SCIS 2004 The 2004 Symposium on Cryptography and Information Security
Sendai, Japan, Jan.27-30, 2004
The Institute of Electronics, Information and Communication Engineers

RSA 暗号を利用した効率的な木構造鍵管理方式
Efficient Tree Based Key management based on RSA function

尾形 わかは * 桧座 哲夫 * ズォング ベト カン *
Wakaha OGATA Tetsuo HIZA Duong Viet Quan

あらまし コンテンツ保護の目的で，正当なユーザのみが復号化できるようにコンテンツを暗号化する技術が注目されている．権利を失ったユーザの鍵を効率的に無効化するために，木構造を用いた鍵管理方式が提案されているが，従来の方式では，ユーザが多くの鍵を保持する必要があったり，大きな公開情報を必要とし たり，鍵生成が効率的でないなどの欠点があった．本研究では，効率的に鍵を生成でき，ユーザの持つべき鍵長がユーザ数によらない RSA 暗号の安全性に基づく効率的な方式を提案し，安全性について議論する．

キーワード 木構造鍵管理方式, RSA 暗号, コンテンツ保護, ブロードキャスト暗号

1 はじめに

コンテンツ保護の観点から，ブロードキャスト暗号方式が注目を浴びている．ブロードキャスト暗号方式では，復号する権利のあるユーザはあらかじめ復号化の鍵をもらっておき，その鍵を用いてブロードキャストされた暗号文を復号化してコンテンツを見ることができる．このとき，全ユーザに共通の鍵を渡すと，権利を失ったユーザが現れた時点で，全てのユーザが保持する鍵を更新しなければならない．一方，全ユーザに異なる鍵を渡し，全ての鍵での暗号文をブロードキャストすれば，個々のユーザを独立に除外することができるが，ブロードキャストされる暗号文が長くなってしまうという欠点がある．

これらの欠点を解決するアプローチのひとつとして，木構造に基づいて鍵割り当てを行う方式が Wong らによって提案された [1]．これは Logical Key Hierarchy（LKH）法と呼ばれる．Noar らは LHK 法を元に Complete Subset（CS）方式を，また同時に別の鍵構成法として Subset Difference（SD）方式も提案している [2]．SD 方式は，ユーザの持つ鍵長が CS 方式に比べ長くなってしまうのに対し，暗号文の長さを短くすることができる．しかし，これらは共に，ユーザの鍵長が全ユーザ数 N に依存して長くなるという欠点を持っている．一方，ユーザの持つ鍵長を一定にすることのできる方式がいくつか提案されている．浅野は，ユーザが持つべき複数の鍵を，1 つのマスターキーと，公開されている情報から復元できるように工夫することによって，鍵長を固定できる方式を提案した [3]．この方式は，鍵生成のために N に依存した量の公開情報を必要とするという欠点を持つ．一方，野島と楫は，2 つの落とし戸付き一方向性関数を利用して，LKH 法においてユーザが持つべき複数の鍵を，1 つの鍵から生成する方式を提案し，具体的な構成例として RSA 関数を用いたものを示している [4]．また，菊地は一方向性関数として Rabin 暗号を用いた構成方法を示している．これら構成方法に対しては，厳密な安全性の議論がなされていない．

本研究では，木構造を用いたブロードキャスト暗号方式について，RSA 関数を用いた鍵構成方法を提案する．提案方式では公開情報やユーザの鍵長は N に依存せず一定である．また，特別な不正に対する安全性が，RSA 暗号の解読と等価であることを証明する．本方式は，野島，楫による構成方法の特別な場合ではない．

2 従来の方式

2.1 独立鍵を用いた方法

以降，ユーザ数 N を 2 のべきとし，$N = 2^h$ とする．2 分木を用いたブロードキャスト暗号方式では，N 個の葉を持つ，深さ h の 2 分木を利用して，ユーザへの鍵割り当て，ブロードキャストする暗号文の構成を行う．

鍵生成: 2 分木の各ノード n_j $(j = 1, \ldots, 2N-1)$ には独立な復号化の鍵 K_j が割り当てられる．各ユーザ U_i $(i = 1, \ldots, N)$ は 2 分木の葉に割り当てられ，木のルートから葉までの経路上の全ての鍵を事前に受け取る．

* 東京工業大学, 〒 152-8552 目黒区大岡山 2-12-1, Tokyo Institute of Technology, 2-12-1 O-okayama, Meguro-ku, Tokyo, Japan, wakaha@craft.titech.ac.jp

