情報処理学会論文誌 Vol.53 No.9 2267–2278 (Sep. 2012)

# 新しい階層表記型アグリゲート署名を用いたコンテンツ引用過程表記手法

稲村 勝樹[1,a)] 岩村 恵市[1]

受付日 2011年11月30日, 採録日 2012年6月1日

**概要**：GDH（Gap-Diffie-Hellman）グループに基づく新しい木構造表記型アグリゲート署名を用いた，コンテンツにおける引用過程を表記する手法を提案する．準備として，署名順序が隣接する署名者間の前後において，後者が自分の署名対象となるメッセージと前者の署名対象となるメッセージの両方に対して署名し，この署名を順次合成していくことで順序付きアグリゲート署名が実現できることを示す．次にこの隣接署名者間の手順を 1 対多に拡張し，その関係を積み重ねることで木構造表記型のアグリゲート署名が構成できることを示し，これを用いてコンテンツの引用過程を表記できることを示す．さらにその実用性を実装検証する．

**キーワード**：消費者生成メディア，電子署名，アグリゲート署名，ペアリング

# An Expression of a Quotation Process in Contents with a New Tree-structure-specified Aggregate Signature Scheme

Inamura Masaki[1,a)] Iwamura Keiichi[1]

Received: November 30, 2011, Accepted: June 1, 2012

***Abstract:*** We propose a copyright expression of a re-use process with quoting using a new tree-structure-specified aggregate signature scheme based on the GDH (Gap-Diffie-Hellman) group. First an order-specified aggregate signature scheme, which repeats that the latter signs his/her message and the former's message for the porpose of verification that this two signer are neighboring, is proposed. Second the order-specified aggregate signature scheme is expanded into a tree-structure-specified aggregate signature scheme, and an expression of a quoting process can be realized with this scheme. In addition, the possibility of its realization is considered by evaluation of throughput of this proposed scheme on the program.

***Keywords:*** Consumer generated media (CGM), Digital signature, Aggregate signature, Pairing

## 1. はじめに

近年，場所や時間の制限にとらわれずに，複数の人がコンテンツファイルを作成・使用する場面が増加している．一例として，インターネットなどを活用して一般消費者がコンテンツを生成していく CGM（Consumer Generated Media：消費者生成メディア）という概念が発生し，YouTube [1] などのように一般消費者が作成したコンテンツを流通させる，あるいは閲覧させることを目的とした CGM サービスが急速に広まってきている．この CGM サービスにおいて，“マッシュアップ” と呼ばれるコンテンツの二次利用，三次利用，... によるコンテンツ作成が行われており，マッシュアップのための表記法を規定したクリエイティブ・コモンズ [2] のような活動も始まっている．今後，マッシュアップにより作成される CGM コンテンツに対しても，コンテンツ利用料を徴収し，著作権者への料金分配が行われるサービスが予定されている．このとき，コンテンツの作成過程（CGM コンテンツへの貢献度）により各著作権者への分配金額が異なることが想定され，コンテンツが引用

[1] 東京理科大学大学院工学研究科
Graduate School of Engineering, Tokyo University of Science, Chiyoda, Tokyo 102–0073, Japan
[a)] minamura@sec.ee.kagu.tus.ac.jp

© 2012 Information Processing Society of Japan

されている順番が重要な情報となってくる．

マッシュアップコンテンツに対する著作権者の権利主張の証拠，あるいはコンテンツの引用過程の表記・管理を行う方式は，梶らによって提案されている [3], [4]．この方式は，W3C によって規定された RDF（Resource Description Framework）[5] の記述方式を拡張し，コンテンツの構成（作成過程など）をメタデータとして詳細に記載できる．この著作権情報が表記されたメタデータの記載内容を基に上記で示した CGM サービスの展開を考えた場合，その記載内容について不正な改ざんや変更がない正当なものであることを保証する仕組みが必須と考えられる．このデーターの内容を保証する手法として電子署名の利用は有効な手段であり，上記の例で示した場面において，複数の署名者による電子署名の作成，およびその署名の検証を効率的に行うことが可能となる多重署名・アグリゲート署名方式 [6], [7] の適用は検討に値する．多重署名方式とは複数の署名者が同一のメッセージに署名を行い，その署名を集約し 1 つの電子署名で表す方式，アグリゲート署名方式とは複数の署名者がそれぞれ異なるメッセージに署名を行い，その署名を集約し 1 つの電子署名で表す方式である．しかし，各署名者間の関係，あるいは数人の署名者がある共通項（例：引用順，職位など）によって 1 つのグループとしてまとめられ，そのようなグループがいくつか存在したときのグループ間の関係の識別について，一般的な多重署名・アグリゲート署名方式では表現が難しいという課題がある．CGM コンテンツ作成の場合で検討すると，マッシュアップのコンテンツ引用順，あるいは素材提供者や編集者といったグループごとの区別が必要となることが想定されるが，上記の理由からこれらの署名方式をそのまま適用することは難しい．また，署名者の署名順序を規定できる順序付き多重署名方式 [8] も存在するが，この方式は全署名者の署名順が一列であるときにその順序を保証するものである．したがって，全署名者がいくつかのグループに分かれているとき（たとえば，映像編集，音響編集などのように提供する素材の種類によって編集者（映像監督や音響監督など）が複数いるようなとき）に，そのグループ間の関係を表現することは不可能である．上記で説明した既存の多重署名・アグリゲート署名方式での課題を**図 1** に示す．

上記の課題に対し，これまで GDH（Gap-Diffie-Hellman）グループに基づく BLS 署名方式 [9]，およびこの方式に基づく多重署名方式 [10], [11] を拡張し，署名者の立場を木構造に配置できるときに，その関係も検証可能な木構造表記型多重署名方式が提案されている [12]．また，計算コストを削減する目的でグループ間の区別のみに特化した階層表記型多重署名方式も提案されている [13]．これらの提案方式は，署名対象となるメッセージがすべての署名者で同一である多重署名方式を拡張し，図 1 で示したうち，枝分

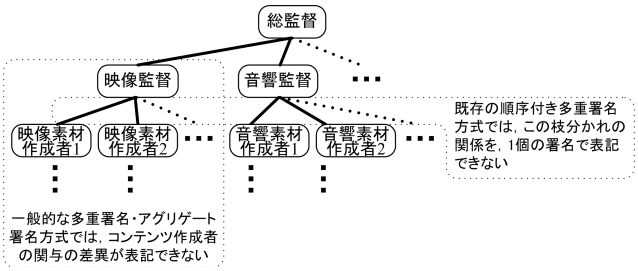


**図 1** 一般的な多重署名・アグリゲート署名の課題
**Fig. 1** Problems with ordinary multi-signature/aggregate signature methods.

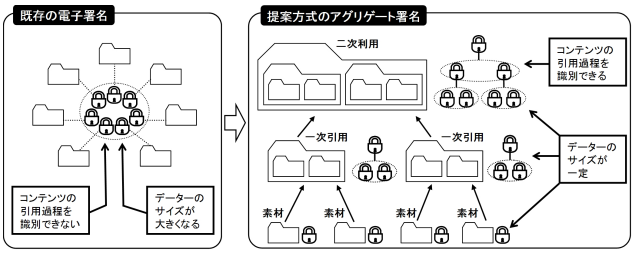


**図 2** 従来方式と提案方式による著作権表記能力の相違
**Fig. 2** A comparison of the proposed method with former methods about copyright expression.

かれの関係を表現できるようにしたものである．したがって，作成が完了したコンテンツに全著作権者が署名を行うシステムに有効である．一方，すでに流通しているコンテンツを引用・編集して新たなコンテンツを作成し，その引用者が追加で署名を行う，あるいは各制作者が内容を追加・変更していきながら共同でコンテンツ制作を行っていくような場面では，各制作者が署名対象とするデータの内容が異なるため，従来の研究では図 1 で示したうち，コンテンツ作成者の関与の差異については完全に表現できない．したがって，署名対象のメッセージが署名者によって異なることを想定したアグリゲート署名を拡張した方式が必要となる．

そこで，まず GDH グループに基づいたアグリゲート署名方式 [7] を拡張し，署名の順番が前後になる 2 人の署名者について，後者が自分とその前者それぞれが署名対象にしているメッセージに署名し，この署名を順次合成していくことで署名順序をも検証可能とするアグリゲート署名方式を考案する．さらにこの署名者間の手順を 1 対多に拡張し，多段的に繰り返すことで木構造表記型を実現したアグリゲート署名方式を考案する．これにより，**図 2** で示したように，コンテンツの引用過程も含めた著作権表記の保証にも適応が可能となる．

本稿では，2 章で GDH グループの定義と BLS 署名，および階層を考慮したセキュリティプロトコルに関する関連研究を示す．3 章で準備として提案する新しい順序付きアグリゲート署名方式について説明と考察を行い，4 章でこの順序付きアグリゲート署名方式を拡張した新しい木構造

© 2012 Information Processing Society of Japan

表記型アグリゲート署名方式について説明を行う．5 章で提案した署名方式の考察を行い，6 章でまとめとする．

## 2. 関連研究

### 2.1 GDH グループ

Okamoto らにより定義された問題 [14] を基に，GDH グループが定義された [9], [10], [15]．最初に CDH（Computational Diffie-Hellman）問題，および DDH（Decisional Diffie-Hellman）問題の 2 種類の Diffie-Hellman 問題について整理され，$\mathbb{G}'$ を位数 $p$ の乗法巡回群としたとき，これらの問題は以下のとおりとなる．

