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

2006－CSEC－34 (44)
2006／7／21

より強力な不正者に対する効率的なブラックボックス追跡のための階層的な鍵割り当て

松下 達之† 今井 秀樹††,†††

† 株式会社 東芝 研究開発センター 〒212-8582 神奈川県川崎市幸区小向東芝町 1
†† 中央大学理工学部 〒112-8551 東京都文京区春日 1-13-27
††† 産業技術総合研究所 情報セキュリティ研究センター 〒101-0021 東京都千代田区外神田 1-18-13
E-mail: †tatsuyuki.matsushita@toshiba.co.jp, ††h-imai@elect.chuo-u.ac.jp

あらまし 公開鍵ベースのブラックボックス追跡方式 [9] における送信オーバヘッドを改善する階層的な鍵割り当て方法を提案する．提案する鍵割り当て方法をこの従来方式に適用すると，従来方式単独の場合よりも，送信オーバヘッドと各受信者の所要メモリ量とのバランスが取れたブラックボックス追跡方式を構成できる．より具体的には，秘密鍵サイズが大きく増加することなく，暗号文サイズを $O(\sqrt{n})$ から $O(k+\log(n/k))$ へ削減できる．ここで，k と n はそれぞれ最大結託人数，全受信者数を表す．提案する鍵割り当て方法を適用した結果得られる方式は，従来方式と同様に，追跡を検知した場合に追跡を逃れる，より巧妙に作成された不正復号器に対しても（秘密情報不要で）ブラックボックス追跡可能である．

キーワード 階層的な鍵割り当て，ブラックボックス追跡，より強力な不正者．

Hierarchical Key Assignment for Efficient Public-Key Black-Box Tracing against Self-Defensive Pirates

Tatsuyuki MATSUSHITA† and Hideki IMAI††,†††

† Corporate Research & Development Center, Toshiba Corporation 1, Komukai Toshiba-cho, Saiwai-ku, Kawasaki 212-8582, Japan
†† Faculty of Science and Engineering, Chuo University 1-13-27 Kasuga, Bunkyo-ku, Tokyo 112-8551, Japan
††† Research Center for information Security, National Institute of Advanced Industrial Science and Technology 1-18-13 Sotokanda, Chiyoda-ku, Tokyo 101-0021, Japan
E-mail: †tatsuyuki.matsushita@toshiba.co.jp, ††h-imai@elect.chuo-u.ac.jp

Abstract We propose a hierarchical key-assignment method by which the transmission overhead in a public-key black-box tracing scheme presented in [9] can be improved. The previous scheme with our hierarchical key-assignment yields a better balance between the transmission overhead and each receiver's storage than the original one alone. More concretely, the ciphertext size can be reduced from $O(\sqrt{n})$ to $O(k+\log(n/k))$ without a substantial increase in the secret-key size, where k and n denote the maximum number of colluders in a coalition and the total number of receivers respectively. The resulting scheme, as well as the previous one, is black-box traceable (without any secret information) against a self-defensive pirate decoder that escapes from tracing if it detects itself being examined.

Key words Hierarchical key assignment, black-box tracing, self-defensive pirates.

1 はじめに

ディジタルコンテンツを加入者へ配信するサービスの増加に伴い，著作権保護の問題は今まで以上に重要となっている．コンテンツ配信システムとして，図 1 に示すシステムを考える．コンテンツ配信者は，セッション鍵を用いてコンテンツを暗号化する．また，コンテンツ配信者は，配信用鍵を用いてセッション鍵を暗号化する．その暗号文をヘッダと呼ぶ．暗号化コンテンツとヘッダは，サービス加入者（ユーザ）へ同報配信される．各ユーザには予め復号鍵（個人鍵）が与えられており，個人鍵

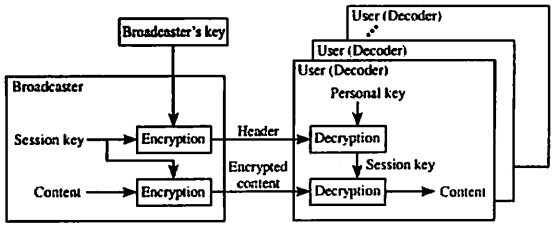


図 1　コンテンツ配信システム

Fig. 1　Content-distribution system

は復号器に格納される．ヘッダは復号器へ入力され，個人鍵を用いて復号される．この復号結果であるセッション鍵を用いて暗号化コンテンツを復号し，コンテンツを得る．

コンテンツ配信における不正行為として，海賊版復号器（不正復号器）の作成が考えられる．不正ユーザが自分の個人鍵を不正に用いて不正復号器を作成し，サービス非加入者に販売した場合，いわゆる「ただ見」が行われてしまい，コンテンツ配信者側に損害が生じる．不正行為を抑止するために，不正者追跡 [4] が研究されている．不正者追跡では，コンテンツ配信システムにおいて不正復号器が押収された際に，追跡者が追跡用鍵を用いて不正復号器の作成に加担した不正ユーザを特定する．（詳しくは 2.2 項において説明する．）

1.1　関連研究

方式の構成，配信用鍵と追跡用鍵の公開可能性，及び不正復号器への仮定の観点から関連研究を説明する．

不正者追跡方式はその構成により分類すると，組み合わせ論的な構成 [4]，木構造を用いた構成 [11]，代数的な構成 [1], [8]，ペアリングを用いた構成 [2], [3]，又はそれらの組み合わせが挙げられる．通常，組み合わせ論的な構成より代数的な構成の方が個人鍵サイズやヘッダサイズの観点において優れている．しかし，どの構成が最良であるかは一概には言えない．提案方式は，木構造を用いた構成と代数的な構成の組み合わせである．

配信用鍵と追跡用鍵の公開可能性はシステム拡張性に影響する．文献 [1], [8] の方式など，配信用鍵を公開できる場合，同一のシステムを複数のコンテンツ配信者が利用できるため，配信者側の拡張性があり望ましい．配信用鍵が秘密である方式（例えば [4]）においても，ヘッダ生成に用いる共通鍵暗号を公開鍵暗号に置き換えることで，配信用鍵を公開できる．配信用鍵が公開可能な方式の中で，追跡用鍵を秘密にする必要がある方式として文献 [2], [6] の方式がある．それに対し，文献 [3], [9] の方式では，追跡用鍵も公開できる．追跡用鍵を公開できる場合，追跡用鍵を知られてもシステムの安全性に影響を及ぼさないため，複数の追跡者に追跡を委託できる．これは追跡者側の拡張性があり望ましい．提案方式では，配信用鍵と追跡用鍵の両方を公開できる．その他，文献 [7], [12]〜[14] では，追跡結果を第三者に対して立証できる方式が研究されている．

不正ユーザを特定する際に，不正復号器をこじ開けて中身を調べる場合と，不正復号器をブラックボックスとしてその入出力を観測するのみの場合がある．後者は，ブラックボックス追跡と呼ばれ，不正復号器の実装形態に依らない追跡が可能となり，望ましい．ブラックボックス追跡における不正復号器への仮定として，以下の三点がある．

