場合は 1 を出力し，それ以外の場合は 0 を出力する．

上述のシミュレーションはいずれも正しく行われているため，敵 $\mathcal{A}^{\mathsf{DDH}}$ は DDH 問題を無視できない確率で解く． ∎

[補題 3] 提案方式はランダムオラクルモデルにおいて離散対数問題が困難であるという仮定の下で Traceability を持つ．

(証明スケッチ) 提案方式の Traceability を無視できない確率で破る敵 $\mathcal{A}^{\mathsf{trace}}$ を利用して離散対数問題を無視できない確率で破る解く敵 $\mathcal{A}^{\mathsf{DL}}$ を構成する．

$\mathcal{A}^{\mathsf{DL}}$ への入力を (g_1, f) とする．

GKg を以下のようにシミュレートする．

- $i \in \{1, \ldots, n\}$ をランダムに選択する．
- $k_{i1}, k_{i2} \in \mathbb{Z}_q$ をランダムに選択する．
- $g_2 = (f g_1^{-k_{i1}})^{1/k_{i2}}$ とする．
- $T_i = g_1^{k_{i1}}$ とする．
- $j \in \{1, \ldots, n\} \setminus \{i\}$ に対して $T_j \in G$ をランダムに選択する．
- $x_1, x_2, y_1, y_2, z \in \mathbb{Z}_q$ をランダムに選択する．
- $c = g_1^{x_1} g_2^{x_2}$, $d = g_1^{y_1} g_2^{y_2}$, $h = g_1^z$ を計算する．
- $\mathcal{H}$ を universal one-way hash family から選択する．
- グループ公開鍵を $\mathrm{gpk} = (g_1, g_2, f, c, d, h, \mathcal{H})$，ユーザ i のメンバ秘密鍵 $\mathbf{gsk}[i] = (k_{i1}, k_{i2})$ とする．

Signing Query と Corruption Query の応答を Anonymity の証明と同様にシミュレートする．

rewind によって 2 つの異なる署名
$\sigma = (u_1, u_2, e, v, A, B, C, s_1, s_2, s_r)$ と
$\sigma' = (u_1', u_2', e', v', A', B', C', s_1', s_2', s_r')$ を得る．
$\beta = \mathcal{H}(g_1, g_2, h, u_1, u_2, e, v, A, B, C, \mathrm{msg})$,
$\beta' = \mathcal{H}(g_1, g_2, h, u_1', u_2', e', v', A', B', C', \mathrm{msg})$ とし，$k_1' = (s_1 - s_1')/(\beta - \beta')$, $k_2' = (s_2 - s_2')/(\beta - \beta')$ とすると $f = g_1^{k_1'} g_2^{k_2'}$ である．また，Traceability の定義より $(k_1', k_2') \neq (k_{i1}, k_{i2})$ であるため $g_2 = g_1^{-(k_{i1} - k_1')/(k_{i2} - k_2')}$ が成り立つ．$-(k_{i1} - k_1')/(k_{i2} - k_2') = \gamma$ とすると $log_{g_1} f = k_{i1} + \gamma k_{i2}$ として $\mathcal{A}^{\mathsf{DL}}$ は無視できない確率で離散対数を求められる． ∎

4.4 提案方式の効率

提案方式の効率を評価するために，通常の電子署名である RSA 署名方式の署名生成の計算量を基準にしたときの従来のグループ署名方式および提案方式の計算量およびデータ長を考える．

従来のグループ署名方式としては非常に高速な方式として知られている [CG04] 方式 [5] の基本方式と比較する．この方式は，それ以前に効率のよい方式として広く認知されていた [ACJT00] 方式 [1] と比較しても 26 倍以上も高速であり，また，full paper に追記されているように双線形写像を利用した方式 [3], [10] と比較しても高速である．

まず，計算量の比較方法の考え方は以下にまとめる．

比較する方式の処理の大部分はべき乗剰余演算にかかるため，他の部分は無視してべき乗剰余演算にどれだけかかるかに注目する．

べき乗剰余演算の計算量は (法のビット長)^2 × べき指数のビット長 に比例するため，法のビット長が同じ場合は全体の計

算量はべき指数のビット長の総和に比例する.

また，法の素因数分解を知っている場合は中国人剰余定理（CRT）を利用できるため，計算量は素因数分解を知らない場合に対して RSA modulus（$n = pq$, p, q: prime, $p \approx q$）の場合で 1/4 から 1/3 程度になる．ここでは計算量が 1/4 になるものとして計算量を見積もる.

さらに，べき乗演算の高速処理手法である Simultaneous Multiple Exponentiation を利用することで $\prod_i g_i^{e_i}$ の形の計算は $\max_i(\{e_i\}) = e_j$ なる $g_j^{e_j}$ の計算と同程度で処理が可能である.

