推薦論文

OMSP レスポンダ：グループ署名における失効メンバ確認モデル

米沢祥子† 佐古和恵†

グループ署名方式は，グループに所属するメンバが，グループのメンバであることを示せるが個人が特定されない署名を生成できる署名方式である．グループ署名方式ではまた，特別な権限を持つグループ管理者が，署名を生成したメンバを特定する機能を備える．グループ署名は匿名性を有するため，メンバを失効させるのが難しい．本論文ではグループ署名方式に対し，グループメンバが有効であるか失効しているか応答する特別なエンティティ（OMSP レスポンダ）を導入した失効メンバ確認モデルを提案し，本モデルに基づく失効メンバ確認方式を提案する．本方式はグループ署名方式の署名者特定機能を利用しているため，任意のグループ署名方式に対して適用できる．

OMSP Responder: How to Deal with Revoked Members in Group Signatures

Shoko Yonezawa† and Kazue Sako†

A group signature scheme allows a group member to sign on behalf of the group while keeping the signer anonymous. A revocation of a group member in the group signature scheme would be difficult due to its anonymity. In this paper, we propose a revocation functionality of group signatures which include a special entity, the OMSP responder. The OMSP responder answers the status of group members. In this scheme, when a verifier checks the validity of a group signature, a verifier can check the member status of the group member who generated the signature by asking the OMSP responder for the status. We can efficiently add the revocation functionality by introducing the OMSP responder.

1. はじめに

グループ署名方式[1)~8)]は，グループに所属するあるユーザが，グループのメンバであることを示せるがグループの誰であるか特定されない署名を作成することができる方式である．グループ署名方式はさらに，グループ管理者と呼ばれる特別なエンティティが署名から実際の署名者を特定できるという機能を持つ．グループ署名は匿名のままでグループに所属するという資格の確認ができるのでプライバシ保護のためのツールとして利用できる．

グループ署名方式では，グループから脱退したメンバが脱退後に生成した署名が，正当な署名として受理されることを防ぐ機能が望まれる．たとえば，ある会社に所属する社員に対する匿名会員サービスを実施するとき，退職した社員が退職後も引き続きサービスを受けられないようにする機能が必要になる．これを，メンバの失効問題と呼ぶ．本論文では，グループ署名方式のメンバ失効問題を簡単に解決できる失効メンバ確認モデルを提案する．

1.1 従来方式

メンバの失効問題に対し，デジタル署名では CRL（Certificate Revocation List）[10)] と OCSP（Online Certificate Status Protocol）[9)] が規定されている．CRL 方式では，失効したユーザに対応する公開鍵証明書を証明書失効リスト（CRL）に記載する．受け取った署名を確認する検証者は，その時点における CRL を取得し，署名に対応する公開鍵証明書が CRL に記載されているかどうかを確認することによって，署名者が失効しているかどうか確認する．OCSP 方式では，検証者は OCSP レスポンダと呼ばれるオンラインサーバに証明書の状態を問い合わせる．OCSP レスポンダは失効した公開鍵証明書の情報を管理しており，検証者が問い合わせた証明書の状態をこの情報に従っ

† 日本電気株式会社
NEC Corporation

本論文の内容は 2004 年 10 月のコンピュータセキュリティシンポジウム 2004 にて報告され，CSEC 研究会主査により情報処理学会論文誌への掲載が推薦された論文である．

て判定し，検証者に応答する．しかしグループ署名方式では，グループ署名に対応する公開鍵はグループで共通に設定されたグループ公開鍵であるため，CRL や OCSP のような個人に対応する公開鍵証明書を利用したメンバ失効方式を適用できない．

グループ署名方式のメンバ失効問題に対し，アルゴリズムの工夫による実現方式が提案されている[3)~5)]．これらの方式では，署名を生成するメンバが，公開された最新のメンバリスト（またはメンバ失効リスト）に対して，“現在も有効なメンバであること” を示すアプローチがとられている．しかしこれらのアプローチでは，任意のグループ署名アルゴリズムに失効機能を効率良く追加できるものではない．

1.2 本 提 案

本論文では，任意のグループ署名方式に対して簡単に失効メンバ確認機能を追加できるモデルを提案する．これを実現するために，OMSP（Online Membership Status Protocol）レスポンダモデルを提案し，OMSP レスポンダモデルに基づくグループ署名方式の構成方法を提案する．OMSP レスポンダモデルでは，PKI の OCSP レスポンダと同様のオンラインサーバ（OMSP レスポンダ）を新たに導入する．OMSP レスポンダは信頼機関であり，グループ署名を生成したメンバの状態を応答する．本モデルを用いると，失効メンバ確認機能を持たないグループ署名方式に対して，運用により簡単に失効メンバ確認機能を追加することができる．提案する構成方法では，グループ署名方式の署名者特定機能を利用して OMSP レスポンダを構成する．このため，任意のグループ署名方式に適用できる構成方法である．

本論文ではまた，OMSP レスポンダモデルの安全性について考察する．最初に，グループ署名方式に OMSP レスポンダを導入したときの安全性を考察し，このモデルにおけるグループ署名方式の安全性を再定義する．次に，提案する構成方法が再定義したグループ署名方式の安全性を満足することを示す．さらに，提案する構成方法における OMSP レスポンダに対する情報秘匿について考察し，OMSP レスポンダに対していくつかの情報を秘匿したままメンバ状態を正しく判定できる手法を提案する．

本論文の構成は以下のとおりである．まず 2 章で，一般的なグループ署名方式のモデルと安全性を紹介する．次に 3 章で，本論文で提案する OMSP レスポンダを用いた失効メンバ確認モデル（OMSP レスポンダモデル）を定義し，本モデルの安全性を定義する．4 章で，OMSP レスポンダモデルに基づく失効メンバ確認機能つきグループ署名方式の具体的な構成方法を提案する．5 章で OMSP レスポンダに対する情報秘匿の定義およびそれを満たす構成方法を示し，最後に 6 章でまとめを述べる．

2. 一般のグループ署名方式

本章では，OMSP レスポンダモデルの構成で基盤とする一般的なグループ署名方式のモデルおよび安全性の定義を紹介する[2),7)]．

グループ署名方式では 3 つのエンティティ（メンバ，検証者，グループ管理者）が存在する．グループのメンバは，署名を生成するエンティティである．検証者は，生成された署名を検証するエンティティである．グループ管理者は，グループへのメンバ登録や，生成された署名に対する署名者の特定を行う特別な権限を持つエンティティである．グループ管理者の権限を小さくするために，メンバ登録権限を持つ管理者（発行者）と署名者特定権限を持つ管理者（開封者）を，グループ管理者の代わりに設定してもよい．ここでは簡単のため，グループ管理者が単一のモデルを採用する．

定義 1 （グループ署名方式） グループ署名方式 $\mathcal{GS} = (\mathsf{GS\text{-}KeyGen}, \mathsf{GS\text{-}Issue}, \mathsf{GS\text{-}Join}, \mathsf{GS\text{-}Sign}, \mathsf{GS\text{-}Verify}, \mathsf{GS\text{-}Open}, \mathsf{GS\text{-}Identify}, \mathsf{GS\text{-}Judge})$ は，メンバ，グループ管理者，検証者を含み，次の処理から構成される．

- $\mathsf{GS\text{-}KeyGen}(1^k) \rightarrow (gpk, isk, osk)$.
  グループ管理者の実行するグループ鍵生成処理．セキュリティパラメータ k を入力として，グループ公開鍵 gpk，発行用秘密鍵 isk，開封用秘密鍵 osk を出力する．gpk を公開し，(isk, osk) を秘密に保持する．
- $\langle \mathsf{GS\text{-}Issue}(isk), \mathsf{GS\text{-}Join}() \rangle (gpk, i, ID) \rightarrow \langle \mathcal{L}[i], (cert_i, sk_i) \rangle$.
  グループ管理者と i 番目のメンバの実行する署名鍵生成処理．共通入力はグループ公開鍵 gpk，メンバ番号 i，およびメンバ識別子 ID である．グループ管理者は他に isk を秘密に入力する．実行後，グループ管理者は i 番目のメンバのメンバ登録証明書 $cert_i$ とメンバ識別子 ID の対を，メンバリスト $\mathcal{L}$ の i 番目のエントリ $\mathcal{L}[i] = \langle cert_i, ID \rangle$ として出力する．メンバは，メンバ登録証明書 $cert_i$ と署名鍵 sk_i を出力する．
- $\mathsf{GS\text{-}Sign}(m, gpk, cert_i, sk_i) \rightarrow \sigma$.
  メンバの実行する署名生成処理．メッセージ m，グループ公開鍵 gpk，メンバ登録証明書 $cert_i$，署名鍵 sk_i を入力し，グループ署名 σ を出力する．