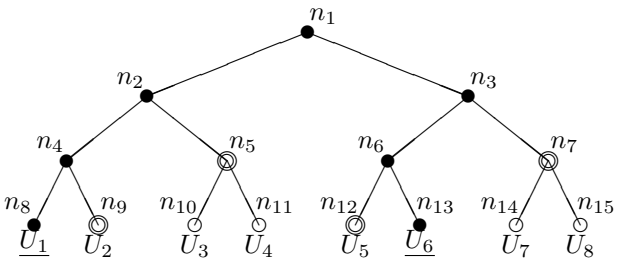

図 1: $N = 8$ の場合の 2 分木と U_1,U_6 の無効化

暗号化と復号化: ブロードキャスト暗号では，暗号化はハイブリッド型で生成される．つまり，コンテンツはランダムに選ばれたセッションキー SK で暗号化され，SK がユーザの持つ鍵で復号化できるように暗号化される．以降では，SK の暗号文作成に手順のみを扱う．

2 分木を用いた鍵の割り当てがされている場合，除外されるユーザがいない場合，SK はルートの鍵で暗号化される．全てのユーザはルートの鍵を知っているため，SK を得る．除外されたユーザがいる場合，そのユーザの知る全ての鍵に対応するノードを 2 分木から取り除き，複数の部分木を得る．得られた部分木のルートノードの鍵で SK を暗号化する．図 1 に，ユーザ数が $N = 8$，ユーザ U_1 と U_6 を無効化した場合の様子を示す．この場合，SK が K_5, K_7, K_9, K_{12} で暗号化され，4 つの暗号文がブロードキャストされる．

効率: 各ユーザの保持すべき鍵長は，$h+1 = \log N+1$ に比例する．また，r 人が除外されたときの暗号文の長さは $O(r \log(N/r))$ であることが知られている [2]．

2.2 浅野の方式 [3]

Asiacrypt '02 において，浅野は 2 つの方式を示しているが，ここでは，ユーザの持つべき鍵長が N によらず一定である方式について簡単に述べる．

この方式では，葉以外の各ノードに，そのノードの全ての子ノードからなる集合の（空集合を除く）全ての真部分集合に対する鍵が割り当てられる．たとえば，図 1 のような 2 分木を用いた場合，ノード n_2 には，子ノードの部分集合 $\{n_4\}$ に対応する鍵と，$\{n_5\}$ に対応する鍵の 2 つが割り当てられる．また，3 分木を用いた場合には，各ノードには 6 つの鍵が割り当てられる．

各ユーザは，ユーザに対応する葉からルートノードへのパス上に存在する葉以外の各ノードに対し，パス上に存在するノードを含む部分集合に対応する鍵を，全て知らなくてはならない．ここで，複数の鍵を 1 つの鍵から計算可能とするために，マスターキー技術が用いられている．マスターキーから複数の鍵を計算するためには，公開情報を用いるため，N に比例する公開情報が必要となる．

暗号化・復号化手順については省略する．

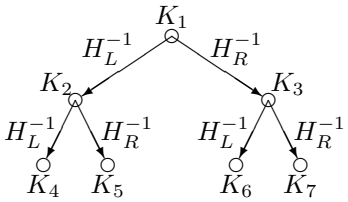

図 2: 2 つの一方向性関数を用いた構成

2.3 2 つの一方向性関数を用いた構成方法 [4]

野島と楫の構成方法では，あるノード n_j に対応する鍵 K_j が，その子ノード n_{2j}, n_{2j+1} に対応する鍵から，それぞれ一方向性関数を用いて計算できるようにし，ユーザには，葉に対応する鍵のみをわたす．一方向性関数として，落とし戸付きのものを用いることで，センターは，ルートノードの鍵 K_1 から，全ての鍵を計算することができる．

手順としては，まず 2 つの一方向性関数 H_R と H_L を用意する．K_1 をランダムに選び，$j = 2, \ldots, 2N-1$ に対して

$$K_j = \begin{cases} H_L^{-1}(K_{\lfloor j/2 \rfloor}) & j \text{ が偶数のとき} \\ H_R^{-1}(K_{\lfloor j/2 \rfloor}) & j \text{ が奇数のとき} \end{cases}$$