**CDH 問題：**$a, b \in \mathbb{Z}_p^*$，および $g \in \mathbb{G}'$ があり，$(g, g^a, g^b)$ の組だけが既知のとき，$g^{ab}$ を求める問題．

**DDH 問題：**$a, b, c \in \mathbb{Z}_p^*$，および $g \in \mathbb{G}'$ があり，$(g, g^a, g^b, g^c)$ の組だけが既知のとき，$c = ab$ であるかを判定する問題．

ここで，CDH 問題は難しい問題である一方，DDH 問題は簡単な問題であるという条件が満たされる場合，この $\mathbb{G}'$ を GDH グループと定義する．

### 2.2 ペアリングによる BLS 署名

2.1 節で定義された GDH グループに基づく様々な研究が行われ [16], [17]，ペアリング [19] と呼ばれる関数を利用することで，楕円曲線上における GDH グループが構成可能であることが示された．

最初に楕円曲線上における co-CDH（Computational co-Diffie-Hellman）問題，および co-DDH（Decisional co-Diffie-Hellman）問題を定義する．$\mathbb{G}'_1$，$\mathbb{G}'_2$ を位数 $p$ の異なる加法巡回群としたとき，これらの問題は以下のとおりとなる．

**co-CDH 問題：**$a \in \mathbb{Z}_p^*$，および $g_1 \in \mathbb{G}'_1$，$g_2 \in \mathbb{G}'_2$ があり，$(g_1, g_2, ag_1)$ の組だけが既知のとき，$ag_2$ を求める問題．

**co-DDH 問題：**$a, b \in \mathbb{Z}_p^*$，および $g_1 \in \mathbb{G}'_1$，$g_2 \in \mathbb{G}'_2$ があり，$(g_1, g_2, ag_1, bg_2)$ の組だけが既知のとき，$a = b$ であるかを判定する問題．

さらにペアリングの特徴を説明する．$\mathbb{G}_1$，$\mathbb{G}_2$ をペアリングの演算が可能な位数 $p$ の異なる加法巡回群とし，$e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_\tau$ を $\mathbb{G}_1$，$\mathbb{G}_2$ の入力から $\mathbb{G}_\tau$ への写像を行うペアリング関数とする．このペアリング関数は双線形性の特徴により，以下の式が成立する．

- $P_1, P_2 \in \mathbb{G}_1$，$Q \in \mathbb{G}_2$ に対し，
$$e(P_1 + P_2, Q) = e(P_1, Q)e(P_2, Q)$$
- $P \in \mathbb{G}_1$，$Q_1, Q_2 \in \mathbb{G}_2$ に対し，
$$e(P, Q_1 + Q_2) = e(P, Q_1)e(P, Q_2)$$
- $a, b \in \mathbb{Z}_p^*$，および $P \in \mathbb{G}_1$，$Q \in \mathbb{G}_2$ に対し，
$$e(aP, bQ) = e(bP, aQ) = e(abP, Q) = e(P, abQ) = e(P, Q)^{ab}$$

$\mathbb{G}_1$，$\mathbb{G}_2$ として楕円曲線上の加法巡回群を用いた場合，一般的に co-CDH 問題は難しい問題と考えられている．一方，このペアリングの特徴により co-DDH 問題は容易に解くことが可能な問題となり，楕円曲線上で GDH グループが構成可能となる．

Boneh らにより，このペアリングの特徴を用いて構成される署名方式が実現できることが示され，Short Signature の提案が行われた [9]．本稿では，この署名方式を BLS 署名と呼ぶことにする．

BLS 署名は以下のとおりに構成される．

**鍵生成：**$g \in \mathbb{G}_1$ を生成元とする．$x \in \mathbb{Z}_p^*$ を選び，$v = xg$ を計算する．$x$ を署名鍵，$v$ を検証鍵とする．

**署名作成：**一方向性ハッシュ関数 $H : \{0,1\}^* \to \mathbb{G}_2$ を定義する．$m$ を署名対象となるメッセージとしたとき，署名者は $h = H(m)$ を計算し，$\sigma = xh$ を生成する．$\sigma$ を $m$ に対する電子署名とする．

**署名検証：**検証者に $g$，$v$，$m$，$\sigma$ が与えられたとき，その検証者は $h = H(m)$ を計算し，$e(g, \sigma) = e(v, h)$ であるかを判定する．

上記で定義されている一方向性ハッシュ関数としては *MapToGroup* がある [9], [15]．このとき，電子署名が正しく作成されているのであれば，署名検証における判定式の左辺は $e(g, \sigma) = e(g, xh) = e(g, h)^x$，右辺は $e(v, h) = e(xg, h) = e(g, h)^x$ となり，同じ値になる．さらにこの方式について存在的偽造不能性が証明されている [9]．

この BLS 署名を基に，署名対象となるメッセージが各署名者ですべて異なっている電子署名を集約し，署名サイズが署名者数に依存せずに一定値以下とすることを可能としたアグリゲート署名方式が提案されている [7]．

新たに $\mathbb{U} = \{u_1, \cdots, u_n\}$ を署名作成が可能な署名者のグループとして定義し，さらに $\mathbb{L} = \{u_{i_1}, \cdots, u_{i_l}\} \subseteq \mathbb{U}$ を実際にアグリゲート署名作成に参加した署名者のグループと定義する．さらに $\mathbb{J} = \{i_1, \cdots, i_l\}$ を，このアグリゲート署名へ参加した署名者を示す符号（$u_k$ の $k$ に相当）の全員分の集合とする．このとき，アグリゲート署名は以下のとおりに構成される．

**鍵生成：**$g \in \mathbb{G}_1$ を生成元とする．署名者 $u_i \in \mathbb{U}$ について $x_i \in \mathbb{Z}_p^*$ を選び（すべての署名者の署名鍵は各々異なるものとする），$v_i = x_i g$ を計算する．$x_i$ を署名者 $u_i$ の署名鍵，$v_i$ を署名者 $u_i$ の検証鍵とする．

**署名作成：**一方向性ハッシュ関数 $H : \{0,1\}^* \to \mathbb{G}_2$ を定義する．$m_{i_\alpha}$ をアグリゲート署名作成に参加する署名者 $u_{i_\alpha} \in \mathbb{L}$ の署名対象となるメッセージとしたとき，署名者 $u_{i_\alpha} \in \mathbb{L}$ は $h_{i_\alpha} = H(m_{i_\alpha})$ を計算し，$\sigma_{i_\alpha} = x_{i_\alpha} h_{i_\alpha}$ を生成する．$\sigma_{i_\alpha}$ を $m_{i_\alpha}$ に対する署名者 $u_{i_\alpha}$ の電子署

© 2012 Information Processing Society of Japan

名とする．

**アグリゲート：**アグリゲート署名作成に参加するすべての署名者の電子署名 $\sigma_{i_\alpha}$ を集め，$\sigma = \sum_{j\in\mathbb{J}} \sigma_j$ を計算する．$(m_{i_1}, \ldots, m_{i_l}, \mathbb{L}, \sigma)$ をメッセージとアグリゲート署名の組とする．

**署名検証：**検証者に $(m_{i_1}, \ldots, m_{i_l}, \mathbb{L}, \sigma)$, および $g$ が与えられ，さらに $\mathbb{L}$ から検証に必要となるすべての検証鍵 $v_j$（$j \in \mathbb{J}$）が得られたとき，検証者はすべての $m_{i_\alpha}$ から $h_{i_\alpha} = H(m_{i_\alpha})$ を計算し，ペアリングを用いて $e(g,\sigma) = \prod_{j\in\mathbb{J}} e(v_j, h_j)$ であるかを判定する．

このとき，電子署名が正しく作成されているのであれば，署名検証における判定式の左辺は

$$e(g,\sigma) = e\left(g, \sum_{j\in\mathbb{J}} x_j h_j\right) = \prod_{j\in\mathbb{J}} e(g, x_j h_j) = \prod_{j\in\mathbb{J}} e(g,h_j)^{x_j}$$

右辺は

$$\prod_{j\in\mathbb{J}} e(v_j, h_j) = \prod_{j\in\mathbb{J}} e(x_j g, h_j) = \prod_{j\in\mathbb{J}} e(g,h_j)^{x_j}$$

となり，同じ値になる．さらにこの方式について存在的偽造不能性が証明されている [7]．

### 2.3 階層を考慮したセキュリティプロトコル

鍵生成の効率化を目的として階層を考慮したセキュリティプロトコルとしては，階層的 ID ベース暗号と呼ばれる方式がある [20]．ID ベース暗号とは，ユーザの ID 情報から公開鍵が生成可能となることで，証明書の発行を必要としない公開鍵暗号方式である [21]．この方式では，PKG（Private Key Generator：秘密鍵生成センタ）と呼ばれる 1 つの信頼できる機関を設定する．PKG は master secret と呼ばれる秘密鍵を生成し，ユーザの秘密鍵を生成するのに使用される．すべてのユーザの秘密鍵を，同じ master secret を利用して 1 つの PKG が生成するため，ユーザ数の増加とともに PKG の負担が増大するという課題がある．そこで，複数の PKG を階層的に配置し，ルートに対応する PKG は自らの秘密鍵（master secret に対応）を用いて，子供となる PKG の秘密鍵を生成することで，下位の PKG への秘密鍵生成の委任が可能となる．この方式は秘密鍵生成の効率化のために PKG を階層化しており，ユーザが階層化されているものではない．

復号処理の効率化を目的としたものとしては，階層的暗号化と呼ばれる方式がある．これは主に JPEG2000 など，データが階層的に記録され，1 つのファイルで様々な品質での利用が可能となっているコンテンツデータなどに適応されるものである．復号鍵は階層的に導出可能な構造になっており，許可されている品質以上での利用を防止すると同時に，後から高品質での利用を求めるユーザに対し，追加利用する階層までの復号鍵の差分を発行することで利用を許可する方式となっている [22], [23]．