- GS-Verify$(m, \sigma, gpk) \rightarrow$ accept/reject.
  検証者の実行する署名検証処理．検証者はメッセージ m，グループ署名 σ，グループ公開鍵 gpk を受け取り，accept または reject を出力する．
- GS-Open$(m, \sigma, gpk, osk) \rightarrow (cert', \tau)$.
  グループ管理者の実行する署名開封処理．メッセージ m とそれに対する署名 σ，グループ公開鍵 gpk，開封用秘密鍵 osk を入力し，メンバ登録証明書 $cert'$ と開封証明 τ を出力する．
- GS-Identify$(cert', \mathcal{L}) \rightarrow ID'/\perp$
  グループ管理者の実行する署名者特定処理．署名開封処理で出力されたメンバ登録証明書 $cert'$ とメンバリスト $\mathcal{L}$ を入力し，署名を生成したメンバに対応する ID' を出力する．該当するメンバがいない場合は，$\perp$ を出力する．
- GS-Judge$(m, \sigma, gpk, cert', \tau) \rightarrow$ accept/reject.
  開封検証処理．メッセージ m とそれに対する署名 σ，グループ公開鍵 gpk，メンバ登録証明書 $cert'$，開封証明 τ を入力し，accept または reject を出力する．

グループ署名方式の安全性は次の 4 つで表せる[2]．

正当性（correctness） GS-Sign によって生成された署名は，GS-Verify で accept を出力される．また同時に，GS-Open で出力されたメンバ証明書と開封証明に対して，メンバ証明書を入力とした GS-Identify は正しい署名者に対応するメンバ識別子を出力し，開封証明を入力とした GS-Judge は accept を出力する．

匿名性（anonymity） 攻撃者は，2 人のメンバのうちどちらかが生成した署名を与えられたとき，どちらが署名を生成したか識別できない．

追跡可能性（traceability） 攻撃者は，検証を通過するが，GS-Open の出力から計算した GS-Identify の出力が $\perp$ かつ GS-Judge の出力が accept になるような署名を生成できない．

捏造不可能性（non-frameability） 攻撃者はメンバの一部と結託しても，検証を通過し，かつ GS-Open の出力から計算した GS-Identify の出力が攻撃者と結託していないメンバとなるような署名を生成できない．

それぞれの要件の詳細な定義は付録 A.1 を参照されたい．

3. OMSP レスポンダモデル

本章では，グループ署名を生成した署名者のメンバ状態を応答する OMSP（Online Membership Status Protocol）レスポンダと，これに基づく失効メンバ確認モデル（OMSP レスポンダモデル）を提案する．また，OMSP レスポンダモデルに基づく失効メンバ確認機能つきグループ署名方式を定義し，その満たすべき安全性を議論する．

3.1 概　　要

OMSP レスポンダモデルは，グループ署名方式に OMSP レスポンダを新たに組み込み，元のグループ署名方式に失効メンバ確認機能を追加するモデルである．OMSP レスポンダは，署名を生成したメンバの状態を検証者の代わりに判定する信頼機関であり，検証者からのメンバ状態問合せに対し，メンバ状態を判定して結果を応答する．OMSP レスポンダを導入することにより，メンバ失効機能を持たないグループ署名方式に対して簡単にメンバ失効機能を追加できる．

OMSP レスポンダモデルでは 4 つのエンティティが存在する（**図 1** 参照）．**メンバ**は，グループ管理者を通じてグループに加入し，署名を作成する．**グループ管理者**は，システムのパラメータを設定し，メンバの追加・削除とグループ署名の署名者特定を行う．グループ管理者はメンバを追加する際，新しいメンバに署名を生成するための情報を発行し，メンバを削除する際，失効したメンバの情報を OMSP レスポンダに通知する．**検証者**は，受信した署名を検証する．**OMSP レスポンダ**は，グループ管理者から失効したメンバの情報を受け取り，その情報を失効情報リストとして保存し，検証者からの問合せに対して失効情報リストに基づいて判定し，有効または失効の判定結果を検証者に応答する．

3.2 定　　義

次に，OMSP レスポンダモデルに基づく失効メンバ確認機能つきグループ署名方式（以下，OMSP レスポンダ方式と呼ぶ）を定義する．OMSP レスポンダ方式は，一般のグループ署名方式に対して以下の変更が加えられている．

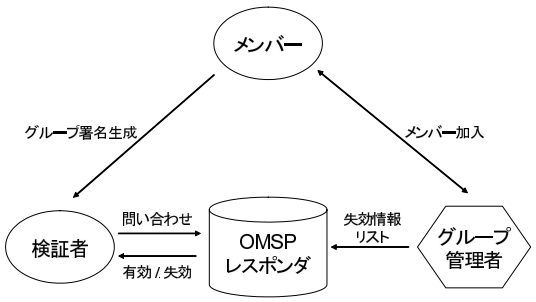

図 1 OMSP レスポンダモデル
Fig. 1 OMSP responder model.

(1) OMSP レスポンダの初期化処理を追加（OMSP-Init）
(2) メンバ失効処理を追加（OMSP-Revoke）
(3) 署名検証処理を OMSP レスポンダと検証者との間で行われるプロトコルに変更（⟨OMSP-Verify, OMSP-Check⟩）

定義 2 （OMSP レスポンダ方式） OMSP レスポンダモデルに基づくグループ署名方式 $\mathcal{GS}^{\mathcal{OMSP}}$ = (OMSP- KeyGen, OMSP- Init, OMSP- Issue, OMSP-Join, OMSP- Revoke, OMSP- Sign, OMSP-Verify, OMSP-Check, OMSP-Open, OMSP-Identify, OMSP-Judge) は，メンバ，グループ管理者，検証者，OMSP レスポンダを含み，次の処理から構成される．

- OMSP-KeyGen$(1^k) \to (gpk, isk, osk)$.
  グループ管理者の実行するグループ鍵生成処理．セキュリティパラメータ k を入力として，グループ公開鍵 gpk，発行用秘密鍵 isk，開封用秘密鍵 osk を出力する．
- OMSP-Init(rsk).
  OMSP レスポンダの初期化処理．OMSP レスポンダにレスポンダ用秘密鍵 rsk を設定する．レスポンダ用秘密鍵 rsk はグループ管理者の開封用秘密鍵 osk から生成される．
- ⟨OMSP-Issue(isk), OMSP-Join$()$⟩$(gpk, i, ID) \to$ ⟨$\mathcal{L}[i]$, $(cert_i, sk_i)$⟩.
  グループ管理者と i 番目のメンバの実行する署名鍵生成処理．共通入力はグループ公開鍵 gpk，メンバ番号 i，およびメンバ識別子 ID である．グループ管理者はほかに isk を秘密に入力する．実行後，グループ管理者はメンバ i のメンバ登録証明書 $cert_i$ とメンバ識別子 ID の対を，メンバリスト $\mathcal{L}$ の i 番目のエントリ $\mathcal{L}[i] = \langle cert_i, ID\rangle$ として出力する．メンバは，メンバ登録証明書 $cert_i$ と署名鍵 sk_i を出力する．
- OMSP-Revoke$(\mathcal{L}, ID) \to rvinfo$.
  グループ管理者の実行するメンバ失効処理．グループ管理者は，メンバリスト $\mathcal{L}$ と失効させるメンバのメンバ識別子 ID から，失効用情報 $rvinfo$ を出力する．グループ管理者は $rvinfo$ を OMSP レスポンダに通知し，OMSP レスポンダは，$rvinfo$ を失効情報リスト $\mathcal{RL}$ に追加する．
- OMSP-Sign$(m, gpk, cert_i, sk_i) \to \sigma$.
  メンバの実行する署名生成処理．メッセージ m，グループ公開鍵 gpk，メンバ登録証明書 $cert_i$，署名鍵 sk_i を入力し，グループ署名 σ を出力する．
- ⟨OMSP-Verify(m, σ), OMSP-Check$(rsk, \mathcal{RL})$⟩$(gpk) \to$ ⟨accept/reject, $\cdot$⟩.
  検証者と OMSP レスポンダの実行する署名検証処理．グループ公開鍵 gpk を共通入力とする．検証者の秘密入力はメッセージ m と署名 σ であり，OMSP レスポンダの秘密入力はレスポンダ用秘密鍵 rsk と失効情報リスト $\mathcal{RL}$ である．実行後，検証者は accept または reject を出力する．
- OMSP-Open$(m, \sigma, gpk, osk) \to (cert', \tau)$.
  グループ管理者の実行する署名開封処理．メッセージ m とそれに対する署名 σ，グループ公開鍵 gpk，開封用秘密鍵 osk を入力し，メンバ登録証明書 $cert'$ と開封証明 τ を出力する．
- OMSP-Identify$(cert', \mathcal{L}) \to ID'/\perp$.
  グループ管理者の実行する署名者特定処理．署名開封処理で出力されたメンバ登録証明書 $cert'$ とメンバリスト $\mathcal{L}$ を入力し，署名を生成したメンバに対応する値 ID' を出力する．該当するメンバがいない場合は，$\perp$ を出力する．
- OMSP-Judge$(m, \sigma, gpk, cert', \tau) \to$ accept/reject.
  開封検証処理．メッセージ m とそれに対する署名 σ，グループ公開鍵 gpk，メンバ登録証明書 $cert'$，開封証明 τ を入力し，accept または reject を出力する．