を順に計算する（図 2 を参照）．ユーザ U_i には，K_{i+N-1} を渡す．

暗号化・復号化手順は，独立鍵を用いた方式と同様である．

一方向性関数の例として，$H_L(x) = x^{e_L} + c_L \bmod n, H_R(x) = x^{e_R} + c_R \bmod n$ が与えられているが，2 つの関数の選び方の条件は厳密には与えられていない．この関数構成例では，ある 2 つのノードの鍵が一致した場合には，その子孫の鍵構成が全て同一になるため，ルートノードの鍵を再選択するという欠点があった．

2.4 Rabin tree 方式 [5]

菊地は，一方向性関数として，2 つの異なる法をもつ Rabin 暗号，

$$\begin{aligned} H_R(x) &= x^2 \bmod n_R, \\ H_L(x) &= x^2 \bmod n_L \end{aligned}$$

を用いることを提案している．この場合，ユーザが自分の鍵から必要な鍵を生成するのに必要な計算量は RSA を用いた場合より小さくなるが，N が大きくなった場合，全ての鍵を構成することが可能なルートの鍵の選択が困難となる．

3 提案方式

3.1 概要

提案方式では，一方向性関数にノード番号を含める．もともと，一方向性関数を用いた木構造では，左右方向

のハッシュ関数をそれぞれ区別するために，ユーザが自分のユーザ番号を必ず記憶しなければならないので，新たな記憶量にはつながらない．

[4] で議論された，複数ノードの鍵が一致した場合に，該当ノードの子孫の鍵構成が全て同一になってしまうという問題も，ノード番号を含めることにより回避できる．

コンテンツの暗号化・復号化手順は，独立鍵を用いた方式 (2.1 節) と同様である．

3.2 センター鍵構成法

RSA 暗号のパラメータ組 n, e, d を用意し，n, e を公開する．そして，ルート鍵 K_1 をランダムに選び，$j = 2, \ldots, 2N-1$ に対して

$$K_j = (K_{\lfloor j/2 \rfloor} + j)^d \bmod n$$

を順に計算し，葉の鍵 K_{i+N-1} をそれぞれ対応する各ユーザ U_i に渡す．センターはルート鍵 K_1 と d のみを秘密情報として記憶しておくことで，木構造の任意のノードの鍵を必要に応じて再計算できる．

3.3 コンテンツ復号鍵 (SK) 取得法

ユーザ U_i は，センターから割り当てられた秘密鍵 K_{i+N+1} と公開パラメータ組 n, e から，

$$K_{\lfloor j/2 \rfloor} = ({K_j}^e - j) \bmod n$$

を，SK の復号鍵が得られるまで 0〜h 回適用する．ユーザが正当であれば，葉からルートまでの経路の鍵セット $K_{i+N+1}, K_{\lfloor j/2 \rfloor}, \ldots, K_1$ のなかに SK を復号する鍵が含まれる．

3.4 性能比較

表 1 に従来方式と提案方式の性能比較をあげる．野島らの方式については，RSA を用いた例である．なお，独立鍵を用いた方式については，鍵ビット長が短くてすむため，一概に比較はできない．

センターが保持する秘密情報量: 落とし戸付き一方向性関数を用いない木構造方式，またはルート鍵から各ノードの鍵が一意に定まらない RabinTree 方式では，全ユーザ (N 個) の鍵情報を記憶する必要がある．2 つの一方向性関数を用いる方式では，1 個のルート鍵 K_1 と 2 つの一方向性関数の落とし戸情報を記憶することで，全ノードの鍵を計算可能である．提案方式では，一方向性関数は 1 つしか用いないので，情報量が削減される．

ユーザが保持する秘密情報量: 独立鍵方式では，センターから割り当てられた複数の鍵を静的に記憶する．一方向性関数を用いる方式では，センターから割り当てられた 1 つの葉の鍵からルートまでの経路の，復号に必要な鍵を動的に計算する．

センターが公開しておく情報量: 一方向性関数を用いる方式では，ハッシュ関数のパラメータを公開する．公開パラメータはその都度ブロードキャストするか，あらかじめ記憶する必要がある．前者では通信量の増大，後者では (秘密情報ではないのでコストは低いが) 記憶量の増大となる．同じ安全性を保つのであれば，小さいほうが望ましい．浅野の方式では，$O(N)$ の素数を公開する必要がある．野島らの構成法では n, C_L, C_R, e_L, e_R が公開されるのに対して，提案方式では 1/2 の情報量にあたる n, e を公開すればよい．

センターが全ノードの鍵を構成するのに必要な計算量: RabinTree 方式では，木を構成可能なルート鍵を探すために，指数時間の計算量が必要になるとされている．野島らの方式や提案方式では，任意のルート鍵から木を構成できるので，計算量はノード数に単純に比例する．