(1-1) 不正復号器は，常に出力する．(1-2) 不正復号器は，追跡を検知した場合，自己防衛機構を作動させ，その後の入力を一切受け付けない(注1)．

(2-1) 不正復号器はリセット可能であり，テスト（一組の入出力を観測すること）は各回独立に行うことができる．(2-2) 不正復号器は，過去の入出力を記憶しており，それに基づいて動作する．前者を仮定した追跡方式を，後者を仮定した追跡方式に電子透かしを導入して変換する方法が文献 [5] において示されているため，前者を仮定することが多い．

(3-1) 追跡者は，不正復号器から出力されるセッション鍵の値を観測できる．(3-2) 追跡者は，コンテンツが正しく再生されたか否か以外を観測できない．

提案方式では，より巧妙に作成された不正復号器を想定し，(1-2)，(2-1)，及び (3-2) を仮定する．(1-2) 及び (2-1) を仮定する不正復号器は，文献 [5] における "type-2" 不正復号器とみなすことができる．上記三つを仮定した不正復号器に対するブラックボックス追跡方式として文献 [1], [2], [5], [8], [9] の方式が知られている．文献 [1], [8] の方式では，（何らかの方法により）事前に容疑者が最大結託人数以下に絞り込まれていなければならないという問題がある．文献 [5] の方式では，この絞り込みが不要で，かつヘッダサイズを $O(\sqrt{n})$ とすることが可能である．ここで，n は全ユーザ数である．しかし，追跡アルゴリズムから出力される容疑者リストのみから不正ユーザを正しく特定する確率とヘッダサイズがトレードオフの関係にあるという問題がある．文献 [2], [9] の方式では，両方の問題を解決し，ヘッダサイズが $O(\sqrt{n})$ であるブラックボックス追跡方式が示されている．特に，文献 [2] の方式は，許容する結託人数に上限がなく，また，復号に要する計算コストが一定であるという優れた性能を有する．

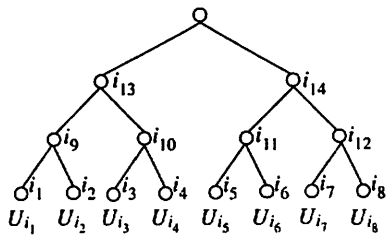
1.2　成　　果

文献 [2], [9] の方式のヘッダサイズは $O(\sqrt{n})$ であるが，これは効率的であるとは言えず，より削減されることが望ましい．本稿では，個人鍵サイズが $O(1)$ から $O(\log(n/k))$ へ増加することを許容し，ヘッダサイズを $O(\sqrt{n})$ から $O(k+\log(n/k))$ へ削減できる，配信用鍵と追跡用鍵が公開可能なブラックボックス追跡方式を提案する．ここで k は最大結託人数である．

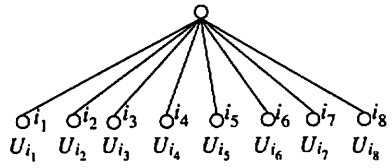
提案方式は文献 [9] の方式の拡張であり，文献 [9] の方式との主な違いは個人鍵の割り当て方法にある．ここで，文献 [9] の方式を単純に拡張するのみではヘッダサイズ削減を達成できない．（詳しくは 3 節において説明する．）

文献 [2] の方式と比較すると，提案方式では，結託耐性への要求を緩めることにより，ヘッダサイズを削減しているとみなせる．提案方式は文献 [9] の方式に基づいているため，復号に要する計算コストが $O(k)$ となってしまうが，（文献 [2] の方式とは異なり）追跡用鍵を公開できる．

(注1)：本稿では，簡単のため，不正復号器が追跡を検知した場合，確定的に自己防衛機構が作動すると仮定する．確率的な場合については稿を改める．



(a) 完全二分木の場合



(b) 深さが 1 の場合

図 2 T ($L = 8$) の構造
Fig. 2 Structure of T ($L = 8$)

本稿の構成を以下に示す．2 節において定義を行い，3 節において解決すべき問題を明確にする．4 節において提案方式を述べ，その安全性と効率性を 5 節において分析する．最後に 6 節において結論を述べる．

2 定　　義

2.1 木 構 造

本稿において用いられる木構造に関する表記を以下に定義する．

[定義 1]（木構造に関する表記） 木構造における各節をノード，最上位にある（親ノードのいない）ノードをルート，最下位にある（子ノードのいない）ノードをリーフ，親ノードが同じノードを兄弟ノードと呼ぶ．また，ルートからノードまでの段数（枝の数）をそのノードの深さと呼ぶ．L 個のリーフを持つ木を T と表す．T におけるルートを除く全てのノード（リーフも含む）の集合を $\mathcal{N}_T$ と表す．簡単のため，$\mathcal{N}_T = \{0, \ldots, |\mathcal{N}_T| - 1\}$ とする．あるノード v ($v \in \mathcal{N}_T$) について，v を祖先に持つ（v を親に持つ場合や v 自身がリーフである場合も含む）全てのリーフにそれぞれ対応するユーザ集合を $\mathcal{U}_v$ と表す．与えられた T に対して，ユーザ集合族 Y_T を $Y_T = \{(0, \mathcal{U}_0), \ldots, (|\mathcal{N}_T| - 1, \mathcal{U}_{|\mathcal{N}_T|-1})\}$ と定義する．

$L = 8$ の場合の T の構造を図 2 に例示する．例えば，図 2(a) において，$\mathcal{N}_T = \{0, \ldots, 2L - 3\ (= 13)\}$ であり，$\mathcal{U}_{i_1}$ は一番左のリーフ i_1 に対応するユーザ集合を表し，$\mathcal{U}_{i_9} = \mathcal{U}_{i_1} \cup \mathcal{U}_{i_2}$ である．図 2(b) において，$\mathcal{N}_T = \{0, \ldots, L - 1\ (= 7)\}$ である．

2.2 ブラックボックス追跡

ブラックボックス追跡方式は，以下の四つの処理から成る．

鍵生成 信頼できる第三者は個人鍵を生成し，各ユーザへ秘密に配布する．個人鍵は復号器に格納される．

暗号化 コンテンツ配信者は，セッション鍵を暗号化する．次に，コンテンツ配信者は，ヘッダを同報配信する．ここで，混乱を避けるために，コンテンツ暗号化に用いる暗号アルゴリズムは安全であり，公開されていると仮定し，セッション鍵を用いたコンテンツ暗号化と同報配信については省略し，ヘッダの構成に焦点を当てる．また，同報通信路は受信された情報が改変されていないという意味において信頼できると仮定する．

復号 ユーザは，受信したヘッダを復号器へ入力し，セッション鍵を計算する．

ブラックボックス追跡 不正復号器が押収されたとする．追跡者は，追跡用ヘッダを不正復号器に入力し，正しく復号されたか否かを観測する．その出力結果に基づき，不正ユーザを特定する．

上記モデルに基づく，ブラックボックス追跡方式を以下に定義する．

[定義 2]（ブラックボックス追跡方式） ブラックボックス追跡方式は，以下に示す四つの多項式時間アルゴリズム (Gen, Enc, Dec, BBT) から成る．

Gen: 鍵生成アルゴリズム Gen は確率的アルゴリズムである．入力として，セキュリティパラメータ ℓ，全ユーザ数 n，及び最大結託人数 k を取る．出力は配信用鍵 BK，各ユーザの個人鍵 d_{u_i} ($i = 1, \ldots, n$)，追跡用鍵 TK，及びユーザ集合族 Y である．