3.3 安全性の定義

本章では，OMSP レスポンダを導入することによって，グループ署名方式の安全性定義（正当性，匿名性，追跡可能性，捏造不可能性）がどのように変わるか考察し，安全性を再定義する．

匿名性では，OMSP レスポンダへの問合せによってメンバの状態に関する情報を得られるので，それ以上の情報が漏れていないことを保証するための定式化を与える．追跡可能性と捏造不可能性では，攻撃者が正しく動作する OMSP レスポンダを利用した場合でも安全性が保証できるような定式化を与える．なお，OMSP レスポンダは信頼機関であるため，攻撃者と結託することは想定しない．

正当性（correctness） OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ において，OMSP-KeyGen で作られたすべての (gpk, isk, osk)，⟨OMSP-Issue, OMSP-Join⟩ で作られたすべての $\mathcal{L}$ および $(cert_i, sk_i)$ $(1 \le i \le n)$，すべての $rvinfo$ = OMSP-Revoke$(\mathcal{L}, ID)$，すべての $m \in \{0,1\}^*$ に対して，以下が成り立つとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は正当（correct）であるという．

(1) $rvinfo \notin \mathcal{RL} \iff$ ⟨OMSP-Verify$(m,$ OMSP-Sign$(m,\ gpk,\ cert_i,\ sk_i))$, OMSP-Check$(rsk,$

$\mathcal{RL}$)$\rangle(gpk) \rightarrow \langle$accept, $\cdot\rangle$.

(2) $\sigma :=$OMSP-Sign$(m, gpk, cert_i, sk_i)$ に対して，$(cert', \tau) :=$ OMSP-Open (m, σ, gpk, osk) が
- OMSP-Identify$(cert', \mathcal{L}) = ID$ かつ
- OMSP-Judge$(m, \sigma, cert', \tau) =$ accept

を満たす．

匿名性（anonymity） OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ の匿名性に対する攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ はグループ管理者の発行用公開鍵を与えられ，メンバの秘密鍵を自由に生成できる．$\mathcal{A}$ はまた，OMSP レスポンダを自由に利用できる．さらに $\mathcal{A}$ は，メンバを自由に失効することができる．$\mathcal{A}$ は攻撃対象として失効されていないメンバを 2 人選び，どちらか一方が生成した署名をチャレンジとして与えられ，どちらが署名を生成したか推測する．署名を生成したメンバを識別できたとき，$\mathcal{A}$ の攻撃は成功であると定義する☆．以上をふまえて，$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{anon}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A})$

(1) ゲームの実行者は OMSP-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, isk) を $\mathcal{A}$ に渡す．また，OMSP-Init(rsk) を実行して OMSP レスポンダに秘密鍵 rsk を設定する．メンバリスト $\mathcal{L}$ と失効情報リスト $\mathcal{RL}$ を初期化する．

(2) 攻撃者 $\mathcal{A}$ は次のオラクルを利用することができる．

メンバ追加オラクル： $\mathcal{A}$ は，自ら OMSP-Join および OMSP-Issue を実行して生成した $\langle cert_i, ID\rangle$ の，メンバリスト $\mathcal{L}$ への追加を要求する．ゲームの実行者は $\mathcal{L}[i] = \langle cert_i, ID\rangle$ を追加する．

メンバ失効オラクル： $\mathcal{A}$ はメンバ識別子 ID に対応するメンバの失効を要求する．ゲームの実行者は，$rvinfo =$ OMSP-Revoke$(\mathcal{L}, ID)$ を失効情報リスト $\mathcal{RL}$ に追加する．

開封オラクル： $\mathcal{A}$ は署名 (m, σ) の開封を要求する．ゲームの実行者は，$(cert', \tau) =$ OMSP-Open(m, σ, gpk, osk) を計算し，$(cert', \tau)$ を応答する．

OMSP レスポンダ： $\mathcal{A}$ は署名 (m, σ) の有効性を確認する．$\mathcal{A}$ とゲームの実行者は，$\langle$OMSP-Verify(m, σ), OMSP-Check$(rsk, \mathcal{RL})\rangle(gpk)$ を実行する．

(3) 攻撃者 $\mathcal{A}$ は，メッセージ m^* と，$\mathcal{RL}$ に含まれないメンバ i_0，i_1 に対応するメンバ登録証明書と署名鍵の対 $(cert_{i_0}, sk_{i_0})$，$(cert_{i_1}, sk_{i_1})$ を選ぶ．ゲームの実行者は，メンバ i_b $(b \in \{0, 1\})$ の m^* に対する署名 $\sigma^* =$ OMSP-Sign$(m^*, gpk, cert_{i_b}, sk_{i_b})$ を計算する．σ^* をチャレンジとして $\mathcal{A}$ に渡す．

(4) チャレンジを受け取った後，攻撃者 $\mathcal{A}$ は引き続き (2) で示したオラクルを利用できるが，メンバ失効オラクルと開封オラクルに対する問合せが以下のように制限される．

メンバ失効オラクル： 同様に利用できるが，i_0 または i_1 を失効させることはできない．

開封オラクル： 同様に利用できるが，チャレンジ (m^*, σ^*) を問い合わせることはできない．

(5) 攻撃者 $\mathcal{A}$ は b の推測値 b' を出力する．

(6) $b = b'$ ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k) = \left|\Pr[\mathbf{Game}^{\mathrm{anon}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A}) = 1] - 1/2\right|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対し，$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は匿名（anonymous）である．

追跡可能性（traceability） OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ の追跡可能性を破る攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ はグループ管理者の開封用秘密鍵を与えられ，署名を自由に開封できる．$\mathcal{A}$ はさらに，OMSP レスポンダの利用とメンバの自由な失効を許されている．$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{trace}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A})$

(1) ゲームの実行者は OMSP-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, osk) を $\mathcal{A}$ に渡す．また，OMSP-Init(rsk) を実行して OMSP レスポンダに秘密鍵 rsk を設定する．メンバリスト $\mathcal{L}$ と失効情報リスト $\mathcal{RL}$ を初期化する．

(2) 攻撃者 $\mathcal{A}$ は次のオラクルを利用することができる．

鍵発行オラクル： $\mathcal{A}$ はメンバ i のグループへの追加を要求する．ゲームの実行者は OMSP-Issue を，$\mathcal{A}$ は OMSP-Join を実行する．$\mathcal{A}$ は $(cert_i, sk_i)$ を取得する．メンバリスト $\mathcal{L}$ には $\mathcal{L}[i] = \langle cert_i, ID\rangle$ が追加される．

メンバ失効オラクル： $\mathcal{A}$ はメンバ識別子 ID に対応するメンバの失効を要求する．ゲームの実行者は，$rvinfo =$ OMSP-Revoke$(\mathcal{L}, ID)$