電子署名方式においては，多重署名における階層化表現のアイデアが示されている [11]．しかし，一部に具体的な演算手法が定義されていない数式が示されており，実現性に課題がある．

実現可能なアグリゲート署名における階層表現については，Yamamoto らによって提案されている方式 [24] がある．これは，自身以前の接続関係の情報に自身の識別情報とメッセージを 1 つにしたデータを作成し，これを署名対象とすることでそれまでの署名者との関係を表している．したがって，署名のプロトコルとして順序性を保証しているものではなく，メッセージの一部に階層情報を記載しておくという手法となっている．一方，この提案方式は署名作成時には過去の署名参加者が作成したアグリゲート署名をすべて集める必要がある．このため，本稿の目的である「コンテンツを引用して新たなコンテンツを作成していく過程を保証すること」を考えた場合，過去にコンテンツ作成にかかわった人の情報をすべて集めてくる必要があり，編集者の負荷が徐々に増えるといった課題がある．上記とは別に，白川らによる提案方式 [25] もある．この方式では，ある階層以下でメッセージの改変が必要になった際に，その階層以下の署名だけを入れ替えることで新たなアグリゲート署名が作成可能となるといった利点がある．この利点により，たとえば編集コンテンツの著作権保護システムに適応させることで，作成コンテンツの一部改修などの編集作業を可能にしている．一方，階層を識別するための中間鍵が必要となり，階層の深さに依存して中間鍵の個数が増加することにともなう署名サイズの増加という課題が残されている．

## 3. 順序付きアグリゲート署名（提案 1）

1 章で説明したとおり，引用後に作成されるマッシュアップコンテンツは引用前のコンテンツとは異なるデータとなる．したがって，マッシュアップコンテンツにおける著作権の継承を表現するために，署名対象が各署名者で異なるアグリゲート署名を基に，木構造表記型の仕組みを導入する必要がある．また 2.3 節で説明したとおり，これまでの階層表記型アグリゲート署名ではマッシュアップコンテンツの引用過程を保証するために用いるには課題がある．

本稿の目的は上記を考慮した木構造表記型署名方式を提案することであるが，本章ではまずはその準備として BLS 署名方式に基づいた新しい順序付きアグリゲート署名（以後，“提案 1” とする）に関する議論を行い，4 章で提案 1 を拡張した木構造表記型アグリゲート署名に関する議論を行う．以下，この提案 1 について説明する．

### 3.1 記号，および前提条件

提案 1 において使用される記号，および前提条件につい

© 2012 Information Processing Society of Japan

て以下に示す．

記号：

$\mathbb{G}_1$，$\mathbb{G}_2$：ペアリングの演算が可能な楕円曲線上の点の集合（$\mathbb{G}_1$ の要素をペアリング関数の第 1 引数，$\mathbb{G}_2$ の要素をペアリング関数の第 2 引数とする）．

$g$：$\mathbb{G}_1$ の要素である生成元．

$e$：ペアリング関数．

$u_o$：第 $o$ 署名者（ただし $u_o \neq \text{null}$）．

$x_o$，$v_o$：$u_o$ の署名鍵，および検証鍵．

$\mathbb{L}_o$：$u_1$ から $u_o$ までの署名順序情報

$m_o$：$u_o$ が署名対象とするメッセージ．

$H$：$\{0,1\}^* \to \mathbb{G}_2$ となる一方向性ハッシュ関数．

$\sigma_o$：$u_1$ から $u_o$ までの順序付きアグリゲート署名．

前提条件：

- 公開鍵基盤は整備されており，認証局によりすべての署名者の署名鍵，および検証鍵のペアが正当に発行されている．
- 署名者に発行されている鍵ペア以外に，新たな鍵ペアの発行は行わない．
- 署名者は署名作成時において，定められた手順により正当に署名作成を行う．
- アグリゲート署名作成中において，署名者間の通信は安全に行われ，作成中の中間情報を第三者が入手することは不可能である．

### 3.2 プロトコル

#### 3.2.1 鍵生成

$g \in \mathbb{G}_1$ を生成元とする．署名者 $u_i$ について $x_i \in \mathbb{Z}_p^*$ を選び（すべての署名者の署名鍵は各々異なるものとする），$v_i = x_i g$ を計算する．$x_i$ を署名者 $u_i$ の署名鍵，$v_i$ を署名者 $u_i$ の検証鍵とする．

#### 3.2.2 署名作成

以下の手順で，アグリゲート署名が作成される．

(1) 第 1 署名者 $u_1$ は，メッセージ $m_1$ から $h_1 = H(m_1)$ を求め，2.2 節で説明した BLS 署名と同様な署名作成処理を行うことで $\sigma_1 = x_1 h_1$ を計算する．また，$\mathbb{L}_1 = \{(\text{null}, u_1)\}$ を作成する．この $\sigma_1$，$\mathbb{L}_1$，および $m_1$（または $h_1$）を第 2 署名者 $u_2$ に送信する．

(2) 第 $i$ 署名者 $u_i$ は，第 $i-1$ 署名者 $u_{i-1}$ から受信した $m_{i-1}$ を用いて $h_{i-1} = H(m_{i-1})$ を求める（または $h_{i-1}$ を受信したら，それを利用する）．さらに署名者 $u_i$ は，自分が本来署名したいメッセージ $m_i$ から $h_i = H(m_i)$ を求める．これと署名者 $u_{i-1}$ から受信した $\sigma_{i-1}$ を用いて

$$\begin{aligned}\sigma_i &= \sigma_{i-1} + x_i h_{i-1} + x_i h_i \\ &= x_1 h_1 + \sum_{j=2}^{i} (x_j h_{j-1} + x_j h_j)\end{aligned}$$

を計算する．また，受信した $\mathbb{L}_{i-1}$ を用いて

$$\begin{aligned}\mathbb{L}_i &= \mathbb{L}_{i-1} + \{(u_{i-1}, u_i)\} \\ &= \{(\text{null}, u_1), (u_1, u_2), \ldots, (u_{i-1}, u_i)\}\end{aligned}$$

を作成する．この $\sigma_i$，$\mathbb{L}_i$，および $m_i$（または $h_i$）を第 $i+1$ 署名者 $u_{i+1}$ に送信する．この手順を最後から 1 人手前の署名者まで再帰的に行う．

(3) 最後の署名者 $u_n$ は，直前の署名者 $u_{n-1}$ から受信した $m_{n-1}$ を用いて $h_{n-1} = H(m_{n-1})$ を求める（または $h_{n-1}$ を受信したら，それを利用する）．さらに署名者 $u_n$ は，自分が本来署名したいメッセージ $m_n$ から $h_n = H(m_n)$ を求める．これと署名者 $u_{n-1}$ から受信した $\sigma_{n-1}$ を用いて

$$\begin{aligned}\sigma_n &= \sigma_{n-1} + x_n h_{n-1} + x_n h_n \\ &= x_1 h_1 + \sum_{j=2}^{n} (x_j h_{j-1} + x_j h_j)\end{aligned}$$

を計算する．また，受信した $\mathbb{L}_{n-1}$ を用いて

$$\begin{aligned}\mathbb{L}_n &= \mathbb{L}_{n-1} + \{(u_{n-1}, u_n)\} \\ &= \{(\text{null}, u_1), (u_1, u_2), \ldots, (u_{n-1}, u_n)\}\end{aligned}$$

を作成する．この $\sigma_n$，および $\mathbb{L}_n$ を，署名者 $u_1, \ldots, u_n$ の，署名対象 $m_1, \ldots, m_n$ に対するアグリゲート署名として公開する．

#### 3.2.3 署名検証

以下の手順で，アグリゲート署名の検証を行う．

(1) 検証者は，$\mathbb{L}_n$ に示されているすべての署名者の検証鍵 $v_1, \ldots, v_n$，および署名対象となるすべてのメッセージ $m_1, \ldots, m_n$ を集める．署名の順序は，$\mathbb{L}_n$ の要素のうち第 1 項が null の要素の第 2 項に示されている署名者が最初の署名者，その署名者が第 1 項に示されている要素の第 2 項に示されている署名者が第 2 署名者というように決定する．

(2) 検証者は，集めたメッセージから $h_i = H(m_i)$ を求める．

(3) 検証者は，

$$\begin{aligned}&e(v_1, h_1) \left( \prod_{j=2}^{n} e(v_j, h_{j-1} + h_j) \right) \\ &\quad = e(v_1, h_1) e(v_2, h_1 + h_2) \ldots e(v_n, h_{n-1} + h_n)\end{aligned}$$

を計算し，この値と $e(g, \sigma_n)$ の値が一致することを確認する．

もし正しく署名が作成されていれば，(3) の第 2 の式は

$$\begin{aligned}e(g, \sigma_n) &= e\left(g, x_1 h_1 + \sum_{j=2}^{n} (x_j h_{j-1} + x_j h_j)\right) \\ &= e(g, x_1 h_1) e(g, x_2 (h_1 + h_2)) \ldots e(g, x_n (h_{n-1} + h_n))\end{aligned}$$

© 2012 Information Processing Society of Japan

$$=e(x_1g,h_1)e(x_2g,h_1+h_2)\dots e(x_ng,h_{n-1}+h_n)$$

となり，(3) の第 1 の式と一致する．

#### 3.2.4 アグリゲート署名への新たな署名者の追加

最後の署名者 $u_n$ は $\sigma_n$，および $\mathbb{L}_n$ を署名対象 $m_1,\dots,m_n$ に対するアグリゲート署名として公開している．