Enc: 暗号化アルゴリズム Enc は確率的アルゴリズムである．入力として，BK，Y，及びセッション鍵 s を取る．出力はヘッダ H である．

Dec: 復号アルゴリズム Dec は確定的アルゴリズムである．入力として，d_{u_i} 及び H を取る．出力は s 又は正しくないセッション鍵である．なお，全てのセッション鍵について，$\mathrm{Dec}(d_{u_i}, \mathrm{Enc}(BK, Y, s)) = s$ が要求される．

BBT: ブラックボックス追跡アルゴリズム BBT は確率的アルゴリズムである．入力として，TK，Y，及び不正復号器 PD を取る．出力は不正ユーザ ID u_j である．

最も単純な Y の例は，$\mathcal{U}$を全ユーザ集合として $Y = \{\mathcal{U}\}$ であるため通常 Y は省略されるが，3 節において従来方式と提案方式の違いを説明するために明示的に記述する．本稿では，Y が木構造により表され，BK と TK が公開可能なブラックボックス追跡方式を考える．

2.3 安 全 性

以下に定義する識別不可能性とブラックボックス追跡可能性を満たすとき，ブラックボックス追跡方式は安全であると言う．

[定義 3]（識別不可能性） Π = (Gen, Enc, Dec, BBT) をブラックボックス追跡方式とする．ヘッダが与えられたとき，非ユーザ（盗聴者）がそのヘッダに対応するセッション鍵とランダムなセッション鍵を有意な確率で識別することが不可能であるならば，Π は識別不可能性を満たすと言う．

[定義 4]（ブラックボックス追跡可能性） Π=(Gen, Enc, Dec, BBT) をブラックボックス追跡方式とし，PD を不正復号器と

する．高々k人の不正ユーザによりPDが作成されたとき，PDの入出力を観測することのみにより少なくとも一人の不正ユーザを圧倒的な確率で正しく特定できるならば，Πはブラックボックス追跡可能性を満たすと言う．

3 従来方式

始めに文献[9]の方式を述べる．次に，ヘッダサイズ削減への試みとしてこれを単純に拡張した方式ではヘッダサイズが削減されないことを説明し，解決すべき問題を明確にする．

3.1 文献[9]の方式

$\mathsf{Gen}(1^\ell, n, k)$: $|q| = \ell$，$q|p-1$，かつ$q \geqq n+2k-1$を満たす二つの素数p, qを生成する．G_qを$\mathbb{Z}_p^*$の位数qの部分群とする．G_qの生成元gを生成する．本稿では，特に断りのない限り計算は$\mathbb{Z}_p^*$上で行われるものとする．

$\mathcal{U}$ ($\mathcal{U} \subseteqq \mathbb{Z}_q \backslash \{0\}$)を全ユーザ集合とする．$\mathcal{U}$を要素数が$2k$以下であり，かつ互いに素な部分集合に分割する．分割されたユーザ部分集合数をLとする．ルートとL個のリーフのみから成る，深さ1のL分木Tを定め，ユーザ集合族Y_Tを生成する．Tを図示すると図2(b)となり，分割された各ユーザ部分集合とリーフは1対1に対応している．

$a_0, \ldots, a_{2k-1}, b_0, \ldots, b_{L-1} \in_R \mathbb{Z}_q$を選択し，公開鍵$e$ ($= BK = TK$)を次式により計算する．

$$e = (p, q, g, \{y_{0,i} = g^{a_i}\}_{i=0,\ldots,2k-1}, \{y_{1,i} = g^{b_i}\}_{i=0,\ldots,L-1}).$$

$\mathcal{U}_v$に属するユーザuの個人鍵は$d_u = (u, v, f_v(u))$と表される．

$$f_v(u) = \sum_{i=0}^{2k-1} a_{v,i} u^i \bmod q, \tag{1}$$

$$a_{v,i} = \begin{cases} a_i & (i \neq v \bmod 2k), \\ b_v & (i = v \bmod 2k). \end{cases}$$

$\mathsf{Enc}(e, Y_T, s)$: $s \in_R G_q$であるとする．$r_0, r_1 \in_R \mathbb{Z}_q$を選択する．$T$の各リーフ$v_j$ ($1 \leqq j \leqq t\ (= L)$)について，r_{v_j}にr_0又はr_1を代入し，次式によりH_{v_j}を計算する．

$$H_{v_j} = (g^{r_{v_j}}, h_{v_j,0}, \ldots, h_{v_j,2k-1}),$$

$$h_{v_j,i} = \begin{cases} y_{0,i}^{r_{v_j}} & (i \neq v_j \bmod 2k), \\ s y_{1,v_j}^{r_{v_j}} & (i = v_j \bmod 2k). \end{cases}$$

得られた$(H_{v_1}, \ldots, H_{v_t})$をヘッダ$H$とする．

$\mathsf{Dec}(d_u, H)$: ユーザuは$\mathcal{U}_{v_j}$に属しているとする．ユーザuは，次式によりH_{v_j}からセッション鍵sを計算する．

$$\left\{ \prod_{i=0}^{2k-1} h_{v_j,i}^{u^i} \Big/ (g^{r_{v_j}})^{f_{v_j}(u)} \right\}^{1/u^{v_j} \bmod 2k}$$

$$= s \left(g^{r_{v_j} \sum_{i=0}^{2k-1} a_{v_j,i} u^i} \Big/ g^{r_{v_j} f_{v_j}(u)} \right)^{1/u^{v_j} \bmod 2k}$$

$$= s.$$

$\mathsf{BBT}(e, Y_T, PD)$: 始めに概要を説明し，次に具体的な処理を述べる．追跡者は，ユーザが不正ユーザであるかを一人ずつ検査していく．j回目 ($j = 1, \ldots, n$)の検査において，追跡者はユーザu_jを選択し，ユーザ$u_1, \ldots, u_j$のみが無効化されているヘッダを生成する．無効化ユーザ集合が$\mathcal{X}$であるヘッダの復号結果が正しく，かつ無効化ユーザ集合が$\mathcal{X} \cup \{u\}$であるヘッダの復号結果が正しくない場合，ユーザuを不正ユーザと決定する．

Tにおけるリーフを左から順に$i_1, \ldots, i_L$とし，簡単のため，$|\mathcal{U}_{i_1}| = \cdots = |\mathcal{U}_{i_L}| = 2k$，$n = 2kL$とする．各ユーザ部分集合の要素を次式によりラベル付けする．

$$\mathcal{U}_{i_j} = \{u_{2k(j-1)+1}, \ldots, u_{2kj}\}\ (1 \leqq j \leqq L). \tag{2}$$

$1 \leqq j \leqq n$について，以下の処理を繰り返す．

- ctr_jに0を代入し，以下のテストをm回繰り返す(注2)．
  1. セッション鍵$s \in_R G_q$を生成し，$\mathcal{X} = \{u_1, \ldots, u_j\}$として，ヘッダ$H$を生成する．($H$の具体的な計算方法は省略する．)
  2. 不正復号器PDにHを入力し，その出力を観測する．
  3. PDが正しく復号した場合，ctr_jの値を1だけ増す．(PDにおいて自己防衛機構が作動した場合，ユーザu_jを不正ユーザと決定する．)