☆ OMSP レスポンダを利用すると，有効なメンバと失効されているメンバを簡単に識別できてしまう．そこで，それ以外の情報が漏れないことを安全性の定義とする．

を失効情報リスト $\mathcal{RL}$ に追加する．

OMSP レスポンダ：$\mathcal{A}$ は署名 (m, σ) の有効性を確認する．$\mathcal{A}$ とゲームの実行者は，$\langle$OMSP-Verify(m, σ), OMSP-Check$(rsk, \mathcal{RL})\rangle(gpk)$ を実行する．

(3) 攻撃者 $\mathcal{A}$ は，メッセージと偽造署名の対 (m^*, σ^*) を出力する．

(4) $\langle$OMSP-Verify(m^*, σ^*), OMSP-Check$(rsk, \mathcal{RL})\rangle(gpk) = \langle$accept, $\cdot\rangle$ かつ，$(cert^*, \tau^*) =$ OMSP-Open$(m^*, \sigma^*, gpk, osk)$ に対して OMSP-Identify $(cert^*, \mathcal{L}) = \perp$ もしくは OMSP-Judge$(m^*, \sigma^*, gpk, cert^*, \tau^*) =$ reject ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{trace}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k) = |\Pr[\mathbf{Game}^{\mathrm{trace}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A}) = 1]|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対し，$\mathbf{Adv}^{\mathrm{trace}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は追跡可能（traceable）である．

捏造不可能性（non-frameability） OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ の捏造不可能性を破る攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ は，メンバの部分集合 $\mathcal{C}$ との結託が許されている．$\mathcal{A}$ にはグループ管理者の発行用秘密鍵と開封用秘密鍵を与えられる．さらに $\mathcal{A}$ は，OMSP レスポンダの利用とメンバの自由な失効が許されている．$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{nf}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A})$

(1) ゲームの実行者は OMSP-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, isk, osk) を $\mathcal{A}$ に渡す．また，OMSP-Init(rsk) を実行して，OMSP レスポンダに秘密鍵 rsk を設定する．メンバリスト $\mathcal{L}$，失効情報リスト $\mathcal{RL}$，および結託メンバ集合 $\mathcal{C}$ を初期化する．

(2) 攻撃者 $\mathcal{A}$ は次のオラクルを利用することができる．

グループ加入オラクル：$\mathcal{A}$ はメンバ i の追加を要求する．ゲームの実行者は OMSP-Join を，$\mathcal{A}$ は OMSP-Issue を実行し，$\mathcal{A}$ は $\mathcal{L}[i] = \langle cert_i, ID\rangle$ を取得する．

メンバ追加オラクル：$\mathcal{A}$ は自ら OMSP-Join および OMSP-Issue を実行して生成した $\langle cert_i, ID\rangle$ の，メンバリスト $\mathcal{L}$ への追加を要求する．ゲームの実行者は $\mathcal{L}[i] = \langle cert_i, ID\rangle$ を追加し，i を $\mathcal{C}$ に追加する．

メンバ失効オラクル：$\mathcal{A}$ はメンバ識別子 ID に対応するメンバの失効を要求する．ゲームの実行者は，$rvinfo =$ OMSP-Revoke$(\mathcal{L}, ID)$ を失効情報リスト $\mathcal{RL}$ に追加する．

署名オラクル：$\mathcal{A}$ はメッセージ m に対するメンバ $i \notin \mathcal{C}$ の署名を要求する．ゲームの実行者は $\sigma =$ OMSP-Sign$(m, gpk, cert_i, sk_i)$ を計算し，σ を応答する．

OMSP レスポンダ：$\mathcal{A}$ は署名 (m, σ) の有効性を確認する．$\mathcal{A}$ とゲームの実行者は，$\langle$OMSP-Verify(m, σ), OMSP-Check$(rsk, \mathcal{RL})\rangle(gpk)$ を実行する．

(3) 攻撃者 $\mathcal{A}$ は，メッセージと偽造署名の対 (m^*, σ^*) を出力する．

(4) $\langle$OMSP-Verify(m^*, σ^*), OMSP-Check$(rsk, \mathcal{RL})\rangle(gpk) = \langle$accept, $\cdot\rangle$ かつ，$(cert^*, \tau^*) =$ OMSP-Open$(m^*, \sigma^*, gpk, osk)$ に対して OMSP-Identify $(cert^*, \mathcal{L}) = i^* \notin \mathcal{C}$ かつ OMSP-Judge$(m^*, \sigma^*, gpk, cert^*, \tau^*) =$ accept ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{nf}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k) = |\Pr[\mathbf{Game}^{\mathrm{nf}}_{\mathcal{GS}^{\mathcal{OMSP}}}(\mathcal{A}) = 1]|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対し，$\mathbf{Adv}^{\mathrm{nf}}_{\mathcal{GS}^{\mathcal{OMSP}},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は捏造不可能（non-frameable）である．

4. OMSP レスポンダモデルに基づくグループ署名方式

本章では，OMSP レスポンダモデルに基づく失効メンバ確認機能つきグループ署名方式の具体的な構成方法を提案する．提案する構成では，OMSP レスポンダに開封用秘密鍵 osk を与え，署名の開封を可能にする．グループ管理者は，失効したメンバに対応するメンバ登録証明書 $cert_i$ を OMSP レスポンダに通知する．メッセージ m と署名 σ を受け取った検証者は，署名を生成したメンバの状態を確認するために，(m, σ) を OMSP レスポンダに送信する．OMSP レスポンダは σ の正当性を検証した後，osk を用いてメンバ登録証明書 $cert'$ を開封し，このメンバ登録証明書が失効されているかどうか調べ，結果を応答する．この構成は 3 章で定義した安全性を満たす．

4.1 基本方式

グループ署名方式 $\mathcal{GS} =$ (GS-KeyGen, GS-Issue, GS-Join, GS-Sign, GS-Verify, GS-Open, GS-Identify, GS-Judge) を用いて，基本方式を次のように構成する．

基本方式

OMSP-KeyGen グループ管理者は $(gpk, isk, osk) =$ GS-KeyGen(1^k) を生成する．

OMSP-Init グループ管理者は，開封用秘密鍵 osk を OMSP レスポンダに送信する．OMSP レスポン

ダは osk を秘密に保存する．

OMSP-Issue, OMSP-Join　グループ管理者と i 番目のメンバは $\langle$GS-Issue(isk), GS-Join()$\rangle$(gpk, i, ID) を実行する．メンバは $\langle cert_i, sk_i\rangle$ を取得し，グループ管理者は $\mathcal{L}[i] = \langle cert_i, ID\rangle$ を取得する．メンバリスト $\mathcal{L}$ はグループ管理者が秘密に保存する．

OMSP-Revoke　グループ管理者は失効用情報として，失効したメンバに対応するメンバ登録証明書 $cert_{i'}$ を OMSP レスポンダに通知する．グループ管理者はメンバリスト $\mathcal{L}$ から，失効したメンバのメンバ識別子 ID を持つ $\mathcal{L}[i'] = \langle cert_{i'}, ID\rangle$ を探索する．続いて，$\mathcal{L}[i']$ から $cert_{i'}$ を取得し，$rvinfo = cert_i$ を OMSP レスポンダに通知する．OMSP レスポンダは $cert_{i'}$ を失効情報リスト $\mathcal{RL} = \{cert_{r_j}\}$ $(j = 1, \ldots, q)$ に追加する（q は失効されたメンバの数）．

OMSP-Sign　メンバ i はメッセージ m に対して，$\sigma =$ GS-Sign(m, gpk, $cert_i$, sk_i) を生成する．

OMSP-Verify, OMSP-Check

(1) メッセージ m と署名 σ を受け取った検証者は，最初に GS-Verify(m, σ, gpk) を実行し，reject ならば reject を出力する．accept ならば，OMSP レスポンダに $req = (m, \sigma)$ を送信する．

(2) OMSP レスポンダは $req = (m, \sigma)$ を受け取ると，GS-Verify(m, σ, gpk) を実行する．reject ならば，unknown を検証者に送信する．accept ならば $cert' =$ GS-Open(m, σ, gpk, osk) を計算する．次に，失効情報リスト $\mathcal{RL} = \{cert_{r_1}, \ldots, cert_{r_q}\}$ に $cert'$ が含まれるかどうか調べる．含まれないならば grant，含まれるなら deny を検証者に送信する．