ここで，新たな署名者 $u_{n+1}$ が自分を追加して新たなアグリゲート署名を作成する場合を考える．このとき，署名者 $u_n$ は作成したアグリゲート署名（$\sigma_n,\mathbb{L}_n$），および $m_n$（または $h_n$）を新たな署名者 $u_{n+1}$ に送信し，署名者 $u_{n+1}$ が 3.2.2 項 (3) の手順を行う．すなわち署名者 $u_{n+1}$ は

$$\begin{aligned}\sigma_{n+1}&=\sigma_n+x_{n+1}h_n+x_{n+1}h_{n+1}\\&=x_1h_1+\sum_{j=2}^{n+1}(x_jh_{j-1}+x_jh_j)\end{aligned}$$

を計算し，

$$\begin{aligned}\mathbb{L}_{n+1}&=\mathbb{L}_n+\{(u_n,u_{n+1})\}\\&=\{(\mathrm{null},u_1),(u_1,u_2),\dots,(u_n,u_{n+1})\}\end{aligned}$$

を作成する．この $\sigma_{n+1}$，および $\mathbb{L}_{n+1}$ を新たなアグリゲート署名として公開する．

この手順で示したように，新たな署名者が公開されているアグリゲート署名に自分を追加した新たなアグリゲート署名を作成する場合，それまでの最終署名者は新たな処理を必要とせず，公開されている情報を新たな署名者に送信するだけでよい．

## 4. コンテンツ引用過程表記手法（提案 2）

3 章で示した提案 1 は，署名者が 1 列方向で順番に署名を作成する際に，前後の署名者間の関係を示す演算を導入し，その手順を繰り返すことで，署名者の順序を規定している．この手順を隣接する署名者に対応して 1 対多に拡張し，多段的に繰り返すことで，木構造表記型アグリゲート署名を構成することが可能となる．

本章では，この木構造表記型アグリゲート署名方式を用いたコンテンツ引用過程表記手法（以後，“提案 2” とする）について説明する．**図 3** に提案 2 が想定する署名者の関係を例示する．

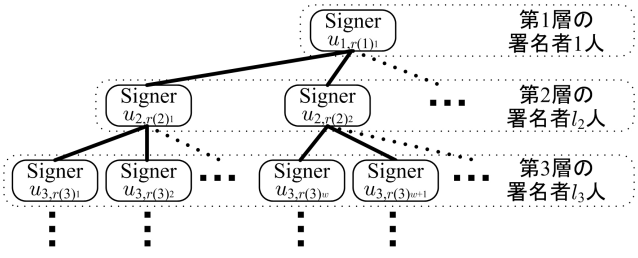


**図 3** 提案 2 における署名者の関係
**Fig. 3** Relations among signers under the proposal 2.

### 4.1 記号，および前提条件

提案 2 において使用される記号について，3.1 節で定義したものに加えて，下記を新たに定義する．

$u_{q,r(q)_w}$：コンテンツの作成者，あるいは編集者であり，アグリゲート署名作成に参加した署名者（各署名者は木構造におけるノードに配置され，$q$ は，ルートノードを階層 1 としたときの，その作成者・編集者が位置するノードの階層を示し（$1\le q\le n$），$r(q)_w$ は階層 $q$ に位置する署名者の識別符号（$l_q$ を $q$ 階層にいる全署名者数としたとき，$1\le w\le l_q$）とする．ただし $u_{q,r(q)_w}\neq\mathrm{null}$）．

$x_{q,r(q)_w},v_{q,r(q)_w}$：$u_{q,r(q)_w}$ の署名鍵，および検証鍵．

$\sigma_{q,r(q)_w}$：$u_{q,r(q)_w}$ までの木構造表記型アグリゲート署名．

$\mathbb{L}_{q,r(q)_w}$：部分的な木構造で表現される $u_{q,r(q)_w}$ 以下の署名者の関係を示した付随情報（たとえば**図 4** のような署名者の関係の場合は

$$\begin{aligned}\mathbb{L}_{q,r(q)_w}=\{&(u_{q,r(q)_w},\{u_{q+1,r(q+1)_\alpha},\dots\}),\\&(u_{q+1,r(q+1)_\alpha},\{u_{q+2,r(q+2)_\beta},\dots\}),\dots,\\&(u_{i-1,r(i-1)_\gamma},\{u_{i,r(i)_\delta},\dots\}),\dots\\&(\mathrm{null},\{u_{i,r(i)_\epsilon}\}),\dots\}\end{aligned}$$

ただし，他の部分木における添字の符号との関係で，$\alpha,\dots,\epsilon$ は途中の数字からの数字となる場合がある）．

$m_{q,r(q)_w}$：$u_{q,r(q)_w}$ が署名対象とするメッセージ（コンテンツのハッシュ値やメタ情報など）．

また，前提条件については 3.1 節で示したものと同じである．

### 4.2 プロトコル

本節では，木構造におけるリーフノードをオリジナルコンテンツ作成者，中間ノードは下位のノードのコンテンツを編集していく編集者，ルートを最終編集者と規定し，提案 2 に用いるために考案した新しい木構造表記型多重署名について説明する．

#### 4.2.1 鍵生成

$g\in\mathbb{G}_1$ を生成元とする．署名者 $u_{q,r(q)_w}$ について，$x_{q,r(q)_w}\in\mathbb{Z}_p^*$ を選び（すべての署名者の署名鍵は各々異なるものとする），$v_{q,r(q)_w}=x_{q,r(q)_w}g$ を計算する．$x_{q,r(q)_w}$ を署名

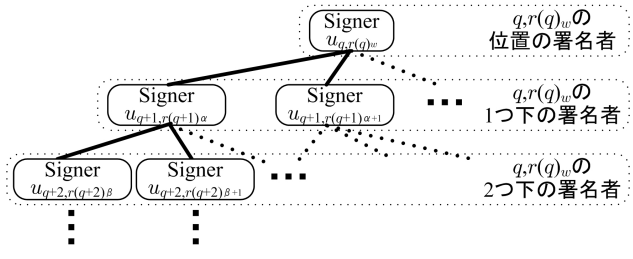


**図 4** $\mathbb{L}_{q,r(q)_w}$ が示す署名者の関係
**Fig. 4** Relations among signers showed by $\mathbb{L}_{q,r(q)_w}$.

© 2012 Information Processing Society of Japan

者 $u_{q,r(q)_w}$ の署名鍵，$v_{q,r(q)_w}$ を署名者 $u_{q,r(q)_w}$ の検証鍵とする．

#### 4.2.2 署名作成

以下の手順で，アグリゲート署名が作成される．

(1) リーフノードに位置する署名者 $u_{l,r(l)_{\alpha 0}}$ は，メッセージ $m_{l,r(l)_{\alpha 0}}$ から $h_{l,r(l)_{\alpha 0}} = H(m_{l,r(l)_{\alpha 0}})$ を求め，2.2 節で説明した BLS 署名と同様な署名作成処理を行うことで

$$\sigma_{l,r(l)_{\alpha 0}} = x_{l,r(l)_{\alpha 0}} h_{l,r(l)_{\alpha 0}}$$

を計算する．また，

$$\mathbb{L}_{l,r(l)_{\alpha 0}} = (\mathrm{null}, \{u_{l,r(l)_{\alpha 0}}\})$$

を作成する．この $\sigma_{l,r(l)_{\alpha 0}}$，$\mathbb{L}_{l,r(l)_{\alpha 0}}$，および $m_{l,r(l)_{\alpha 0}}$（または $h_{l,r(l)_{\alpha 0}}$）を自分の親ノードに位置する署名者 $u_{l-1,r(l-1)_{\beta 0}}$ に送信する．

(2) 中間ノードに位置する（リーフノードとルート以外の）署名者 $u_{s,r(s)_\beta}$ は，自分の子ノードに位置する署名者 $u_{s+1,r(s+1)_{\gamma_1}}, \ldots, u_{s+1,r(s+1)_{\gamma_{k'}}}$ から受信したすべての $m_{s+1,r(s+1)_i}$（ただし $\gamma_1 \le i \le \gamma_{k'}$）を用いて $h_{s+1,r(s+1)_i} = H(m_{s+1,r(s+1)_i})$ を求める（または $h_{s+1,r(s+1)_i}$ を受信したら，それを利用する）．さらに署名者 $u_{s,r(s)_\beta}$ は，自分が本来署名したいメッセージ $m_{s,r(s)_\beta}$ から $h_{s,r(s)_\beta} = H(m_{s,r(s)_\beta})$ を求める．これとすべての子ノードの署名者 $u_{s+1,r(s+1)_i}$ から受信した $\sigma_{s+1,r(s+1)_i}$ を用いて

$$\begin{aligned}&\sigma_{s,r(s)_\beta}\\ &= \sum_{i=\gamma_1}^{\gamma_{k'}} (\sigma_{s+1,r(s+1)_i} + x_{s,r(s)_\beta} h_{s+1,r(s+1)_i})\\ &\quad + x_{s,r(s)_\beta} h_{s,r(s)_\beta}\end{aligned}$$

を計算する．また，

$$\begin{aligned}&\mathbb{L}_{s,r(s)_\beta}\\ &= \sum_{i=\gamma_1}^{\gamma_{k'}} \mathbb{L}_{s+1,r(s+1)_i}\\ &\quad + \{(u_{s,r(s)_\beta}, \{u_{s+1,r(s+1)_1}, \ldots, u_{s+1,r(s+1)_{k'}}\})\}\end{aligned}$$

を作成する．この $\sigma_{s,r(s)_\beta}$，$\mathbb{L}_{s,r(s)_\beta}$，および $m_{s,r(s)_\beta}$（または $h_{s,r(s)_\beta}$）を自分の親ノードに位置する署名者 $u_{s-1,r(s-1)_{\beta'}}$ に送信する．この手順をルートの子ノードに位置する署名者まで再帰的に行う．