最後に，$ctr_{j-1} - ctr_j$が最大となる整数$j \in \{1, \ldots, n\}$を求め，ユーザu_jを不正ユーザと決定する．ここで，$ctr_0 = m$とする．

3.2 ヘッダサイズ削減の試み

文献[9]の方式において想定されている木構造は図2(b)に示される深さ1の木であり，この方式をワンレベル方式とみなすと，図2(a)に示される木構造を想定し，文献[11]の枠組(Complete Subtree method)を用いて，文献[9]の方式を以下に概略を述べるマルチレベル方式に拡張すれば，ヘッダサイズ削減が期待できる(注3)．

$\mathsf{Gen}'(1^\ell, n, k)$: 図2(a)に示す完全二分木$T$を定め，ユーザ集合族$Y_T$を生成する．ユーザ$u$の個人鍵$d_u$は，$d_u = \{(u, v, F_v(u)) \mid v \in \mathcal{N}_T,\ u \in \mathcal{U}_v\}$である．例えば，図2(a)において，$u \in \mathcal{U}_{i_1}$の場合，$d_u = \{(u, i_1, F_{i_1}(u)), (u, i_9, F_{i_9}(u)), (u, i_{13}, F_{i_{13}}(u))\}$となる．

鍵生成多項式Fの単純な構成方法として，二つの方法が考えられる．一つは，全てのノードに対して，文献[9]の方式の単一システムから生成される鍵生成多項式を用いることである．例えば，$d_u = \{(u, i_1, f_{i_1}(u)), (u, i_9, f_{i_9}(u)), (u, i_{13}, f_{i_{13}}(u))\}$となる（鍵生成多項式$f$は式(1)において定義されている）．しかし，この方法では，結託者が各自の個人鍵を持ち寄り，$a_{v,i}$を未知数とする連立方程式を立て，それを解くことにより$a_{v,i}$の値が判明するため，安全ではない．

もう一つは，文献[9]の方式の複数システムを運用し，異なる深さのノードに対しては，それぞれ異なるシステムから生成された鍵生成多項式を用いることである．例えば，

(注2)：文献[5]の結果を用いると，$m = O(n^2 \log^2 n)$である．

(注3)：文献[11]には，もう一つの枠組(Subset Difference method)があるが，これを提案方式へ効率的に適用することは難しいと思われる．

$d_u = \{(u, i_1, f_{i_1}^{(3)}(u)), (u, i_9, f_{i_9}^{(2)}(u)), (u, i_{13}, f_{i_{13}}^{(1)}(u))\}$ となる．ここで，$f^{(\delta)}$ は深さ δ のノードに対する鍵生成多項式 f を表す．$\gamma \neq \delta$ である場合，$f^{(\gamma)}$ と $f^{(\delta)}$ の同一次数の係数は（圧倒的な確率で）異なるため，前述した結託攻撃は不可能である．以下，$f^{(\delta)}$ に対応する公開鍵を $e^{(\delta)}$ として，この方法を用いた場合を考える．

$\mathsf{Enc}'((e^{(1)}, \ldots, e^{(\log_2 L)}), Y_T, s)$: 以下に示すノード選択アルゴリズム Sel を実行する．入力は Y_T であり，出力は選択されたノードの ID である．

$\mathsf{Sel}(Y_T)$: 以下の三つの条件を満たす，T における $\log_2 L + 1$ 個のノードを選択する(注4)．(1) $\log_2 L + 1$ 個のノードの内，$\log_2 L - 1$ 個はリーフではないノードであり，残りは兄弟リーフである．(2) 上記 $\log_2 L - 1$ 個のノードの深さは互いに異なる．(3) 選択されるノードを $v_1, \ldots, v_{\log_2 L + 1}$ とすると，$\cup_{i=0}^{\log_2 L+1} \mathcal{U}_{v_i} = \mathcal{U}$ であり，かつ $i \neq j$ ならば $\mathcal{U}_{v_i} \cap \mathcal{U}_{v_j} = \emptyset$ である．図 2(a) においては，選択されるノード数は 4 ($= \log_2 8 + 1$) であり，例えば，ノード i_1, i_2, i_{10}, i_{14} が選択できる．

3.1 項に示した Enc を実行する．ただし，以下の二点が Enc と異なる．(1) H_{v_j} は選択されたノード v_j ($1 \leq j \leq t\ (= \log_2 L + 1)$) について計算する．(2) 選択されたノード v_j の深さが δ のとき，次式により表される公開鍵 $e^{(\delta)}$ を用いる．

$$e^{(\delta)} = (p^{(\delta)}, q^{(\delta)}, g^{(\delta)}, \{y_{0,i}^{(\delta)}\}_{i=0,\ldots,2k-1}, \{y_{1,i}^{(\delta)}\}_{i=0,\ldots,2^\delta-1}).$$

選択されたノード v_j の深さ δ により，H_{v_j} はそれぞれ異なる公開鍵 $e^{(\delta)}$ から計算されるため，ヘッダサイズは $O(k \log L)$ となる．図 3 に示す通り，これは非効率的である．

以上を要約すると，文献 [9] の方式の単純な拡張における問題は，(1) 単一システムから生成される鍵生成多項式のみを用いると，結託攻撃が存在することと，(2) 複数システムを運用することで結託攻撃は回避できるが，ヘッダサイズが非効率的となってしまうことである．次節において，結託攻撃に耐性を有し，かつヘッダサイズ削減が可能な方式を提案する．

4 提案方式

提案方式における主なアイデアは，Gen'' における階層的な個人鍵割り当て方法である．$\mathsf{Enc}'', \mathsf{Dec}'', \mathsf{BBT}''$ については，提案する階層的な個人鍵割り当て方法に対応させた形に，文献 [9] の方式を調整したものとみなせる．

$\mathsf{Gen}''(1^\ell, n, k)$: 3.1 項に示した Gen と同じく，素数 p, q と生成元 g を生成し，全ユーザ集合 $\mathcal{U}$ を L 個に分割する．

図 2(a) に示す完全二分木 T を定め，ユーザ集合族 Y_T を生成する．$a_i, b_i \in_R \mathbb{Z}_q$ ($0 \leq i \leq 2k-1$) と $c_i, \lambda_i \in_R \mathbb{Z}_q$ ($0 \leq i \leq 2L-3$) を選択し，公開鍵 e ($= BK = TK$) を次式により計算する．

$$e = (p, q, g, \{y_{0,i} = g^{a_i}\}_{i=0,\ldots,2k-1}, \{y_{1,i} = g^{c_i}\}_{i=0,\ldots,2L-3},$$
$$\{y_{2,i} = g^{\lambda_i}\}_{i=0,\ldots,2L-3}).$$

次に個人鍵生成多項式 $A_v(x), B(x)$ を定義する．

$$A_v(x) = \sum_{i=0}^{2k-1} (a_{v,i} - \lambda_v b_i) x^i \bmod q,$$

$$B(x) = \sum_{i=0}^{2k-1} b_i x^i \bmod q,$$

$$a_{v,i} = \begin{cases} a_i & (i \neq v \bmod 2k), \\ c_v & (i = v \bmod 2k), \end{cases}$$