(3) 検証者は，OMSP レスポンダの判定結果が grant ならば accept を出力する．deny ならば reject を出力する．

OMSP-Open, OMSP-Identify, OMSP-Judge　それぞれ GS-Open, GS-Identify, GS-Judge を実行する．

次に，本構成の安全性について考察する．本構成に対して以下の定理が成り立つ．

定理 1　グループ署名方式 $\mathcal{GS}$ が開封オラクルを用いた攻撃に対して匿名性を満たすならば，基本方式は匿名性を満たす．

証明）基本方式の匿名性を破る攻撃者を $\mathcal{A}$ とする．$\mathcal{A}$ を使ってグループ署名方式 $\mathcal{GS}$ の匿名性を破る攻撃者 $\mathcal{B}$ を構成する．$\mathcal{B}$ は内部で $\mathcal{A}$ を動かすために，$\mathcal{A}$ の環境をシミュレートする必要がある．$\mathcal{A}$ の環境と $\mathcal{B}$ の環境の違いは，OMSP レスポンダとメンバ失効オラクルの有無である．$\mathcal{A}$ の出すその他の問合せに対して，$\mathcal{B}$ は自分の持つ情報やオラクルを用いて応答できる．つまり，$\mathcal{B}$ が OMSP レスポンダとメンバ失効オラクルをその環境でシミュレートできればよいことになる．$\mathcal{B}$ は OMSP レスポンダおよびメンバ失効オラクルを次のようにシミュレートできる．

- メンバ失効オラクル
  $\mathcal{B}$ は $\mathcal{A}$ から失効メンバ j を受け取ると，失効情報リスト $\mathcal{RL}$ に $cert_j$ を追加する．
- OMSP レスポンダ
  $\mathcal{B}$ は $\mathcal{A}$ の OMSP レスポンダへの問合せ (m, σ) を受け取ると，σ を検証する．reject ならば unknown を $\mathcal{A}$ に送信する．accept ならば，(m, σ) をそのまま $\mathcal{B}$ の持つ開封オラクルに送信する．開封オラクルの応答 $cert'$ を受け取ると，$\mathcal{B}$ は $\mathcal{RL}$ に $cert'$ が含まれるかどうか調べる．含まれなかったら grant，含まれたら deny を $\mathcal{A}$ に送信する．

$\mathcal{B}$ はこの環境下で $\mathcal{A}$ を実行する．$\mathcal{A}$ が (m^*, i_0, i_1) を出力したら，$\mathcal{B}$ もこれを出力し，チャレンジ σ^* を受け取る．$\mathcal{B}$ はチャレンジ σ^* を $\mathcal{A}$ に与え，再び $\mathcal{A}$ を実行する．$\mathcal{A}$ が推測値 b' を出力するので，$\mathcal{B}$ も b' を出力する．$\mathcal{B}$ は $\mathcal{A}$ と同じ確率で攻撃に成功する． □

定理 2　グループ署名方式 $\mathcal{GS}$ が追跡可能性を満たすならば，基本方式は追跡可能性を満たす．

証明）基本方式の追跡不可能性を破る攻撃者を $\mathcal{A}$ とする．$\mathcal{A}$ を用いてグループ署名方式の追跡不可能性を破る攻撃者 $\mathcal{B}$ を構成する．定理 1 の証明と同様，$\mathcal{A}$ と $\mathcal{B}$ の環境の差は，メンバ失効オラクルと OMSP レスポンダの有無である．$\mathcal{B}$ の環境でメンバ失効オラクルおよび OMSP レスポンダがシミュレートできれば，$\mathcal{B}$ を使って $\mathcal{A}$ の環境をシミュレートできることになる．

- メンバ失効オラクル
  $\mathcal{B}$ は $\mathcal{A}$ から失効メンバ j を受け取ると，失効情報リスト $\mathcal{RL}$ に $cert_j$ を追加する．
- OMSP レスポンダ
  $\mathcal{B}$ の定義より，$\mathcal{B}$ は開封用秘密鍵 isk を与えられているので，$\mathcal{B}$ は isk を使って OMSP レスポンダをシミュレートすることができる．

この下で $\mathcal{A}$ を実行すると，$\mathcal{A}$ は偽造署名 σ^* を出力する．$\mathcal{B}$ も σ^* を出力すれば，$\mathcal{B}$ は $\mathcal{A}$ と同じ確率で攻撃に成功する． □

定理 3 グループ署名方式 $\mathcal{GS}$ が捏造不可能性を満たすならば，基本方式は捏造不可能性を満たす．

証明）定理 2 と同様に示せる． □

4.2 実 現 例

本節では，具体的なグループ署名方式を用いた OMSP レスポンダ方式の実現方法を示す．

(1)ACJT [1] による構成

Ateniese らによるグループ署名方式[1] を用いた構成方法を示す．各パラメータの詳細な生成方法については上記文献[1] を参照されたい．

OMSP-KeyGen グループ管理者は $gpk = (n, a_0, a, g, h, y = g^x)$，$isk = (p', q')$，$osk = x$ を生成する．

OMSP-Init グループ管理者は OMSP レスポンダに x を送信する．

OMSP-Issue, OMSP-Join プロトコルを通じて，メンバ i は $A_i^{e_i} = a_0 a^{x_i}$ を満たす $cert_i = (A_i, e_i)$ と $sk_i = x_i$ を取得する．グループ管理者は $\mathcal{L}[i] = \langle (A_i, e_i), ID \rangle$ を取得する．

OMSP-Revoke グループ管理者は OMSP レスポンダに，失効したメンバ i' に対応する $A_{i'}$ を通知する．OMSP レスポンダは $A_{i'}$ を $\mathcal{RL}$ に追加する．

OMSP-Sign メンバ i は，メッセージ m に対する署名 $\sigma = (c, s_1, s_2, s_3, s_4, T_1 = A_i y^w \bmod n, T_2 = g^w \bmod n, T_3 = g^{e_i} h^w \bmod n)$（$w$ は乱数）を出力する．

OMSP-Verify, OMSP-Check

(1) (m, σ) を受け取った検証者は，まず σ を検証する．結果が reject ならば reject を出力し，accept ならば $req = (m, \sigma)$ を OMSP レスポンダに送信する．

(2) OMSP レスポンダは $req = (m, \sigma)$ を受け取ると，σ を検証し，reject ならば unknown を検証者に送信する．accept ならば，$A' = T_1 / T_2^x$ を計算し，$\mathcal{RL} = \{A_{r_1}, \ldots, A_{r_q}\}$ に A' が含まれているかどうか調べる．含まれていたら deny，含まれていなかったら grant を検証者に送信する．

(3) 検証者は，OMSP レスポンダからの送信結果が grant ならば accept，deny ならば reject を出力する．

(2)KTY [8] による構成

Kiayias らによる方式[8] を用いた構成方法を示す．各パラメータの詳細な生成方法については上記文献[8] を参照されたい．

OMSP-KeyGen グループ管理者は $gpk = (n, a_0, a, b, g, h, y = g^x)$，$isk = (p', q')$，$osk = x$ を生成する．

OMSP-Init この時点では，OMSP レスポンダには何も設定しない．

OMSP-Issue, OMSP-Join プロトコルを通じて，メンバ i は $A_i^{e_i} = a_0 a^{x_i} b^{z_i}$ を満たす $cert_i = (A_i, e_i, z_i)$ と $sk_i = x_i$ を取得する．グループ管理者は $\mathcal{L}[i] = \langle (A_i, e_i, z_i), ID \rangle$ を取得する．

OMSP-Revoke グループ管理者は OMSP レスポンダに，失効したメンバ i' に対応する $z_{i'}$ を通知する．OMSP レスポンダは $z_{i'}$ を $\mathcal{RL}$ に追加する．

OMSP-Sign メンバ i は，メッセージ m に対する署名 $\sigma = (c, s_1, s_2, s_3, s_4, s_5, T_1, T_2, T_3, T_4, T_5, T_6 = g^{z_i w}, T_7 = g^w)$（$w$ は乱数）を出力する．