(3) ルートに位置する署名者 $u_{1,r(1)_1}$ は，自分の子ノードに位置する署名者 $u_{2,r(2)_1}, \ldots, u_{2,r(2)_\delta}$ から受信したすべての $m_{2,r(2)_i}$（ただし $1 \le i \le \delta$）を用いて $h_{2,r(2)_i} = H(m_{2,r(2)_i})$ を求める（または $h_{2,r(2)_i}$ を受信したら，それを利用する）．さらに署名者 $u_{1,r(1)_1}$ は，自分が本来署名したいメッセージ $m_{1,r(1)_1}$ から $h_{1,r(1)_1} = H(m_{1,r(1)_1})$ を求める．これとすべての子ノードの署名者 $u_{2,r(2)_i}$ から受信した $\sigma_{2,r(2)_i}$ を用いて

$$\begin{aligned}\sigma_{1,r(1)_1} &= \sum_{i=1}^{\delta} (\sigma_{2,r(2)_i} + x_{1,r(1)_1} h_{2,r(2)_i})\\ &\quad + x_{1,r(1)_1} h_{1,r(1)_1}\end{aligned}$$

を計算する．また，

$$\begin{aligned}\mathbb{L}_{1,r(1)_1} &= \sum_{i=1}^{\delta} \mathbb{L}_{2,r(2)_i}\\ &\quad + \{(u_{1,r(1)_1}, \{u_{2,r(2)_1}, \ldots, u_{2,r(2)_\delta}\})\}\end{aligned}$$

を作成する．この $\sigma_{1,r(1)_1}$，および $\mathbb{L}_{1,r(1)_1}$ を，全署名者 $u_{q,r(q)_w}$ の，全署名対象 $m_{q,r(q)_w}$ に対するアグリゲート署名として公開する．

#### 4.2.3 署名検証

以下の手順で，アグリゲート署名の検証を行う．

(1) 検証者は，$\mathbb{L}_{1,r(1)_1}$ に示されているすべての署名者の検証鍵 $v_{q,r(q)_w}$，および署名対象となるすべてのメッセージ $m_{q,r(q)_w}$ を集める．

(2) 検証者は，集めたメッセージから

$$h_{q,r(q)_w} = H(m_{q,r(q)_w})$$

を求める．

(3) 検証者は，

$$e(v_{1,r(1)_1}, h_{1,r(1)_1})$$

と

$$\prod_{q,i,j \in \mathbf{All}} e(v_{q,r(q)_i} + v_{q+1,r(q+1)_j}, h_{q+1,r(q+1)_j})$$

（ただし，**All** とはアグリゲート署名における木構造の直接の親子関係になっている箇所すべて（の署名者）を示す）の積を計算し，この値と $e(g, \sigma_{1,r(1)_1})$ の値が一致することを確認する．

もし正しく署名が作成されていれば，

$$\begin{aligned}&e(g, \sigma_{1,r(1)_1})\\ =&e\left(g, \sum_{i=1}^{\delta} (\sigma_{2,r(2)_i} + x_{1,r(1)_1} h_{2,r(2)_i}) + x_{1,r(1)_1} h_{1,r(1)_1}\right)\\ =&e(g, x_{1,r(1)_1} h_{1,r(1)_1})\\ &\cdot e\left(g, \sum_{i=1}^{\delta} (\sigma_{2,r(2)_i} + x_{1,r(1)_1} h_{2,r(2)_i})\right)\\ &\cdots\\ =&e(g, x_{1,r(1)_1} h_{1,r(1)_1})\\ &\cdot e(g, \sum_{q,i,j \in \mathbf{All}} ((x_{q,r(q)_i} + x_{q+1,r(q+1)_j} h_{q+1,r(q+1)_j}))\end{aligned}$$

© 2012 Information Processing Society of Japan

$$= e(x_{1,r(1)_1} g, h_{1,r(1)_1}) \cdot \prod_{q,i,j \in \mathbf{All}} e((x_{q,r(q)_i} + x_{q+1,r(q+1)_j}) g, h_{q+1,r(q+1)_j})$$

となり，(3) の第 1 の式と第 2 の式との積の値と一致する．

#### 4.2.4 アグリゲート署名への新たな署名者の追加

ルートの署名者 $u_{1,r(1)_1}$ は $\sigma_{1,r(1)_1}$，および $\mathbb{L}_{1,r(1)_1}$ を全署名対象 $m_{q,r(q)_w}$ に対するアグリゲート署名として公開している．

ここで，新たな署名者が自分を追加して新たなアグリゲート署名を作成する場合を考える．ここでは便宜上，新たな署名者を $u_{0,r(0)_1}$，この署名者 $u_{0,r(0)_1}$ が自分自身を追加させるアグリゲート署名の最終署名者を $u_{1,r(1)_1}, \ldots, u_{1,r(1)_\epsilon}$ とする．すなわち，署名者 $u_{0,r(0)_1}$ は $\sigma_{1,r(1)_1}, \ldots, \sigma_{1,r(1)_\epsilon}$，および $\mathbb{L}_{1,r(1)_1}, \ldots, \mathbb{L}_{1,r(1)_\epsilon}$ に対し，自分自身を追加した新たなアグリゲート署名を作成する．

このとき，署名者 $u_{1,r(1)_i}$（ただし $1 \leq i \leq \epsilon$）は作成したアグリゲート署名（$\sigma_{1,r(1)_i}, \mathbb{L}_{1,r(1)_i}$），および $m_{1,r(1)_i}$（または $h_{1,r(1)_i}$）を新たな署名者 $u_{0,r(0)_1}$ に送信し，署名者 $u_{0,r(0)_1}$ が 4.2.3 項 (3) の手順を行う．すなわち署名者 $u_{0,r(0)_1}$ は

$$\sigma_{0,r(0)_1} = \sum_{i=1}^{\epsilon} (\sigma_{1,r(1)_i} + x_{0,r(0)_1} h_{1,r(1)_i}) + x_{0,r(0)_1} h_{0,r(0)_1}$$

を計算し，

$$\mathbb{L}_{0,r(0)_1} = \sum_{i=1}^{\epsilon} \mathbb{L}_{1,r(1)_i} + \{(u_{0,r(0)_1}, \{u_{1,r(1)_1}, \ldots, u_{1,r(1)_\epsilon}\})\}$$

を作成する．この $\sigma_{0,r(0)_1}$，および $\mathbb{L}_{0,r(0)_1}$ を，新たなアグリゲート署名として公開する．

この手順で示したように，新たな署名者が公開されているアグリゲート署名に自分を追加した新たなアグリゲート署名を作成する場合，それまでのルートの署名者は新たな処理を必要とせず，公開されている情報を新たな署名者に送信するだけでよい．

### 4.3 コンテンツ引用過程表記

4.2 節で，任意の構造に対応した木構造表記型アグリゲート署名プロトコルについて説明した．

ここで，CGM サービスでマッシュアップによるコンテンツの引用を行った際に，XML を拡張したものや，クリエイティブ・コモンズで使用されている RDF（正確には RDF を拡張した梶らによる方式 [3], [4]）といったメタデータの記載手法を用いて，著作権情報（ライセンス情報）の 1 つとしてそのコンテンツの引用過程を記載し，引用・編集されたコンテンツとあわせて提供することになる．このメタデータに対し，そのメタデータに対応するコンテンツ（のハッシュ値）とあわせて 1 つのメッセージと見なし，それに署名を付けることでメタデータの内容を保証することができる．この保証について，マッシュアップされるごとにその引用過程を木構造表記型アグリゲート署名として表記し，その電子署名の検証によりコンテンツの引用過程を確認できるようにする目的で，提案方式を適用する．

電子署名のセキュリティ要求事項については，Komano らの多重署名方式におけるセキュリティモデルの構築手法 [26] などが提唱されている．今回は CGM コンテンツのマッシュアップという特性から「引用過程表記」に特化し，オリジナルコンテンツ作成者や編集者の著作権を正当に主張できること，またコンテンツ作成に関する責任を明確にすることに着目し，以下に示す要求事項を満たすことにする．

**アグリゲート署名正当性（存在的偽造不能性）：** 著作権者を偽造したり，引用順序を入れ替えたりすることができないこと．すなわちアグリゲート署名に参加しない第三者がすべての公開情報を入手しても，アグリゲート署名を偽造したり，すでに作成されたアグリゲート署名の署名順序を勝手に変更したりすることが困難であること．

**参加否認不能性：** 1 つのコンテンツに対して隣接する 2 人が署名作成を行っている．その 2 人が結託により作成されたコンテンツに対する責任を逃れることが行えないこと．すなわちアグリゲート署名に参加している署名者が，結託してアグリゲート署名の参加を否認することが困難であること．

**偽装参加不能性：** コンテンツ作成に関与していない第三者に対して，勝手にそのコンテンツに対する責任を負わせることができないこと．すなわちアグリゲート署名に参加している署名者が，アグリゲート署名に参加している署名者かどうかにかかわらず，第三者を勝手に署名者として追加することが困難であること．

本節では，このアグリゲート署名を用いたコンテンツ引用過程表記の手順について，**図 5** に示す 2 分木 3 階層の場

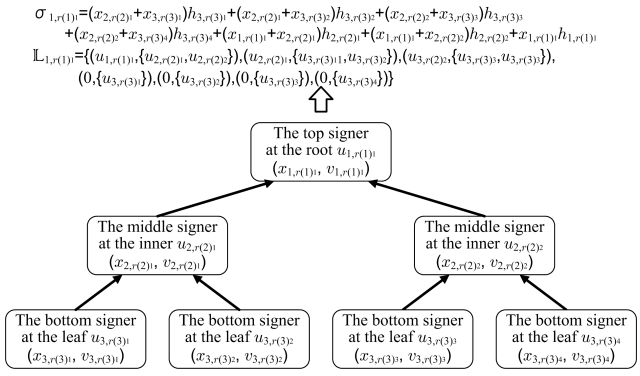


**図 5** 2 分木 3 階層におけるコンテンツ引用過程

**Fig. 5** A quotation process in contents on a binary tree structure with three layers.

© 2012 Information Processing Society of Japan

合を例にとり説明する．

#### 4.3.1 鍵生成

4.2.1 項の手順で $u_{1,r(1)_1}$，$u_{2,r(2)_1}$，$u_{2,r(2)_2}$，$u_{3,r(3)_1}$，$u_{3,r(3)_2}$，$u_{3,r(3)_3}$，$u_{3,r(3)_4}$ の署名鍵，および検証鍵を生成する．