ここで，$A_v(x), B(x)$ においては以下の関係が成立し，これは復号アルゴリズムにおいて用いられる．

$$A_v(x) + \lambda_v B(x) = \sum_{i=0}^{2k-1} a_{v,i} x^i \bmod q. \tag{3}$$

ユーザ u の個人鍵 d_u は $d_u = \{(u, v, A_v(u), B(u)) | v \in \mathcal{N}_T, u \in \mathcal{U}_v\}$ と表される．例えば，図 2(a) において，$u \in \mathcal{U}_{i_1}$ の場合，$d_u = \{(u, i_1, A_{i_1}(u), B(u)), (u, i_9, A_{i_9}(u), B(u)), (u, i_{13}, A_{i_{13}}(u), B(u))\}$ となる．

上記の個人鍵割り当て方法を導入した理由を説明する．k 人の結託者 $x_1, \ldots, x_k$ が同一のユーザ部分集合 $\mathcal{U}_{v_1}$ に属しているとする．ここで，v_1 は T におけるリーフである．このとき，v_1 の祖先ノードを $v_2, \ldots, v_{\log_2 L}$ とすると，結託者は $\mathcal{U}_{v_2}, \ldots, \mathcal{U}_{v_{\log_2 L}}$ にも全員属している．結託攻撃として，結託者は他のユーザの個人鍵を作り出すために，次に示す連立方程式 ($1 \leq j \leq \log_2 L$) を解くことにより，個人鍵生成多項式の係数を求めようとする．

$$\begin{cases} A_{v_j}(x_1) + \lambda_{v_j} B(x_1) &= \sum_{i=0}^{2k-1} a_{v_j,i} x_1^i, \\ A_{v_j}(x_2) + \lambda_{v_j} B(x_2) &= \sum_{i=0}^{2k-1} a_{v_j,i} x_2^i, \\ \qquad\qquad \vdots & \\ A_{v_j}(x_k) + \lambda_{v_j} B(x_k) &= \sum_{i=0}^{2k-1} a_{v_j,i} x_k^i. \end{cases} \tag{4}$$

しかし，式 (4) において，式の数が未知数の数より多い場合でも係数は不定である．従って，結託攻撃は不可能である．

$\mathsf{Enc}''(e, Y_T, s)$: 3.2 項に示した Sel を実行し，3.1 項に示した Enc を実行する．ただし，以下の二点が Enc と異なる．(1) H_{v_j} は選択されたノード v_j ($1 \leq j \leq t\ (= \log_2 L + 1)$) について計算する．(2) $H_{v_j} = (g^{r_{v_j}}, y_{2,v_j}^{r_{v_j}}, h_{v_j,0}, \ldots, h_{v_j,2k-1})$ とする．

本稿では，任意のユーザを無効化することを考えない(注5)．ブラックボックス追跡に焦点を当てると，追跡の際には容疑者を一人ずつ検査する（無効化ユーザ集合 $\mathcal{X}$ に一人ずつ加えていく）ため，任意のユーザ集合の無効化は必ずしも要求されない．

$\mathsf{Dec}''(d_u, H)$: ユーザ u は，$(u, v_j, A_{v_j}(u), B(u)) \in d_u$ と H_{v_j} を用いて，次式によりセッション鍵 s を計算する．

$$\left[\prod_{i=0}^{2k-1} h_{v_j,i}^{u^i} \Big/ \left\{ (g^{r_{v_j}})^{A_{v_j}(u)} \left(y_{2,v_j}^{r_{v_j}}\right)^{B(u)} \right\} \right]^{1/u^{v_j} \bmod 2k}$$

(注4)：配信用ヘッダと追跡用ヘッダにおいて H_{v_j} の数が異なる場合，その違いから両者を識別できる可能性がある．この選択方法は，両者において H_{v_j} の数を同一にする一つの方法である．

(注5)：文献 [10] のユーザ無効化方法を提案方式に適用することにより，任意のユーザ集合の無効化を達成できる．

$$= s\left\{g^{r_{v_j}\sum_{i=0}^{2k-1} a_{v_j,i}u^i} \Big/ g^{r_{v_j}\left(A_{v_j}(u)+\lambda_{v_j}B(u)\right)}\right\}^{1/u^{v_j} \bmod 2k}$$

$$= s \quad (\because \text{式 (3)}).$$

BBT″(e, Y_T, PD): 3.1 項に示した BBT を実行する．ただし，以下に示す Enc‴ を実行することにより追跡用ヘッダを計算する．

Enc‴(e, Y_T, s): Enc″ を実行する．ただし，以下の二点が Enc″ と異なる．(1) ノードの選択に以下の二つの条件を加える．(i) $0 < |\mathcal{U}_{v_j}\backslash\mathcal{X}| < 2k$，$\mathcal{X}\cap\mathcal{U}_{v_j} = \emptyset$，又は $\mathcal{X}\cap\mathcal{U}_{v_j} = \mathcal{U}_{v_j}$ である．(ii) $0 < |\mathcal{U}_{v_j}\backslash\mathcal{X}| < 2k$ であるノード v_j は高々一つ選択される．図 2(a) において，$\mathcal{X} = \{u_1, \ldots, u_{7k}\}$ の場合，例えば，四つのノード i_3, i_4, i_9, i_{14} ($\mathcal{X}\cap\mathcal{U}_{i_{14}} = \emptyset$，$\mathcal{X}\cap\mathcal{U}_{i_9} = \mathcal{U}_{i_9}$，$\mathcal{X}\cap\mathcal{U}_{i_3} = \mathcal{U}_{i_3}$，$0 < |\mathcal{U}_{i_4}\backslash\mathcal{X}|$ $(= |\{u_{7k+1}, \ldots, u_{8k}\}|) < 2k$) が選択される．(2) 以下の処理を加える(注6)．

- $\mathcal{X}\cap\mathcal{U}_{v_j} = \emptyset$ の場合，選択された $\log_2 L + 1$ 個のノードの中に，$0 < |\mathcal{U}_v\backslash\mathcal{X}| < 2k$ であるノード v が存在する場合，$r_{v_j} = r_0$ と代入する．そうでない場合，r_{v_j} に r_0 又は r_1 を代入する．
- $0 < |\mathcal{U}_{v_j}\backslash\mathcal{X}| < 2k$ の場合，$\mathcal{U}_{v_j}\backslash\mathcal{X} = \{x_1, \ldots, x_m\}$ とし，$2k - m - 1 > 0$ である場合，$2k - m - 1$ 個の異なる要素 $x_{m+1}, \ldots, x_{2k-1} \in_R \mathbb{Z}_q\backslash(\mathcal{U}\cup\{0\})$ を選択する．$1 \leqq t \leqq 2k-1$ について $\sum_{i=0}^{2k-1} L_i x_t^i = 0 \bmod q$ を満たす要素 $L_0, \ldots, L_{2k-1} \in \mathbb{Z}_q$ を求める．最後に，次式により $h_{v_j,i}$ を計算し，置き換える．

$$h_{v_j,i} = \begin{cases} g^{L_i y_{0,i}^{r_{v_j}}} & (i \neq v_j \bmod 2k), \\ s g^{L_i y_{1,i}^{r_{v_j}}} & (i = v_j \bmod 2k). \end{cases}$$