OMSP-Verify, OMSP-Check

(1) (m, σ) を受け取った検証者は，σ を検証する．結果が reject ならば reject を出力する．accept ならば，OMSP レスポンダに $req = (m, \sigma)$ を送信する．

(2) OMSP レスポンダは (m, σ) を受け取ると，σ を検証し，reject ならば unknown を検証者に送信する．accept ならば，$\mathcal{RL}$ に含まれる z_{r_j} $(1 \le j \le q)$ に対して，$T_6 = T_7^{z_{r_j}}$ が成り立つかどうか調べる．いずれかの z_{r_j} で成り立つならば deny，すべての z_{r_j} で成り立たなかったら grant を検証者に送信する．

(3) 検証者は，OMSP レスポンダの判定結果が grant ならば accept，deny ならば reject を出力する．

4.3 OMSP レスポンダへの負荷評価

OMSP レスポンダ方式では，署名検証の際に検証者と OMSP レスポンダとの間で通信を必要とする．このシステム構成は IETF で標準化され実運用されている OCSP [9] と同じである．OMSP レスポンダは OCSP と比べ，署名検証処理が大きいうえに，署名開封処理が必要になる．たとえば，現在提案されている中で最速のグループ署名方式[6] では，署名検証処理に楕円曲線上のスカラー倍演算 6 回と 160 bit べき乗剰余演算 5 回とペアリング演算 2 回，署名開封処理に楕円曲線上のスカラー倍演算 2 回の計算が必要となる．しかし，OMSP レスポンダの並列化など負荷分散技術を取り入れることにより，実運用に耐えられると考えられる．

5. OMSP レスポンダに対する情報秘匿

4 章で示した基本方式では，OMSP レスポンダは同一メンバの署名を識別することができる．同一のメンバが生成した複数の署名に対して検証プロトコルを実行したとき，OMSP レスポンダは複数の検証プロトコルから同一のメンバ登録証明書を取得できる．このため，OMSP レスポンダは複数の検証プロトコルが同一メンバの生成した署名を入力としているかどうか識別できる．構成上，本方式では OMSP レスポンダによる同一メンバの識別は避けられない．本方式では OMSP レスポンダを信頼機関と仮定しているものの，他のプライバシに関わる情報を可能な限り知られないようにできるとなおよい．

基本方式を通して OMSP レスポンダが得られる情報には，グループのメンバのうち誰が失効されているか（失効メンバ），問い合わせられた署名をどのメンバが生成したか（署名者），問い合わせられた署名がどのメッセージに対する署名か（メッセージ）がある．これらの情報はメンバのプライバシ情報にあたるので，OMSP レスポンダに対して秘匿されることが望ましい．

そこで本章では，OMSP レスポンダに対する情報秘匿について考察する．次節において OMSP レスポンダに対する上記 3 つの情報（失効メンバ，署名者，メッセージ）の秘匿について定義し，続いて基本方式においてこれらの情報を秘匿するための拡張方式を示す．

5.1 定　　義

失効メンバの秘匿　OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ において，OMSP レスポンダがすべてのメンバに対応する失効用情報 $\{rvinfo_1, \ldots, rvinfo_n\}$（$n$ はメンバ数）から，ある $rvinfo_j = \mathsf{OMSP\text{-}Revoke}(\mathcal{L}, ID)$ $(1 \leq j \leq n)$ に対応する ID を出力できないとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は OMSP レスポンダに対して失効メンバを秘匿する．

署名者の秘匿　OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ において，OMSP レスポンダが検証プロトコルで得た情報から，検証した署名を生成したメンバの識別子 ID を特定できないとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は OMSP レスポンダに対して署名者を秘匿する．

メッセージの秘匿　OMSP レスポンダ方式 $\mathcal{GS}^{\mathcal{OMSP}}$ において，OMSP レスポンダが．メッセージ m_0，m_1 のうちどちらかに対応する検証プロトコルを実行した後，プロトコルから得た情報を用いてどちらのメッセージが検証に使われたか判定できないとき，$\mathcal{GS}^{\mathcal{OMSP}}$ は OMSP レスポンダに対してメッセージを秘匿する．

5.2 情報の秘匿方法

5.2.1 失効メンバの秘匿

基本方式で，$\mathcal{L}[i] = \langle cert_i, ID \rangle$ を知らなければメンバ登録証明書 $cert_i$ からメンバ識別子 ID を特定することができないと仮定する．すると，基本方式は失効メンバを秘匿できる．OMSP レスポンダは OMSP-Revoke で失効したメンバのメンバ登録証明書 $cert'$ を知ることができる．しかし，メンバリスト $\mathcal{L}$ を知らなければ $cert'$ に対応する ID が分からないので，OMSP レスポンダは失効メンバを特定できない．

5.2.2 署名者の秘匿

失効メンバの秘匿と同様，基本方式で $\mathcal{L}[i] = \langle cert_i, ID \rangle$ を知らなければメンバ登録証明書 $cert_i$ からメンバ識別子 ID を特定することができないと仮定すると，基本方式は署名者を秘匿できる．OMSP レスポンダは $\langle$OMSP-Verify, OMSP-Check$\rangle$ を通じて，入力に含まれる署名を生成したメンバのメンバ証明書 $cert''$ を知ることができる．しかし，メンバリスト $\mathcal{L}$ を知らなければ $cert'$ に対応する ID が分からないので，OMSP レスポンダは署名者を特定できない．

5.2.3 メッセージの秘匿

メッセージを秘匿するための拡張方式を 2 種類示す．

拡張方式 1 では，メンバはメッセージのハッシュ値を入力として署名を計算し，検証者は OMSP レスポンダに失効確認する際に，メッセージの代わりにハッシュ値を送信する．こうすることにより，OMSP レスポンダに対してメッセージを秘匿することができる．

拡張方式 1

OMSP-Sign　$H : \{0,1\}^* \rightarrow \{0,1\}^{l_H}$ をハッシュ関数とする（l_H はセキュリティパラメータ）．メンバはメッセージ m に対して，乱数 $r \in \{0,1\}^{l_r}$（l_r はセキュリティパラメータ）を選び $h = H(m||r)$ を計算する．メンバは $\sigma = \mathsf{GS\text{-}Sign}(h, gpk, cert_i, sk_i)$ を計算し，(σ, r) を出力する．

OMSP-Verify, OMSP-Check

(1) メッセージ m と署名 (σ, r) を受け取った検証者は GS-Verify$(H(m||r), \sigma, gpk)$ を計算する．結果が reject ならば reject を出力する．accept ならば，検証者は OMSP レスポンダに $req = (h' = H(m||r), \sigma)$ を送信する．

(2) OMSP レスポンダは $req = (h', \sigma)$ を受け取ると GS-Verify (h', σ, gpk) を実行す

る．reject ならば，unknown を出力する．accept ならば，GS-Open を実行して $cert'$ を得る．次に，失効情報リスト $\mathcal{RL}$ に $cert'$ が含まれるかどうか調べる．含まれないならば grant，含まれるなら deny を検証者に送信する．

（3） 検証者は，OMSP レスポンダの判定結果が grant ならば accept，deny ならば reject を出力する．

拡張方式 1 は基本方式と同等の安全性を満たす．さらに，拡張方式 1 に対して次の定理が成り立つ．

定理 4 H をランダムオラクルと仮定すると，拡張方式 1 は OMSP レスポンダに対してメッセージを秘匿できる．

証明）メッセージ秘匿に対する攻撃者を $\mathcal{A}$ とする．ランダムオラクル H は，$\mathcal{A}$ から $H(m||r)$ の問合せを受けて，$h \in_U \{0,1\}^{l_H}$ を選んで h を応答する．$\mathcal{A}$ は攻撃対象のメッセージ m_0，m_1 を選び，チャレンジ $h^* = H(m_b||r^*)$ $(b \in_U \{0,1\}, r^* \in_U \{0,1\}^{l_r})$ を受け取る．$\mathcal{A}$ がメッセージ m_b に対応するすべてのハッシュ値を探索するためには，ランダムオラクルへ 2^{l_r} 回の問合せが必要である．$\mathcal{A}$ のランダムオラクルへの問合せ回数を q_H とすると，h^* が偶然当たる確率は $q_H/2^{l_r}$ である．$\mathcal{A}$ は多項式時間アルゴリズムであるため，問合せ回数は多項式回に制限されている．つまり，有意な確率で m_b を推測できない．また，q_H 回の問合せで $H(\cdot)$ の値が衝突する確率は $O(q_H^2)/2^{l_H}$ であり，この確率も無視できる． □