#### 4.3.2 署名作成

以下の手順で，アグリゲート署名が作成される．

(1) オリジナルコンテンツ作成者（第3層の署名者）$u_{3,r(3)_1}$ は，メッセージ $m_{3,r(3)_1}$ から $h_{3,r(3)_1} = H(m_{3,r(3)_1})$ を求め，

$$\sigma_{3,r(3)_1} = x_{3.r(3)_1} h_{3.r(3)_1}$$

を計算する．また，

$$\mathbb{L}_{3,r(3)_1} = (\mathrm{null}, \{u_{3,r(3)_1}\})$$

を作成する．この $\sigma_{3,r(3)_1}$，$\mathbb{L}_{3,r(3)_1}$，および $m_{3,r(3)_1}$（または $h_{3,r(3)_1}$）を自分の親ノードに位置する中間編集者 $u_{2,r(2)_1}$ に送信する．第3層の他の作成者も同様の手順により署名を作成し，$u_{3,r(3)_2}$ は $u_{2,r(2)_1}$ へ，$u_{3,r(3)_3}$，および $u_{3,r(3)_4}$ は $u_{2,r(2)_2}$ へ送信する．

(2) 中間編集者（第2層の署名者）$u_{2,r(2)_1}$ は，自分の子ノードに位置する2人のオリジナルコンテンツ作成者 $u_{3,r(3)_1}, u_{3,r(3)_2}$ から受信した $m_{3,r(3)_1}, m_{3,r(3)_2}$ を用いて $h_{3,r(3)_1} = H(m_{3,r(3)_1}), h_{3,r(3)_2} = H(m_{3,r(3)_2})$ を求める（または $h_{3,r(3)_1}$，$h_{3,r(3)_2}$ を受信したら，それを利用する）．さらに中間編集者 $u_{2,r(2)_1}$ は，自分が本来署名したいメッセージ $m_{2,r(2)_1}$ から $h_{2,r(2)_1} = H(m_{2,r(2)_1})$ を求める．これと $u_{3,r(3)_1}$，および $u_{3,r(3)_2}$ から受信した $\sigma_{3,r(3)_1}$，および $\sigma_{3,r(3)_2}$ を用いて

$$\begin{aligned}\sigma_{2,r(2)_1} = {} & \sigma_{3,r(3)_1} + x_{2,r(2)_1} h_{3.r(3)_1} + \sigma_{3,r(3)_2} \\ & + x_{2,r(2)_1} h_{3,r(3)_2} + x_{2,r(2)_1} h_{2,r(2)_1}\end{aligned}$$

を計算する．また，

$$\begin{aligned}\mathbb{L}_{2,r(2)_1} = \{ & (u_{2,r(2)_1}, \{u_{3,r(3)_1}, u_{3,r(3)_2}\}), \\ & (\mathrm{null}, \{u_{3,r(3)_1}\}), (\mathrm{null}, \{u_{3,r(3)_2}\})\}\end{aligned}$$

を作成する．この $\sigma_{2,r(2)_1}, \mathbb{L}_{2,r(2)_1}$，および $m_{2,r(2)_1}$（または $h_{2,r(2)_1}$）をルートに位置する最終編集者 $u_{1,r(1)_1}$ に送信する．第2層のもう1人の中間編集者 $u_{2,r(2)_2}$ も同様の手順により署名と付随情報を作成し，$u_{1,r(2)_1}$ へ送信する．

(3) 最終編集者（ルートに位置する署名者）$u_{1,r(1)_1}$ は，自分の子ノードに位置する2人の中間編集者 $u_{2,r(2)_1}$，$u_{2,r(2)_2}$ から受信した $m_{2,r(2)_1}$，$m_{2,r(2)_2}$ を用いて $h_{2,r(2)_1} = H(m_{2,r(2)_1})$，$h_{2,r(2)_2} = H(m_{2,r(2)_2})$ を求める（または $h_{2,r(2)_1}$，$h_{2,r(2)_2}$ を受信したら，それを利用する）．さらに最終編集者 $u_{1,r(1)_1}$ は，自分が本来署名したいメッセージ $m_{1,r(1)_1}$ から $h_{1,r(1)_1} = H(m_{1,r(1)_1})$ を求める．これと中間編集者 $u_{2,r(2)_1}$，および $u_{2,r(2)_2}$ から受信した $\sigma_{2,r(2)_1}$，および $\sigma_{2,r(2)_2}$ を用いて

$$\begin{aligned}\sigma_{1,r(1)_1} = {} & \sigma_{2,r(2)_1} + x_{1,r(1)_1} h_{2,r(2)_1} + \sigma_{2,r(2)_2} \\ & + x_{1,r(1)_1} h_{2,r(2)_2} + x_{1,r(1)_1} h_{1,r(1)_1}\end{aligned}$$

を計算する．また，

$$\begin{aligned}\mathbb{L}_{1,r(1)_1} = \{ & (u_{1,r(1)_1}, \{u_{2,r(2)_1}, u_{2,r(2)_2}\}), \\ & (u_{2,r(2)_1}, \{u_{3,r(3)_1}, u_{3,r(3)_2}\}), \\ & (u_{2,r(2)_2}, \{u_{3,r(3)_3}, u_{3,r(3)_4}\}), \\ & (\mathrm{null}, \{u_{3,r(3)_1}\}), (\mathrm{null}, \{u_{3,r(3)_2}\}), \\ & (\mathrm{null}, \{u_{3,r(3)_3}\}), (\mathrm{null}, \{u_{3,r(3)_4}\})\}\end{aligned}$$

を作成する．この $\sigma_{1,r(1)_1}$，および $\mathbb{L}_{1,r(1)_1}$ を，コンテンツ作成に関与した全員のアグリゲート署名として公開する．

#### 4.3.3 署名検証

以下の手順で，アグリゲート署名の検証を行う．

(1) 検証者は，$\mathbb{L}_{1,r(1)_1}$ に示されているすべての署名者の検証鍵 $v_{1,r(1)_1}$，$v_{2,r(2)_1}$，$v_{2,r(2)_2}$，$v_{3,r(3)_1}$，$v_{3,r(3)_2}$，$v_{3,r(3)_3}$，$v_{3,r(3)_4}$，および署名対象となるすべてのメッセージ $m_{1,r(1)_1}$，$m_{2,r(2)_1}$，$m_{2,r(2)_2}$，$m_{3,r(3)_1}$，$m_{3,r(3)_2}$，$m_{3,r(3)_3}$，$m_{3,r(3)_4}$ を集める．

(2) 検証者は，集めたメッセージから $h_{1,r(1)_1}$，$h_{2,r(2)_1}$，$h_{2,r(2)_2}$，$h_{3,r(3)_1}$，$h_{3,r(3)_2}$，$h_{3,r(3)_3}$，$h_{3,r(3)_4}$ を求める．

(3) 検証者は，ペアリングの積の値である

$$\begin{aligned}& e(v_{1,r(1)_1}, h_{1,r(1)_1}) \cdot e(v_{1,r(1)_1} + v_{2,r(2)_1}, h_{2,r(2)_1}) \\ & \cdot e(v_{1,r(1)_1} + v_{2,r(2)_2}, h_{2,r(2)_2}) \\ & \cdot e(v_{2,r(2)_1} + v_{3,r(3)_1}, h_{3,r(3)_1}) \\ & \cdot e(v_{2,r(2)_1} + v_{3,r(3)_2}, h_{3,r(3)_2}) \\ & \cdot e(v_{2,r(2)_2} + v_{3,r(3)_3}, h_{3,r(3)_3}) \\ & \cdot e(v_{2,r(2)_2} + v_{3,r(3)_4}, h_{3,r(3)_4})\end{aligned}$$

を計算し，この値と $e(g, \sigma_{1,r(1)_1})$ の値が一致することを確認する．

## 5. 考察

### 5.1 安全性の考察

4.3 節で定義したセキュリティの要求事項について考察する．

**アグリゲート署名正当性（存在的偽造不能性）：** 提案方式では 4.2.2 項で示したように，アグリゲート署名のセット（$\sigma_{1,r(1)_1}$ と $\mathbb{L}_{1,r(1)_1}$ のセット）を出力する．このセットの中の $\mathbb{L}_{1,r(1)_1}$ によって隣接する署名者の関係が示され，それに従い検証鍵と対応するメッセージ（のハッシュ値）を

© 2012 Information Processing Society of Japan

ペアリング関数に順次入力していくことで，アグリゲート署名 $\sigma_{1,r(1)_1}$ を検証する手順となっている．このときのアグリゲート署名の安全性について考察する．

提案2において，攻撃者 $A$ は偽装を行うある1人の（リーフノードの）署名者の検証鍵 $v'_{l_\alpha,r(l_\alpha)_1}$ を入力値としたときに，想定している木構造に応じて残りのノードの署名鍵と検証鍵のペアの出力を行うものとする．$A$ の攻撃が成功するとは，上記の入力値，および出力した鍵ペアに対し，想定した木構造に対応するアグリゲート署名 $\sigma'_{1,r(1)_1}$ を出力できること，すなわち $\sigma'_{1,r(1)_1}$ の偽造に成功することを意味する．このとき，以下の定理が成立する．