ここで，$r = r_1$ と代入する．

$\mathcal{U}_{v_j}\cap\mathcal{X}$ に属するユーザ u が無効化される仕組を説明する．ユーザ u は次式によりセッション鍵を計算しようとする．

$$\left[\prod_{i=0}^{2k-1} h_{v_j,i}^{u^i} \Big/ \left\{\left(g^{r_{v_j}}\right)^{A_{v_j}(u)}\left(y_{2,v_j}^{r_{v_j}}\right)^{B(u)}\right\}\right]^{1/u^{v_j} \bmod 2k}$$

$$= s\left\{g^{\sum_{i=0}^{2k-1} L_i u^i g^{r_{v_j}\sum_{i=0}^{2k-1} a_{v_j,i}u^i}} \Big/ g^{r_{v_j}\left(A_{v_j}(u)+\lambda_{v_j}B(u)\right)}\right\}^{1/u^{v_j} \bmod 2k}.$$

しかし，$\sum_{i=0}^{2k-1} L_i u^i = 0 \bmod q$ が成立しないため，正しいセッション鍵は計算できない．

- $\mathcal{X}\cap\mathcal{U}_{v_j} = \mathcal{U}_{v_j}$ の場合，$z_{v_j} \in_R \mathbb{Z}_q$ を選択し，r_{v_j} に r_0 又は r_1 を代入する．次式により h_{v_j} を計算する．
  - 選択された $\log_2 L + 1$ 個のノードの中に，$0 < |\mathcal{U}_v\backslash\mathcal{X}| < 2k$ であるノード v が存在する場合，

$$h_{v_j,i} = \begin{cases} y_{0,i}^{r_{v_j}} & (i \neq v_j \bmod 2k,\ r_{v_j} = r_0), \\ g^{L_i y_{0,i}^{r_{v_j}}} & (i \neq v_j \bmod 2k,\ r_{v_j} = r_1), \\ g^{z_{v_j}} & (i = v_j \bmod 2k). \end{cases}$$

(注6)：これ以降の処理も文献[9]の方式と同様である

  - そうでない場合，$h_{v_j, v_j \bmod 2k}$ を $g^{z_{v_j}}$ に置き換える．

$\mathcal{U}_{v_j}$ に属するユーザのみが復号に用いる要素をランダムな要素に置き換えることにより，$\mathcal{U}_{v_j}$ に属するユーザ全員を無効化できる．

5 分　　析

提案方式の安全性と効率性について議論する．

5.1 安 全 性

提案方式の安全性は Diffie-Hellman 判定 (decision Diffie-Hellman, DDH) 問題の困難性に基づいている．始めに，提案方式の識別不可能性を証明する．

[定理 1]（識別不可能性） G_q における DDH 問題が困難であるという仮定の下で，提案方式は定義 3 において定められた識別不可能性を満たす．

[証明] 確率的多項式時間アルゴリズム A, B に対して，A が存在すれば B も存在することを A ⇒ B と表記する．また，A ⇒ B かつ B ⇒ A であることを A ⇔ B と表記する．Dis を非ユーザがヘッダに対応するセッション鍵と G_q 上のランダムな要素を識別するための確率的多項式時間アルゴリズムとし，DDH を G_q 上の DDH 問題を解く確率的多項式時間アルゴリズムとする．DDH への入力は，DDH 問題のチャレンジ入力 $(g_1, g_2, g_3, g_4) = (g_1, g_2, g_1^a, g_2^b)$ $(g_1, g_2 \in_R G_q, a, c \in_R \mathbb{Z}_q)$ である．$b = a$ である (g_1, g_2, g_3, g_4) を Diffie-Hellman 組と呼び，$b = c$ である (g_1, g_2, g_3, g_4) をランダム組と呼ぶ．証明すべきことは Dis ⇔ DDH である．

DDH ⇒ Dis は明らかである．Dis ⇒ DDH について，Dis を用いて DDH を構成できることを示し，これを証明する．

DDH(g_1, g_2, g_3, g_4): $a_0, \ldots, a_{2k-1}, \alpha_0, \ldots, \alpha_{2L-3}, \beta, \lambda_0 \ldots, \lambda_{2L-3} \in_R \mathbb{Z}_q$ を選択し，e を次式により計算する．

$$e = (p, q, g_1, \{y_{0,i} = g_1^{a_i}\}_{i=0,\ldots,2k-1}, \{y_{1,i} = g_1^{\alpha_i} g_2^{\beta}\}_{i=0,\ldots,2L-3},$$
$$\{y_{2,i} = g_1^{\lambda_i}\}_{i=0,\ldots,2L-3}).$$

$s \in_R G_q$ を選択し，4 節に示した Enc″ を実行する．ただし，r_{v_j} に r_0 又は 0 を代入し，H_{v_j} を次式により計算する．

$$H_{v_j} = \left(g_1^{r_{v_j}} g_3, \left(g_1^{r_{v_j}} g_3\right)^{\lambda_{v_j}}, h_{v_j,0}, \ldots, h_{v_j,2k-1}\right),$$

$$h_{v_j,i} = \begin{cases} \left(g_1^{r_{v_j}} g_3\right)^{a_i} & (i \neq v_j \bmod 2k), \\ s\left(g_1^{r_{v_j}} g_3\right)^{\alpha_i}\left(g_2^{r_{v_j}} g_4\right)^{\beta} & (i = v_j \bmod 2k). \end{cases}$$

ここで，(g_1, g_2, g_3, g_4) が Diffie-Hellman 組である場合，ヘッダに対応するセッション鍵は s である．そうでない場合，セッション鍵は G_q 上のランダムな要素である．

Dis に e, H, s を与える．Dis が，s は H に対応すると判定した場合，"Diffie-Hellman 組" と出力する．そうでない場合，"ランダム組" と出力する．Dis はヘッダに対応するセッション鍵と G_q 上のランダムな要素を識別可能であるから，DDH は与えられた DDH 問題を解くことができる． □

次に，以下の三つの補助定理を用いて，提案方式のブラック

ボックス追跡可能性を証明する．なお，補助定理1の証明は概略のみを示し，補助定理2と補助定理3の証明は省略する．(補助定理2と補助定理3も補助定理1の証明と同様の方法により証明できる.)