比較する際のセキュリティパラメータは，[CG04] 方式の推奨パラメータを利用した場合を基準とする．推奨パラメータは 2048bit の RSA modulus を利用しているため，RSA では同じく 2048bit の RSA modulus を利用する．提案方式で利用する乗法巡回群 G については $\mathbb{Z}_p^*$ と楕円曲線の 2 つを利用する．$\mathbb{Z}_p^*$ では p を 2048bit の素数，q を 224bit の $p-1$ を割り切る素数としたときの位数 q の $\mathbb{Z}_p^*$ の部分群を利用する．この値は FIPS186-3 の Draft [13] でも利用されている値であり，2048bit の RSA modulus と同程度のセキュリティパラメータとみなせる．楕円曲線ではこれと同等のセキュリティパラメータである 224bit の素数から生成される楕円曲線を利用する.

以上を考慮したうえで，RSA 署名方式，[CG04] グループ署名方式，提案グループ署名方式の主な処理の計算量とデータ長を以下の表にまとめる.

	RSA	[CG04]	提案方式	
			$\mathbb{Z}_p*$	楕円
署名生成計算量(注1)	512	4180		1568
署名検証計算量(注1)	(注2)	2310		672
署名鍵長	(注3) 4096	4438	448	448
検証鍵長	(注4) ≈2048	16666	12288	2688
グループ秘密鍵長	—	282	1568	1568
署名長	2048	12426	15008	3808

5. ま と め

本研究では DDH 問題に基づいた高速グループ署名方式を提案した.

提案方式は結託耐性，JOIN プロトコル，リボケーション機能を持たないが非常に高速でデータ長が短い．また，提案方式は DDH 問題に基づく最初の実用的グループ署名方式である.

今後の課題としては提案方式の制約であった結託耐性，JOIN プロトコル，リボケーション機能などを実現することが考えられる.

(注1)：べき指数のビット長の総和．CRT を利用可能な場合は 1/4 で計算.

(注2)：公開鍵 d の長さに依存．一般に小さい.

(注3)：p,q,e を持つ場合.

(注4)：d を小さくした場合.

文 献

[1] G. Ateniese, J. Camenisch, M. Joye, and G. Tsudik, "A practical and provably secure coalition-resistant group signature scheme," In Proc.of CRYPTO 2000, LNCS 1880, pp.255–270, 2002.

[2] M. Bellare, D. Micciancio, and B. Warinschi, "Foundations of Group Signatures: Formal Definitions, Simplified Requirements, and a Construction Based on General Assumptions," In Proc. of EUROCRYPT 2003, LNCS 2656, pp.614–629, 2003.

[3] D. Boneh, X. Boyen, and H. Shacham, "Short group signatures," In Proc. of CRYPTO 2004, LNCS 3152, pp.41–55, 2004.

[4] D. Boneh and H. Shacham, "Group Signatures with Verifier-Local Revocation," In Proc. of the 11'th ACM conference on Computer and Communications Security (CCS), pp.168–177, 2004.

[5] J. Camenisch and J. Groth, "Group Signatures: Better Efficiency and New Theoretical Aspects," Forth Int. Conf. on Security in Communication Networks – SCN 2004, LNCS 3352, 120–133, 2005. full paper は以下の URL から取得可能（2006 年 6 月現在） http://www.brics.dk/ jg/

[6] J. Camenisch and M. Stadler, "Efficient group signature schemes for large groups," In Proc. of CRYPTO '97, LNCS 1294, pp.410–424, 1997.

[7] D. Chaum, J.H. Evertse, and J. van de Graaf, "An improved protocol for demonstrating possession of discrete logarithms and some generalizations," In Proc. of EUROCRYPT '87, LNCS 304, pp.127–141, 1987.

[8] D. Chaum and E. van Heyst, "Group Signatures," In Proc. of EUROCRYPT '91, LNCS 547, pp.257–265, 1991.

[9] R. Cramer and V. Shoup, "A Practical Public Key Cryptosystem Provably Secure against Adaptive Chosen Ciphertext Attack," In Proc. of CRYPTO '98, LNCS 1462, pp.13–25, 1998.

[10] J. Furukawa and H. Imai, "An efficient group signature scheme from bilinear maps," In Proc. of ACISP 2005, LNCS 3574, pp.455–467, 2005.

[11] A. Kiayias and M. Yung, "Extracting Group Signatures from Traitor Tracing Schemes," In Proc. of EUROCRYPT 2003, LNCS 2656, pp.630–648, 2003.

[12] C.P. Schnorr, "Efficient Signature Generation by Smart Cards," Journal of Cryptology, Vol.4, pp.161–174, 1991.

[13] "March 13, 2006: Draft Federal Information Processing Standard (FIPS) 186-3 — Digital Signature Standard (DSS)," http://csrc.nist.gov/publications/drafts.html (2007 年 6 月現在).