**定理：**ランダムオラクルモデルにおいて，提案2における $\sigma'_{1,r(1)_1}$ の偽造可能性と BLS 署名の偽造可能性は等価である．

**証明：**ここでは，Boldyreba が行った手法 [10], [27] を用いて，安全性を証明する．$A$ を $\sigma'_{1,r(1)_1}$ を偽造しようとする攻撃者とする．$B$ を BLS 署名を偽造する攻撃者としたとき，$A$ の攻撃が成功するなら，$B$ の攻撃が成功することを示す（$B$ の攻撃が成功するなら，$A$ の攻撃が成功することは自明であるため省略する）．

$B$ は1つの検証鍵 $v'_{l_\alpha,r(l_\alpha)_1}$ を持っており，ランダムオラクル，および署名オラクルへの応答を行う．

$B$ は $A$ を Honest Player として実行する．まず $B$ は $A$ に $v'_{l_\alpha,r(l_\alpha)_1}$ を与え，$A$ はそれ以外のリーフノードの署名鍵と検証鍵のペア $(x'_{l_\alpha,r(l_\alpha)_\beta}, v'_{l_\alpha,r(l_\alpha)_\beta})$，およびリーフノード以外のノードの署名鍵と検証鍵のペア $(x'_{q,r(q)_w}, v'_{q,r(q)_w})$ を出力する．また，$A$ はランダムオラクルと署名オラクルへの応答により $\sigma'_{1,r(1)_1}$ とその署名の対象となるメッセージ $m'_{q,r(q)_w}$（偽造ノードを含む全ノード分）を求め，$B$ に返答する．$B$ は $\sigma'_{1,r(1)_1}$ を用いて，

$$\begin{aligned}&\sigma'_{1,r(1)_1}\\&\quad-\sum_{i,j,k\in\textbf{All'}}((x'_{q_i,r(q_i)_j}+x'_{q_i,r(q_i)_j})\cdot H(m'_{q_i+1,r(q_i+1)_k}))\\&\quad-x'_{1,r(1)_1}H(m'_{1,r(1)_1})-x'_{l_\alpha+1,r(l_\alpha+1)_\gamma}H(m'_{l_\alpha,r(l_\alpha)_1})\\&\quad=x'_{l_\alpha,r(l_\alpha)_1}H(m'_{l_\alpha,r(l_\alpha)_1})\end{aligned}$$

（ただし，**All'** とはアグリゲート署名における木構造の直接の親子関係になっている箇所すべて（の署名者）のうち，$x'_{l_\alpha,r(l_\alpha)_1}$ が含まれている箇所を除外したものを示す）の計算により，$v'_{l_\alpha,r(l_\alpha)_1}$ に対応する BLS 署名を作成することができ，$B$ の攻撃が成功する．（QED）

以上により，BLS 署名の偽造が困難であれば，提案方式におけるアグリゲート署名の偽造も困難である．

**参加否認不能性：**隣接する2人の署名者 $u_{q_i,r(q_i)_\alpha}$，$u_{q_i+1,r(q_i+1)_\beta}$ の関係を示すために，前者が署名対象にしているメッセージのハッシュ値 $H(m_{q_i+1,r(q_i+1)_\beta})$ に対し，この2人の署名者が自分の署名鍵 $x_{q_i,r(q_i)_\alpha}$，$x_{q_i+1,r(q_i+1)_\beta}$ を用いて

$$(x_{q_i,r(q_i)_\alpha}+x_{q_i+1,r(q_i+1)_\beta})H(m_{q_i+1,r(q_i+1)_\beta})$$

を計算している．このとき，この2人の署名者のものではない異なる署名鍵 $x'_{q_i,r(q_i)_\alpha}$，$x'_{q_i+1,r(q_i+1)_\beta}$ を用いて $x_{q_i,r(q_i)_\alpha}+x_{q_i+1,r(q_i+1)_\beta}=x'_{q_i,r(q_i)_\alpha}+x'_{q_i+1,r(q_i+1)_\beta}$ が成り立つなら，$u_{q_i,r(q_i)_\alpha}$，$u_{q_i+1,r(q_i+1)_\beta}$ は自分らが署名に参加したのではなく $x'_{q_i,r(q_i)_\alpha}$，$x'_{q_i+1,r(q_i+1)_\beta}$ を署名鍵として保有している署名者が参加したという主張を許すことになり，署名参加の否認が可能となる．

上記の結託攻撃に対しては，いずれの署名鍵も $\mathbb{Z}_p^*$ の要素であることから，どの2個の要素を選んでも和の値が異なるような鍵空間 $\mathbb{K}\subset\mathbb{Z}_p^*$ を準備し，署名鍵が $\mathbb{K}$ の要素となるようにすることで，回避することが可能となる．このような $\mathbb{K}$ を構成する単純な例としては，全署名者の署名鍵をその値が小さい順に並べたとき，超増加列（各数がそれまでの整数の和よりも大きくなる数列）になっている場合が考えられる．

**超増加列が $\mathbb{K}$ を構成できることの証明** $\mathbb{K}'\subset\mathbb{Z}_p^*$ となる $\mathbb{K}'$ の要素を小さい順から並べると，超増加列になっているものとする．任意の2個の鍵 $k_m, k_n\in\mathbb{K}'$（ただし $k_m<k_n$）を選んだとき，この鍵の和 $k_m+k_n$ と，少なくともどちらか1個が違う鍵のときの和の値とを比較する．比較対象の鍵の値が2個とも $k_n$ より小さい場合，超増加列の「$k_n$ より小さいすべての鍵の値の総和を求めても $k_n$ より小さい」という性質から，$k_n$ より小さい2個の鍵の和は $k_m+k_n$ より小さく，値は一致しない．また，1個の鍵が $k_n$ より大きい場合（仮に $k_o>k_n$ とおく），超増加列の「$k_o$ より小さいすべての鍵の値の総和を求めても $k_o$ より小さい」という性質から，$k_m+k_n<k_o$ が成り立つため，$k_m+k_n$ は $k_o$ を含んだ2個の鍵の和の値とも一致しない．したがって，$\mathbb{K}'$ は鍵空間 $\mathbb{K}$ の条件を満たす．

さらに巡回群においては，Erdös が効率良く $\mathbb{K}$ を構成できる（$\mathbb{K}$ の要素数を，単純な超増加列のときよりも多くすることができる）値の組合せに関する理論を提唱している [28]．

**偽装参加不能性：**偽装を行おうとする第三者の署名鍵を $x_{q_{\rm not},r(q_{\rm not})_\gamma}$ とする．このとき，ある署名参加者 $u_{q_i,r(q_i)_\alpha}$ が任意のメッセージ $m_{q_{\rm not},r(q_{\rm not})_\gamma}$，およびこのメッセージに対する第三者の BLS 署名 $x_{q_{\rm not},r(q_{\rm not})_\gamma}H(m_{q_{\rm not},r(q_{\rm not})_\gamma})$ を入手し，ルートに位置する署名者 $u_{1,r(1)_1}$ までのすべての手順が終了した後に

$$(x_{q_i,r(q_i)_\alpha}+x_{q_{\rm not},r(q_{\rm not})_\gamma})H(m_{q_{\rm not},r(q_{\rm not})_\gamma})$$

を計算すれば，$\sigma_{1,r(1)_1}$ に加算することでこの第三者を勝手に署名者として追加する，すなわちなりすますことが可能となる．

© 2012 Information Processing Society of Japan

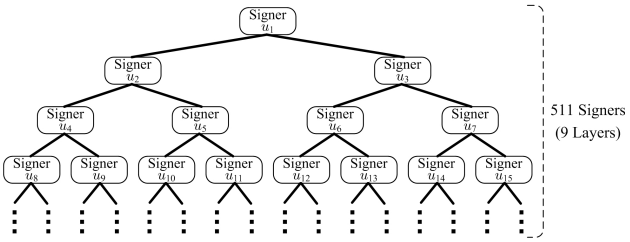


図 6 シミュレーションにおける階層構造
**Fig. 6** The binary tree structure in the simulation program.

上記の攻撃に対しては，各署名者が自分より下位の署名者の署名対象のメッセージ（の一部）を自分の署名対象のメッセージに追記しておくといった回避策で対抗することが可能となる．

### 5.2 実験結果

4.2 節で示したプロトコルについて，現在一般的な CGM サービスで行われている引用では，階層も数階層程度で引用するコンテンツ数も数個〜十数個であるのがほとんどだが，本章ではそれよりもはるかに負荷のかかる状況を想定し，**図 6** で示すような 2 分木 9 階層（署名者数：511）で構成されるアグリゲート署名ソフトを作成し，署名作成，および署名検証の実行時間を計測した．実験環境は以下のとおりである．

**OS**：Microsoft Windows 7 Ultimate（32 bit）．
**CPU**：Intel Core i7 870 2.93 GHz.
**メモリサイズ**：3.24 GB（32 bit 版の上限）．
**署名対象コンテンツ**：WMV（約 5.5 MB，22 秒）．
**Pairing Function**：Tate Pairing.
**一方向性ハッシュ関数**：*MapToGroup* with SHA-256.
**楕円曲線**：
素数 $p =$
0xD637ADBFE4F8637D8EE014EA1CD1167460087B0B.
$y^2 = x^3 + ax + b$
$a =$
0xD637ADBFE4F8637D8EE014EA1CD1167460087B08.
$b =$
0x143791687B84B21278CA3FEDCF598CDC503C2BA9.
ベースポイント $(s, t)$
$s =$
0x6DDC54524162D776AC39CFD5CABE79E9CF21C073.
$t =$
0xBAD5C1575061429186951DED22AC197D5B2FEF4C.
**署名鍵のサイズ**：20 [Byte].
**検証鍵のサイズ**：41 [Byte].
**署名サイズ**：241 [Byte]（6 次拡大体）．