次に，拡張方式 2 を示す．拡張方式 2 では，メンバは空文に対する署名 σ_1 と，σ_1 の一部を入力としたメッセージ m に対する署名 σ_2 を計算する．σ_2 の計算に σ_1 の一部を利用するのは，σ_1 と σ_2 を関連付けるためである．(m, σ_1, σ_2) を受け取った検証者は，σ_1 を OMSP レスポンダに送信する．OMSP レスポンダはメッセージ m を使わずに σ_1 を検証できるので，OMSP レスポンダに m を送信する必要がない．また，σ_1 を検証することによって，σ_1 が確かにグループのメンバによって生成されたことを確認できる．以下，Ateniese らのグループ署名方式[1] による実現例を示す．

拡張方式 2

OMSP-Sign メンバは空文に対する署名 $(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ を計算する．次に，T_1，T_2，T_3 を変えずに，別の乱数を用いてメッセージ m に対する署名 $(T_1, T_2, T_3, c', s_1', s_2', s_3', s_4')$ を計算する．$(T_1, T_2, T_3, c, c', s_1, s_2, s_3, s_4, s_1', s_2', s_3', s_4')$ を出力する．

OMSP-Verify, OMSP-Check

（1） 署名 $(m, T_1, T_2, T_3, c, c', s_1, s_2, s_3, s_4, s_1', s_2', s_3', s_4')$ を受け取った検証者は，$(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ が空文に対して署名の検証式を満たすかどうか，および $(T_1, T_2, T_3, c', s_1', s_2', s_3', s_4')$ がメッセージ m に対して署名の検証式を満たすかどうか確認する．出力が reject ならば，reject を出力する．両方とも検証式を通過したら，OMSP レスポンダに $(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ を送信する．

（2） OMSP レスポンダは $(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ を受け取ると，$(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ が空文に対して署名の検証式を満たすかどうかを調べる．reject ならば，unknown を検証者に送信する．accept ならば，T_1，T_2 から A' を計算し，$\mathcal{RL} = \{A_{r_1}, \ldots, A_{r_q}\}$ に A' が含まれるかどうか調べ，含まれていたら deny，含まれていなかったら grant を検証者に送信する．

（3） 検証者は，OMSP レスポンダの判定結果が grant ならば accept，deny ならば reject を出力する．

拡張方式 2 は基本方式と同等の安全性を満たす．さらに，拡張方式 2 に対して次の定理が成り立つ．

定理 5 拡張方式 2 は OMSP レスポンダに対してメッセージを秘匿できる．

証明）メッセージの秘匿を破る攻撃者を $\mathcal{A}$ とする．検証プロトコルで $\mathcal{A}$ が得る情報は $(T_1, T_2, T_3, c, s_1, s_2, s_3, s_4)$ である．これは空文に対する署名であるので，メッセージに関する情報を含まない．したがって，$\mathcal{A}$ は 1/2 より有意な確率でメッセージを判別できない．

6. ま と め

本論文では，OMSP レスポンダを導入したグループ署名方式の失効メンバ確認モデル（OMSP レスポンダモデル）を提案し，OMSP レスポンダモデルに基づく失効メンバ確認機能つきグループ署名方式の構成方法を示した．OMSP レスポンダは検証者からの問合せを受け，署名を生成したメンバの状態を判定し，検証者に応答する．OMSP レスポンダモデルを用いると，任意のグループ署名方式に対して簡単に失効メンバ確認機能を追加できる．本論文ではまた，OMSP

レスポンダモデルに基づくグループ署名方式の安全性を考察し，提案する構成が再定義したグループ署名方式の安全性を満たすことを証明した．さらに，OMSP レスポンダに対する情報秘匿を定義し，OMSP レスポンダに対して失効されているメンバ，署名を生成したメンバ，および検証者の受け取ったメッセージを秘匿できる方式を提案した．

参 考 文 献

1) Ateniese, G., Camenisch, J., Joye, M. and Tsudik, G.: A Practical and Provable Secure Coalition-Resistant Group Signature Scheme, *Advances in Cryptology—CRYPTO 2000*, LNCS, Vol.1880, pp.255–270, Springer-Verlag (2000).
2) Bellare, M., Shi, H. and Zhang, C.: Foundations of Group Signatures: The Case of Dynamic Groups, *CT-RSA*, LNCS, Vol.3376, pp.136–153, Springer-Verlag (2005).
3) Boneh, D. and Shacham, H.: Group Signatures with Verifier-Local Revocation, *Proc. 11th ACM conference on Computer and Communications Security* (*CCS 2004*), pp.168–177, ACM Press (2004).
4) Bresson, E. and Stern, J.: Efficient Revocation in Group Signatures, *Proc. PKC 2001*, LNCS, Vol.1992, pp.190–206, Springer-Verlag (2001).
5) Camenisch, J. and Lysyanskaya, A.: Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials, *Advances in Cryptology—CRYPTO 2002*, LNCS, Vol.2442, pp.61–76, Springer-Verlag (2002).
6) Furukawa, J. and Imai, H.: An Efficient Group Signature Scheme from Bilinear Maps, *Proc. ACISP '05*, LNCS, Vol.3574, pp.455–467, Springer-Verlag (2005).
7) Furukawa, J. and Yonezawa, S.: Group Signatures with Separate and Distributed Authorities, *Proc. SCN 2004*, LNCS, Vol.3352, pp.77–90, Springer-Verlag (2005).
8) Kiayias, A., Tsiounis, Y. and Yung, M.: Traceable Signatures, *Advances in Cryptology—EUROCRYPT 2004*, LNCS, Vol.3027, pp.571–589, Springer-Verlag (2004).
9) RFC 2560: X. 509 Internet Public Key Infrastructure Online Certificate Status Protocol—OCSP (1999).
10) RFC 3280: Internet X. 509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile (2002).

付　　録

A.1 グループ署名の安全性

グループ署名方式の安全性は，正当性，匿名性，追跡可能性，捏造不可能性の 4 つで定式化できる．ここでは，メンバの追加を考慮したグループ署名方式の安全性について論じた文献 2)，7) に基づいて安全性を定義する．

正当性（correctness） グループ署名方式 $\mathcal{GS}$ において，GS-KeyGen で作られたすべての (gpk, isk, osk)，$\langle$GS-Issue, GS-Join$\rangle$ で作られたすべての $\mathcal{L}$ および $(cert_i, sk_i)$ $(1 \le i \le n)$，すべての $m \in \{0,1\}^*$ に対して，以下が成り立つとき，$\mathcal{GS}$ は正当（correct）であるという．

(1) GS-Verify$(m,$ GS-Sign $(m, gpk, cert_i, sk_i), gpk) =$ accept
(2) $\sigma :=$GS-Sign$(m, gpk, cert_i, sk_i)$ に対して，$(cert', \tau) :=$ GS-Open(m, σ, gpk, osk) が
- GS-Identify$(cert', \mathcal{L}) = ID$ かつ
- GS-Judge$(m, \sigma, cert', \tau) =$ accept

を満たす．

匿名性（anonymity） 匿名性とは，攻撃者が 2 人のメンバのうちどちらかが生成した署名を与えられたとき，どちらが生成したか識別できない性質である．匿名性を破る攻撃者は，グループ管理者の持つメンバ追加権限を持つことを許されている．攻撃者はまた，任意のメンバと結託することを許されている．匿名性を定式化するために，グループ署名方式 $\mathcal{GS}$ の匿名性に対する攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{anon}}_{\mathcal{GS}}(\mathcal{A})$
(1) ゲームの実行者は GS-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, isk) を $\mathcal{A}$ に渡す．また，メンバリスト $\mathcal{L}$ を初期化する．
(2) 攻撃者 $\mathcal{A}$ は次のオラクルを利用できる．
メンバ追加オラクル： $\mathcal{A}$ は自ら GS-Join および GS-Issue を実行して生成した $\langle cert_j, ID\rangle$ の，メンバリスト $\mathcal{L}$ への追加を要求する．ゲームの実行者は $\mathcal{L}[j] = \langle cert_j, ID\rangle$ を追加する．
開封オラクル： $\mathcal{A}$ は署名 (m, σ) の開封を要求する．ゲームの実行者は，$(cert', \tau) =$ GS-Open(m, σ, gpk, osk) を計算し，$(cert', \tau)$ を応答する．
(3) 攻撃者 $\mathcal{A}$ は，メッセージ m^* と，メンバ i_0,

i_1 に対応するメンバ登録証明書と署名鍵の対 $(cert_{i_0}, sk_{i_0})$，$(cert_{i_1}, sk_{i_1})$ を出力する．ゲームの実行者は，メンバ i_b $(b \in \{0,1\})$ の m^* に対する署名 $\sigma^* = \mathsf{GS\text{-}Sign}(m^*, gpk, cert_{i_b}, sk_{i_b})$ を計算する．σ^* をチャレンジとして $\mathcal{A}$ に渡す．