4 安全性についての考察

$U_{i_1}, U_{i_2}, \ldots, U_{i_k}$ が結託をして，他のユーザの鍵を求めようとする不正を考える．$A = \{U_{i_1}, U_{i_2}, \ldots, U_{i_k}\}$ とすると，A は，$\{U_{i_1}, U_{i_2}, \ldots, U_{i_k}\}$ に対応するノードからルートノードへの全てのパス上にある鍵 K_i を知ることができる．A の知ることのできる鍵の集合を $\mathcal{K}_A$ と書くことにする．

ここで，ユーザ集合 A が不正なユーザとして除外される場合のブロードキャスト情報を計算するのに使用される鍵を $\bar{\mathcal{K}}_A^-$ と書くことにする．たとえば，$A = \{U_1, U_6\}$ のとき, 図 1 より $\bar{\mathcal{K}}_A^- = \{K_5, K_7, K_9, K_{12}\}$ である．

A の目的は，$\mathcal{K}_A$ から，別の鍵 $K_j \notin \mathcal{K}_A$ を求めることであるが，K_j が求まれば，$\bar{\mathcal{K}}_A^-$ に含まれる $K_{j'}$ が容易に計算できるため，以降は，$K_j \in \bar{\mathcal{K}}_A^-$ を求めることを A の目的と考える．

さて, 任意の $K_j \in \bar{\mathcal{K}}_A^-$ に対し, $K_{j'} \in \mathcal{K}_A$ かつ $\lfloor j/2 \rfloor = \lfloor j'/2 \rfloor$ であるような j' が存在する．ここでは，RSA 問題が難しいと仮定すると，$K_{j'}$ から K_j を得ることが困難であることを証明する．

定理 1 鍵 K_{2i} から鍵 K_{2i+1} を求めることは，*RSA* 問題と等価である．

(Proof) RSA 問題が解ければ，K_{2i} から $K_{2i+1} = ({K_{2i}}^e + 1)^d \bmod n$ が求められることは明らか．

次に，鍵 K_{2i} から鍵 K_{2i+1} を求めるアルゴリズム M^+ が存在するならば RSA 問題が解けることを示す．すなわち，鍵 K_{2i} を入力として K_{2i+1} を出力するアルゴリズム $M^+(K_{2i}) = ({K_{2i}}^e + 1)^d \bmod n$ を用いて，任意の α に対して $R(\alpha) = \alpha^d$ を計算するアルゴリズムが構成可能であることを，数学的帰納法を用いて示す．

まず，$M^+(1) = (1^e + 1)^d = 2^d \pmod n$.

表 1: 性能比較

	センター 秘密情報量	ユーザ 秘密情報量	公開関数 情報量	センター 鍵構成計算量
独立鍵を用いた方式 [1]	$2N-1^*$	$\log N^*$	-	-
浅野の方式 [3]	2	1	$O(N)$	$O(N)$
野島らの方式 (RSA 適用例)[4]	3	1	5	$O(N)$
RabinTree 方式 [5]	N	1	2	$O(2^N)$
提案方式	2	1	2	$O(N)$

(*) 独立鍵を用いた方式については，鍵ビット長はブロック暗号の鍵長 (128bit)．その他については，鍵長は 1024bit 程度.

(1) α が 1 bit のとき

$$R(0) = 0 \bmod n,$$
$$R(1) = 1 \bmod n,$$

よって $R(\alpha)$ を計算可能.

(2) α が l bit $(l \geq 1)$ のとき，$R(\alpha)$ を計算可能と仮定し，$(l+1)$ bit の任意の β について考える.

(i) β の最下位 bit が 0 のとき,

$$R(\beta) = R(2\lfloor\frac{\beta}{2}\rfloor) \pmod n.$$

(ii) β の最下位 bit が 1 のとき,

$$\begin{aligned} R(\beta) &= R(2\lfloor\frac{\beta}{2}\rfloor + 1) \\ &= ((2\lfloor\frac{\beta}{2}\rfloor)^{de} + 1)^d \\ &= M^+((2\lfloor\frac{\beta}{2}\rfloor)^d) \\ &= M^+(R(2\lfloor\frac{\beta}{2}\rfloor)) \pmod n. \end{aligned}$$

ここで,

$$\begin{aligned} R(2\lfloor\frac{\beta}{2}\rfloor) &= (2\lfloor\frac{\beta}{2}\rfloor)^d \\ &= 2^d \times (\lfloor\frac{\beta}{2}\rfloor)^d \\ &= M^+(1) \times R(\lfloor\frac{\beta}{2}\rfloor) \pmod n \end{aligned}$$

は，$\lfloor\frac{\beta}{2}\rfloor$ の bit 数が l であるので計算可能.