[補助定理1]（ヘッダの識別不可能性） k 人の結託者が無効化されていないとき，k 人の結託者が配信用ヘッダと追跡用ヘッダを識別することは G_q 上のDDH問題を解くことと計算量的に等価である．

[証明概略] k 人の結託者集合を$\mathcal{C}$とし，$\mathrm{Dis}_{\mathcal{C}}$ を C に属する結託者が配信用ヘッダと追跡用ヘッダを識別するための確率的多項式時間アルゴリズムとする．その他の表記は，定理1の証明において用いた表記を流用する．証明すべきことは，$\mathcal{X}\cap C=\emptyset$, $|\mathcal{C}|=k$ を満たす任意の$\mathcal{C}$に対して $\mathrm{Dis}_{\mathcal{C}} \Leftrightarrow$ DDH である．

DDH $\Rightarrow \mathrm{Dis}_{\mathcal{C}}$ は明らかである．$\mathrm{Dis}_{\mathcal{C}} \Rightarrow$ DDH について，$\mathrm{Dis}_{\mathcal{C}}$ を用いてDDHを構成できることを示し，これを証明する．

DDH(g_1,g_2,g_3,g_4): 全ユーザ集合 $\mathcal{U}$ ($\mathcal{U}\subseteqq\mathbb{Z}_q\backslash\{0\}$) を選択する．$\mathcal{U}$を要素数が $2k$ 以下であり，かつ互いに素な部分集合に分割し，完全二分木 T を定める．各ユーザは式(2)を用いてラベル付けされているとする．整数 $m\in\{1,\ldots,n-k\}$ を選択し，無効化ユーザ集合 $\mathcal{X}=\{u_1,\ldots,u_m\}$ を定める．$\mathcal{X}\cap\mathcal{C}=\emptyset$ を満たす k 人の結託者 $x_1,\ldots,x_k$ を選択する．

k 人の結託者の個人鍵 $d_{x_1},\ldots,d_{x_k}$ を与える．これらの値を用いて，$d_{x_i}=\{(x_i,v,A_v(x_i),B(x_i))|v\in\mathcal{N}_T,\ x_i\in\mathcal{U}_v\}$ かつ $a_0=\log_{g_1}g_2$ を満たす公開鍵 $e=(p,q,g_1,\{g_1^{a_i}\}_{i=0,\ldots,2k-1},\{g_1^{c_i}\}_{i=0,\ldots,2L-3},\{g_1^{\lambda_i}\}_{i=0,\ldots,2L-3})$ を計算する．ここで，e の計算は $A_v(x),B(x)$ を知ることなく行うことができる．

$d_{x_i}=\{(x_i,v,A'_v(x_i),B'(x_i))|v\in\mathcal{N}_T,\ x_i\in\mathcal{U}_v\}$ かつ $a'_0=\log_{g_3}g_4$ を満たす公開鍵 $e'=(p,q,g_3,\{g_3^{a'_i}\}_{i=0,\ldots,2k-1},\{g_3^{c'_i}\}_{i=0,\ldots,2L-3},\{g_3^{\lambda_i}\}_{i=0,\ldots,2L-3})$ を計算する．ここで，e' の計算は $A'_v(x),B'(x)$ を知ることなく行うことができる．また，$\log_{g_1}g_2=\log_{g_3}g_4$ のとき，$A_v(x)+\lambda_vB(x)=A'_v(x)+\lambda_vB'(x)$ である．

e と e' を用いて (g_1,g_2,g_3,g_4) がDiffie-Hellman組である場合配信用ヘッダとなり，そうでない場合追跡用ヘッダとなるヘッダ H を計算する．

$\mathrm{Dis}_{\mathcal{C}}$ に $d_{x_1},\ldots,d_{x_k},e,H$ を与える．$\mathrm{Dis}_{\mathcal{C}}$ が，H は配信用ヘッダであると判定した場合，"Diffie-Hellman組" と出力する．そうでない場合，"ランダム組" と出力する．$\mathrm{Dis}_{\mathcal{C}}$ は配信用ヘッダと追跡用ヘッダを識別可能であるから，DDHは与えられたDDH問題を解くことができる．また，$\mathcal{X}\cap\mathcal{C}=\emptyset$, $|\mathcal{C}|=k$ を満たす$\mathcal{C}$は任意に選択できるので，$\mathcal{X}\cap\mathcal{C}=\emptyset$, $|\mathcal{C}|=k$ を満たす任意の$\mathcal{C}$に対して $\mathrm{Dis}_{\mathcal{C}} \Leftrightarrow$ DDH である． □

[補助定理2]（追跡用ヘッダにおける秘匿性） 与えられた追跡用ヘッダにおいて k 人の結託者が無効化されているとき，k 人の結託者が追跡用ヘッダに対応するセッション鍵を計算することは G_q におけるDDH問題を解くことと少なくとも計算量的に等価である．

[補助定理3]（容疑者の識別不可能性） k 人の結託者以外のユーザが追跡用ヘッダを復号可能であるか否かを，k 人の結託者が識別することは G_q 上のDDH問題を解くことと計算量的に等価である．

[定理2]（ブラックボックス追跡可能性） G_q におけるDDH問題が困難であるという仮定の下で，提案方式は定義4において定められたブラックボックス追跡可能性を満たす．

[証明] $ctr_0=m$ は $\mathcal{X}=\emptyset$ としたときの検査結果値とみることができる．補助定理2より，圧倒的な確率で $ctr_n=0$ である．従って，$ctr_{j-1}-ctr_j\geq m/n$ を満たす $j\in\{1,\ldots,n\}$ が必ず存在する．ユーザ u_i が不正ユーザでない場合，補助定理3から，不正復号器は $\mathcal{X}=\{u_1,\ldots,u_{i-1}\}$ としたヘッダと $\mathcal{X}=\{u_1,\ldots,u_i\}$ としたヘッダを有意な確率で識別することは不可能であるため，$ctr_{j-1}-ctr_j\ll m/n$ となる．以上より，$ctr_{j-1}-ctr_j$ の値が最大となるとき，ユーザ u_j は圧倒的な確率で不正ユーザである．

次に，追跡ヘッダを検出すると動作しなくなる不正復号器を考える．結託者集合を$\mathcal{C}$とすると，補助定理1より，$\mathcal{X}\cap\mathcal{C}=\emptyset$ である限り，不正復号器は入力される追跡ヘッダを標準ヘッダと同様に復号する．従って，$\mathcal{X}=\{u_1,\ldots,u_i\}$ として検査したときに自己防衛機構が作動した場合，$\{u_1,\ldots,u_i\}\cap\mathcal{C}\neq\emptyset$ である．ユーザ u_i が不正ユーザでない場合，補助定理3から，不正復号器が $\mathcal{X}=\{u_1,\ldots,u_i\}$ である追跡ヘッダと $\mathcal{X}=\{u_1,\ldots,u_{i-1}\}$ である追跡ヘッダを識別することは不可能であるため，$\mathcal{X}=\{u_1,\ldots,u_{i-1}\}$ である追跡ヘッダを入力したときに自己防衛機構が作動しているはずである．従って，$\mathcal{X}=\{u_1,\ldots,u_j\}$ のときに自己防衛機構が作動した場合，u_j は圧倒的な確率で不正ユーザである． □

5.2 効率性

表1において，従来方式と提案方式を個人鍵サイズ，ヘッダサイズ，配信用鍵サイズ(注7)，追跡用鍵の公開可能性，結託耐性，及び復号に要する計算コストの観点から比較する．比較対象として，合成数位数の巡回群における双線形写像を巧みに用いて優れた性能を達成している文献[2]の方式も含める．

文献[9]と文献[2]の方式では，個人鍵サイズが一定である．それに対し，3.2項に示した単純な拡張方式と提案方式では，個人鍵サイズは木の深さに比例するが，例えば，$n=10^6, k=10^3$ のとき $\log_2 L<9$ であり，実用上問題とならない．