(4) チャレンジを受け取った後，攻撃者 $\mathcal{A}$ は引き続き (2) で示したオラクルを利用できるが，開封オラクルにチャレンジ (m^*, σ^*) を問い合わせることはできない．

(5) 攻撃者 $\mathcal{A}$ は b の推測値 b' を出力する．

(6) $b = b'$ ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{GS},\mathcal{A}}(k) = \left|\Pr[\ \mathbf{Game}^{\mathrm{anon}}_{\mathcal{GS}}(\mathcal{A}) = 1] - 1/2]\right|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対して，$\mathbf{Adv}^{\mathrm{anon}}_{\mathcal{GS},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}$ は匿名（anonymous）である．

追跡可能性（traceability） 追跡可能性は，攻撃者が，検証を通過するがメンバリストに含まれるメンバが特定されないような署名を偽造できない性質である．攻撃者は，グループ管理者の署名開封権限を持つことを許されている．攻撃者はまた，任意のグループメンバと結託することを許されている．追跡可能性を定式化するために，グループ署名方式 $\mathcal{GS}$ の追跡可能性を破る攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{trace}}_{\mathcal{GS}}(\mathcal{A})$

(1) ゲームの実行者は GS-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, osk) を $\mathcal{A}$ に渡す．また，メンバリスト $\mathcal{L}$ を初期化する．

(2) 攻撃者 $\mathcal{A}$ は次のオラクルに問い合わせることができる．

鍵発行オラクル： $\mathcal{A}$ はメンバ j のグループへの追加を要求する．ゲームの実行者は GS-Issue を，$\mathcal{A}$ は GS-Join を実行する．$\mathcal{A}$ は ($cert_j$, sk_j) を取得する．メンバリスト $\mathcal{L}$ には $\mathcal{L}[j] = \langle cert_j, ID\rangle$ が追加される．

(3) 攻撃者 $\mathcal{A}$ は，メッセージと偽造署名の対 (m^*, σ^*) を出力する．

(4) $\mathsf{GS\text{-}Verify}(m^*, \sigma^*, gpk) = \mathsf{accept}$ かつ，$(cert^*, \tau^*) = \mathsf{GS\text{-}Open}(m^*, \sigma^*, gpk, osk)$ に対して $\mathsf{GS\text{-}Identify}(cert^*, \mathcal{L}) = \perp$ もしくは $\mathsf{GS\text{-}Judge}(m^*, \sigma^*, gpk, cert^*, \tau^*) = \mathsf{reject}$ ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{trace}}_{\mathcal{GS},\mathcal{A}}(k) = |\Pr[\mathbf{Game}^{\mathrm{trace}}_{\mathcal{GS}}(\mathcal{A}) = 1]|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対して，$\mathbf{Adv}^{\mathrm{trace}}_{\mathcal{GS},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}$ は追跡可能（traceable）である．

捏造不可能性（non-frameability） 捏造不可能性は，攻撃者が，検証を通過するがメンバリストのうち攻撃者が結託していないメンバが特定されるような署名を偽造できない性質である．攻撃者は，グループ管理者のすべての権限を持つことを許されている．攻撃者はまた，メンバの部分集合と結託することを許されている．捏造不可能性を定式化するために，グループ方式 $\mathcal{GS}$ の捏造不可能性を破る攻撃者 $\mathcal{A}$ を仮定する．$\mathcal{A}$ に対して次のゲームを考える．

Game $\mathbf{Game}^{\mathrm{nf}}_{\mathcal{GS}}(\mathcal{A})$

(1) ゲームの実行者は GS-KeyGen を実行し，(gpk, isk, osk) を取得する．(gpk, isk, osk) を $\mathcal{A}$ に渡す．また，メンバリスト $\mathcal{L}$ および結託メンバ集合 $\mathcal{C}$ を初期化する．

(2) 攻撃者 $\mathcal{A}$ は次のオラクルに問い合わせることができる．

グループ加入オラクル： $\mathcal{A}$ はメンバ j の追加を要求する．ゲームの実行者は GS-Join を，$\mathcal{A}$ は GS-Issue を実行し，$\mathcal{A}$ は $\mathcal{L}[j] = \langle cert_j, ID\rangle$ を取得する．

メンバ追加オラクル： $\mathcal{A}$ は自ら GS-Join および GS-Issue を実行して生成した $\langle cert_j, ID\rangle$ の，メンバリスト $\mathcal{L}$ への追加を要求する．ゲームの実行者は $\mathcal{L} = \langle cert_j, ID\rangle$ を追加し，j を $\mathcal{C}$ に追加する．

署名オラクル： $\mathcal{A}$ はメッセージ m に対するメンバ $j \notin \mathcal{C}$ の署名を要求する．ゲームの実行者は $\sigma = \mathsf{GS\text{-}Sign}(m, gpk, cert_j, sk_j)$ を計算し，σ を応答する．

(3) 攻撃者 $\mathcal{A}$ は，メッセージと偽造署名の対 (m^*, σ^*) を出力する．

(4) $\mathsf{GS\text{-}Verify}(m^*, \sigma^*, gpk) = \mathsf{accept}$ かつ，$(cert^*, \tau^*) = \mathsf{GS\text{-}Open}(m^*, \sigma^*, gpk, osk)$ に対して $\mathsf{GS\text{-}Identify}(cert^*, \mathcal{L}) = j \notin \mathcal{C}$ かつ $\mathsf{GS\text{-}Judge}(m^*, \sigma^*, gpk, cert^*, \tau^*) = \mathsf{accept}$ ならば，ゲームの実行者は 1 を出力する．

$\mathbf{Adv}^{\mathrm{nf}}_{\mathcal{GS},\mathcal{A}}(k) = |\Pr[\ \mathbf{Game}^{\mathrm{nf}}_{\mathcal{GS}}(\mathcal{A}) = 1]|$ とする．すべての多項式時間アルゴリズム $\mathcal{A}$ に対して，$\mathbf{Adv}^{\mathrm{nf}}_{\mathcal{GS},\mathcal{A}}(k)$ が無視できるほど小さいとき，$\mathcal{GS}$ は捏造不可能（non-frameable）である．

(平成 17 年 4 月 21 日受付)
(平成 17 年 12 月 2 日採録)

推 薦 文

本発表は，グループ署名を実用化する際に問題となりうる失効者リストの管理に対して現実的な解決策を示している．公開鍵証明書における同種の問題に対して提案されている OCSP を，グループ署名に対応して拡張し，安全性を厳密に議論している．グループ署名の実用化に向けての道筋を示しており，本発表の結果に基づく更なるグループ署名研究の進展も期待できる。暗号理論の実用化に関する新規性と完成度を備えた発表であるので，論文として推薦したい．

（コンピュータセキュリティ研究会主査
村山優子）

米沢 祥子

平成 13 年お茶の水女子大学理学部情報科学科卒業．平成 15 年東京大学大学院情報理工学系研究科電子情報学専攻修士課程修了．同年日本電気（株）入社．現在，同社インターネットシステム研究所にて，暗号プロトコル，特にグループ署名技術の研究開発に従事．

佐古 和恵

昭和 61 年京都大学理学部数学科卒業．同年日本電気（株）入社．現在，同社インターネットシステム研究所主席研究員．博士（工学）．電子投票・電子入札・電子抽選等，暗号プロトコルの研究に従事．電子情報通信学会・IACR 会員．