この環境下で 20 回計測し，その平均値を示す（**表 1**）．

署名作成について，1 人が署名作成を行う時間は 10 msec 以下である．これは編集者が引用によりコンテンツ作成を行った際に署名作成を行うとしても，署名生成は 1 回の引用に対して 1 度行えばよいため，その作成頻度が頻繁でなければ，この処理速度は処理時間にほとんど影響がないと考えられる．

表 1 計測結果
**Table 1** The results on the simulation program.

| 署名者 1 人あたりの署名時間 | 検証時間 |
|---|---|
| ≤10 [msec]（計測限界値以下） | 13.0 [sec] |

署名検証について，図 6 で想定した署名者数・関係の場合にその処理時間が 13 秒である．ここで，この提案方式の利用シーンとして，CGM コンテンツの最小再生時間をテレビコマーシャル程度（15 秒）と仮定し，再生中に署名検証を行い，再生終了後にその検証結果をふまえて著作権者の表記を行うことを想定した場合，マルチコアの機能などにより再生処理と検証処理を同時に行える環境であれば，この処理時間は十分対応できていると考えられる．一方，先に著作権に関する署名の検証処理を行ってからコンテンツ再生を行うような利用シーンを想定した場合は，少なくとも 13 秒待機後にコンテンツ再生が始まることになる．また，再生後に著作権表記を行う利用シーンでも，コンテンツの再生時間が 13 秒より短い場合や，コンテンツの著作権者の検証のみを行う場合は，やはり検証処理のための待機時間が発生する．この待機時間が許容されるかどうか検証する必要があると同時に，検証時間を短縮するための処理の高速化を検討する必要がある．

## 6. まとめ

署名者の前後において，後者が自分とその前者のメッセージに署名し，この署名を順次合成していくことで構成された順序付きアグリゲート署名方式，およびその合成手順を 1 対多に拡張することで構成された木構造表記型アグリゲート署名方式について説明し，それを応用したコンテンツ引用過程表記手法について提案した．BLS 署名方式を拡張することで，特定のサーバの関与を必要とせずに，既存の署名鍵のみで署名者の署名順序や関係を保証できるアグリゲート署名の作成，および既存の検証鍵のみでの検証が可能であることを示した．

さらに実際に木構造表記型アグリゲート署名のソフトウェアを作成し，計測実験によりその実用性を確認した．

今回，この提案方式に関する安全性について，コンテンツ引用過程表記手法の観点から議論を行った．今後，一般的なアグリゲート署名の安全性の議論に基づき，提案したアグリゲート署名の安全性の考察を行い，必要であればさらなる改良を検討する予定である．

### 参考文献

[1] YouTube, available from ⟨http://www.youtube.com/⟩.

© 2012 Information Processing Society of Japan

[2] Creative Commons, available from ⟨http://creativecommons.org/⟩.
[3] 梶 克彦，長尾 確：Annphony：メタコンテンツ処理のためのプラットフォーム，情報科学技術レターズ –FIT 2006, Vol.5, pp.381–384, 電子情報通信学会 (2006).
[4] 梶 克彦，長尾 確：部分引用の管理に基づく Web コンテンツのマッシュアップ，情報処理学会第 69 回全国大会，5D-1 (2007).
[5] W3C, Resource Description Framework, available from ⟨http://www.w3.org/RDF/⟩ (1999).
[6] Itakura, K. and Nakamura, K.: A public-key cryptosystem suitable for digital multisignatures, *NEC Research & Development*, Vol.71, pp.1–8 (1983).
[7] Boneh, D., Gentry, C., Lynn, B. and Shacham, H.: Aggregate and Verifiably Encrypted Signatures from Bilinear Maps, *Advances in Cryptology – EUROCRYPT 2003*, LNCS 2656, pp.416–432, Springer-Verlag (2003).
[8] 齊藤泰一：順序指定可能な多重署名，暗号と情報セキュリティシンポジウム – SCIS '97，33A (1997).
[9] Boneh, D., Lynn, B. and Shacham, H.: Short Signatures from the Weil Pairing, *Advances in Cryptology – ASIACRYPT 2001*, LNCS 2248, pp.514–532, Springer-Verlag (2001).
[10] Boldyreva, A.: Threshold Signatures, Multisignatures and Blind Signatures Based on the Gap-Diffie-Hellman-Group Signature Scheme, *Public Key Cryptography – PKC 2003*, LNCS 2567, pp.31–46, Springer-Verlag (2003).
[11] Lin, C.Y., Wu, T.C., and Zhang, F.: A Structured Multisignature Scheme from the Gap Diffie-Hellman Group, *Cryptology ePrint Archive*, Report 2003/090, available from ⟨http://eprint.iacr.org/2003/090⟩ (2003).
[12] 稲村勝樹，田中俊昭：コンテンツの二次利用を実現する著作権保証方式，暗号と情報セキュリティシンポジウム – SCIS 2009，1B2-4 (2009).
[13] 稲村勝樹，渡辺 龍，田中俊昭：回覧文書閲覧確認に適した階層表記型多重署名方式の提案と実装評価，電子情報通信学会論文誌，Vol.J93-B, No.10, pp.1378–1387 (2010).
[14] Okamoto, T. and Pointcheval, D.: The Gap-Problems: A New Class of Problems for the Security of Cryptographic Schemes, *Public Key Cryptography – PKC 2001*, LNCS 1992, pp.104–118, Springer-Verlag (2001).
[15] Boneh, D. and Franklin, M.K.: Identity-Based Encryption from the Weil Pairing, *Advances in Cryptology – CRYPTO 2001*, LNCS 2139, pp.213–229, Springer-Verlag (2001).
[16] Joux, A. and Nguyen, K.: Separating Decision Diffie-Hellman from Diffie-Hellman in cryptographic groups, *Cryptology ePrint Archive*, Report 2001/003, available from ⟨http://eprint.iacr.org/2001/003⟩ (2001).
[17] Joux, A. and Nguyen, K.: Separating Decision Diffie-Hellman from Computational Diffie-Hellman in Cryptographic Groups, *Springer J. Cryptology*, Vol.16, No.4 pp.239–247 (2003).
[18] Joux, A.: A One Round Protocol for Tripartite Diffie-Hellman, *Algorithm Number Theory Symposium – ANTS IV*, LNCS 1838, pp.385–394, Springer-Verlag (2000).
[19] Washington, L.C.: *Elliptic Curves: Number Theory and Cryptography* (*Discrete Mathematics and Its Applications*), CRC Press (2003).
[20] Gentry, C. and Silverberg, A.: Hierarchical ID-based Cryptography, *Advances in Cryptology – ASIACRYPT 2002*, LNCS 2501, pp.548–566, Springer-Verlag (2002).
[21] Shamir, A.: Identity-Based Cryptosystems and Signature Schemes, *Advances in Cryptology – CRYPTO '84*, LNCS 196, pp.47–53, Springer-Verlag (1984).
[22] 今泉祥子，藤吉正明，渡邊 修，貴家仁志：結託攻撃耐性を有する JPEG 2000 の階層性を考慮したアクセス制御型暗号化法，暗号と情報セキュリティシンポジウム – SCIS 2006，4F1-5 (2006).
[23] 今泉祥子，藤吉正明，貴家仁志：再帰型ハッシュ連鎖を用いた暗号鍵生成方式と多次元階層的アクセス制御への適用，映像情報メディア学会誌，Vol.65, No.2, pp.193–202 (2011).
[24] Yamamoto, D. and Ogata, W: Structured Aggregate Signatures, *Symposium on Cryptography and Information Security – SCIS 2006*, 3A4-3 (2006).
[25] 白川瑞樹，岩村恵市：ボトムアップ型デジタルコンテンツ編集システム，情報処理学会研究報告，Vol.2010-CSEC-49, No.7, pp.1–6 (2010).
[26] Komano, Y., Ohta, K., Shimbo, A. and Kawamura, S.: Provably Secure Multisignatures in Formal Security Model and Their Optimality, *IEICE Trans. Fundamentals of Electronics, Communications and Computer Sciences*, Vol.E91-A, No.1, pp.107–118 (2008).
[27] Boldyreva, A.: Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme, *Cryptology ePrint Archive*, Report 2002/118, available from ⟨http://eprint.iacr.org/2002/118⟩ (2002).
[28] Erdös, P.: Some Applications of Ramsey's Theorem to Additive Number Theory, *Europ. J. Combinatorics*, Vol.1, pp.43–46 (1980).

**稲村 勝樹**

1972 年生．1998 年東京工業大学工学部有機材料工学科卒業．2000 年北陸先端科学技術大学院大学情報科学研究科博士前期課程修了．同年第二電電（株）（現 KDDI（株））入社，（株）京セラ DDI 未来通信研究所（現（株）KDDI 研究所）配属．2011 年東京理科大学大学院工学研究科博士後期課程入学．現在，KDDI（株），および同大学院に在籍．情報セキュリティに関する運用業務や研究開発に従事．電子情報通信学会，映像情報メディア学会各会員．

**岩村 恵市** （正会員）

1958 年生．1980 年九州大学工学部情報工学科卒業．1982 年同大学大学院情報工学研究科修士課程修了．同年キャノン（株）入社．1994 年東京大学博士（工学）．現在，東京理科大学工学部電気工学科教授．主に符号理論，並列処理，情報セキュリティ，電子透かしの研究に従事．IEEE，電子情報通信学会各会員，情報ハイディングおよびその評価基準（IHC）研究会委員長．

© 2012 Information Processing Society of Japan