全ユーザ数を 10^6 とした場合の，各方式の最大結託人数に対するヘッダサイズを図3に示す．提案方式におけるヘッダサイズは概ね最大結託人数にのみ比例しているとみなすことができる．これに対し，単純な拡張方式では，木の深さが与える影響を無視できず，非効率的である．文献[9]と提案方式を比較すると，$k<n/16$ のとき常に提案方式の方が効率的である．文献[2]の方式と提案方式を比較すると，大雑把に言うと k の値を $1.5\sqrt{n}$ 程度より小さい値に設定するとき提案方式の方が効率的である．どちらの場合も実用上あり得るパラメータ選択の範囲であると思われる．

(注7)：p,q,g は公開鍵に含めない．文献[2]の方式についても同様に n ($=pq$), g,h,E は含めず，$|p|=|q|$ と想定する．

表 1 効率性比較

Table 1 Efficiency comparison ($\mathcal{P}$, $\mathcal{H}$, $\mathcal{S}$, $\mathcal{B}$: sets of possible personal keys, headers, session keys, and broadcaster's keys respectively, n: the total number of users, k: the maximum coalition size, L: the total number of leaves in a tree ($L = n/2k$))

	個人鍵サイズ ($\log\|\mathcal{P}\|/\log\|\mathcal{S}\|$)	ヘッダサイズ ($\log\|\mathcal{H}\|/\log\|\mathcal{S}\|$)	配信用鍵サイズ ($\log\|\mathcal{B}\|/\log\|\mathcal{S}\|$)	追跡用鍵は公開か？	許容する結託人数	復号に要する計算コスト
[9]	1	$4k+L+2$	$2k+L$	公開	制限有	$O(k)$
単純な拡張方式 (3.2 項)	$\log_2 L$	$(2k+1)(\log_2 L+1)$	$2(k\log_2 L+L-1)$	公開	制限有	$O(k)$
提案方式	$\log_2 L+1$	$2(2k+\log_2 L+2)$	$2(k+2L-2)$	公開	制限有	$O(k)$
[2]	1	$6\sqrt{n}$	$3\sqrt{n}$	秘密	無制限	$O(1)$

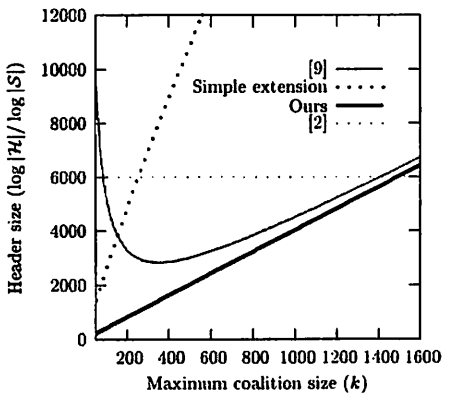


図 3 全ユーザ数が 10^6 の場合のヘッダサイズ

Fig. 3 Header size when the total number of users is 10^6

表 1 に示した全ての方式において，配信用鍵は公開できる．文献 [9] の方式と提案方式の配信用鍵サイズはともに $O(\sqrt{n})$ ($k = O(\sqrt{n})$) であるが，k の値を n の値に近づけるにつれて，文献 [2] の方式がより効率的になる．

提案方式では，追跡用鍵を公開することができるため，誰でも追跡者になることができる．これに対し，文献 [2] の方式では，追跡用鍵を秘密にしなければならない．追跡用鍵が秘密である場合，追跡処理を分散させるために追跡を行うエンティティを複数にすると，追跡用鍵が漏洩する可能性が増してしまうが，追跡用鍵が公開である場合，上記問題は生じない．

提案方式では，許容できる結託人数に上限があり，また，復号に要する冪剰余演算回数が最大結託人数に比例してしまう．これに対し，文献 [2] の方式では，結託人数に上限はなく，また，復号に要するペアリング回数は定数である．上記短所はあるが，ヘッダサイズを小さく抑えることが最優先である場合，提案方式は一つの選択肢になり得る．

6 まとめ

より巧妙に作成された不正復号器に対してもブラックボックス追跡可能な従来方式 [9] のヘッダサイズ削減を可能とする，階層的な鍵割り当て方法を提案した．この階層的な鍵割り当て方法を従来方式に適用した結果得られる方式は，従来方式と同等のブラックボックス追跡性能を有し，かつ個人鍵サイズが大きく増加することなしに，最大結託人数にのみ比例すると実用上みなせるヘッダサイズを達成する．

文献

[1] D. Boneh and M. Franklin. An efficient public key traitor tracing scheme. In *Proc. CRYPTO'99*, LNCS 1666, pages 338–353. Springer-Verlag, 1999.

[2] D. Boneh, A. Sahai, and B. Waters. Fully collusion resistant traitor tracing with short ciphertexts and private keys. In *Proc. EUROCRYPT 2006*, LNCS 4004, pages 573–592. Springer-Verlag, 2006.

[3] H. Chabanne, D. Phan, and D. Pointcheval. Public traceability in traitor tracing schemes. In *Proc. EUROCRYPT 2005*, LNCS 3494, pages 542–558. Springer-Verlag, 2005.

[4] B. Chor, A. Fiat, and M. Naor. Tracing traitors. In *Proc. CRYPTO'94*, LNCS 839, pages 257–270. Springer-Verlag, 1994.

[5] A. Kiayias and M. Yung. On crafty pirates and foxy tracers. In *Security and Privacy in Digital Rights Management: Revised Papers from the ACM CCS-8 Workshop DRM 2001*, LNCS 2320, pages 22–39. Springer-Verlag, 2002.

[6] A. Kiayias and M. Yung. Traitor tracing with constant transmission rate. In *Proc. EUROCRYPT 2002*, LNCS 2332, pages 450–465. Springer-Verlag, 2002.

[7] A. Kiayias and M. Yung. Breaking and repairing asymmetric public-key traitor tracing. In *Revised Papers from the ACM CCS-9 Workshop DRM 2002*, LNCS 2696, pages 32–50. Springer-Verlag, 2003.

[8] K. Kurosawa and Y. Desmedt. Optimum traitor tracing and asymmetric schemes. In *Proc. EUROCRYPT'98*, LNCS 1403, pages 145–157. Springer-Verlag, 1998.

[9] T. Matsushita and H. Imai. A public-key black-box traitor tracing scheme with sublinear ciphertext size against self-defensive pirates. In *Proc. ASIACRYPT 2004*, LNCS 3329, pages 260–275. Springer-Verlag. 2004.

[10] T. Matsushita and H. Imai. A flexible-revocation scheme for efficient public-key black-box traitor tracing. *IEICE Trans. Fundamentals*, E88-A(4):1055–1062, 2005.

[11] D. Naor, M. Naor, and J. Lotspiech. Revocation and tracing schemes for stateless receivers. In *Proc. CRYPTO 2001*, LNCS 2139, pages 41–62. Springer-Verlag, 2001.

[12] B. Pfitzmann. Trials of traced traitors. In *Proc. Information Hiding*, LNCS 1174, pages 49–64. Springer-Verlag, 1996.

[13] B. Pfitzmann and M. Waidner. Asymmetric fingerprinting for larger collusions. In *Proc. ACM Conference on Computer and Communication Security (ACM-CCS'97)*, pages 151–160, 1997.

[14] Y. Watanabe, G. Hanaoka, and H. Imai. Efficient asymmetric public-key traitor tracing without trusted agents. In *Proc. CT-RSA 2001*, LNCS 2020, pages 392–407. Springer-Verlag, 2001.