よって，$(l+1)$ bit の任意の β についても $R(\beta)$ が計算可能.

(1), (2) より，RSA 問題を解くアルゴリズムが構成可能になる. ∎

定理 2 鍵 K_{2i+1} から鍵 K_{2i} を求めることは，*RSA* 問題と等価である.

(Proof) RSA 問題が解ければ，K_{2i+1} から $K_{2i} = ({K_{2i+1}}^e - 1)^d \bmod n$ が求められることは明らか.

次に，鍵 K_{2i+1} から鍵 K_{2i} を求めるアルゴリズム M^- が存在するならば RSA 問題が解けることを示す．すなわち，鍵 K_{2i+1} を入力として K_{2i} を出力するアルゴリズム $M^-(K_{2i+1}) = ({K_{2i+1}}^e - 1)^d \bmod n$ を用いて，任意の α に対して $R(\alpha) = \alpha^d$ を計算するアルゴリズムが構成可能であることを，数学的帰納法を用いて示す.

まず，d, e は常に奇数であるから,

$$M^-(-1) = ((-1)^e - 1)^d = (-2)^d \pmod n.$$

(1) α が 1 bit のとき

$$R(0) = 0 \bmod n,$$
$$R(1) = 1 \bmod n,$$

よって $R(\alpha)$ を計算可能.

(2) α が l bit $(l \geq 1)$ のとき，$R(\alpha)$ を計算可能と仮定し，$(l+1)$ bit の任意の β について考える.

(i) β の最下位 bit が 0 のとき,

$$R(\beta) = R(2\lfloor\frac{\beta}{2}\rfloor) \pmod n.$$

(ii) β の最下位 bit が 1 のとき,

$$\begin{aligned} R(\beta) &= R(2\lfloor\frac{\beta}{2}\rfloor + 1) \\ &= ((2\lfloor\frac{\beta}{2}\rfloor)^{de} + 1)^d \\ &= -((-2\lfloor\frac{\beta}{2}\rfloor)^{de} - 1)^d \\ &= -M^-((-2\lfloor\frac{\beta}{2}\rfloor)^d) \\ &= -M^-(-R(2\lfloor\frac{\beta}{2}\rfloor)) \pmod n. \end{aligned}$$

ここで,

$$\begin{aligned} R(2\lfloor\frac{\beta}{2}\rfloor) &= (2\lfloor\frac{\beta}{2}\rfloor)^d \\ &= -(-2)^d \times (\lfloor\frac{\beta}{2}\rfloor)^d \\ &= -M^-(-1) \times R(\lfloor\frac{\beta}{2}\rfloor) \pmod n \end{aligned}$$

は，$\lfloor \frac{\beta}{2} \rfloor$ の bit 数が l であるので計算可能.

よって，$(l+1)$ bit の任意の β についても $R(\beta)$ が計算可能.

(1), (2) より，RSA 問題を解くアルゴリズムが構成可能になる. ∎

注意: A は $K_{j'}$ 以外にも多くの鍵情報を知っているため，定理 1，2 によって，$\mathcal{K}_A$ から任意の $K_j \in \bar{\mathcal{K}}_A^-$ を求めるのが困難であることを示したことにはならない. たとえば，K_2 から K_3 が求まらないことは証明できるが，K_2, K_4, K_5 から K_3 が求まらないとはいえない. しかし，これも難しい問題であると予想される.

5 まとめ

RSA を利用した落とし戸付き木構造鍵管理方式として，従来よりも効率が良い構成法を提案し，特別な不正に対する安全性を示した. 今後の課題としては，一般的なアタックに対する安全性の証明を試みること，効率を維持しつつ，安全性の証明が可能になるようなハッシュ関数構成を考案することなどがあげられる.

参考文献

[1] C. K. Wong, M. Gouda and S. S. Lam, “Secure Group Communications Using Key Graphs”, Proceedings of ACM SIGCOMM’98, 1998.

[2] D. Naor, M. Naor and J. Lospiech, “Revocation and Tracing Schemes for Stateless Receivers”, Crypto’01, LNCS 2139, pp.41-62, Springer, 2001.

[3] 浅野, “A Revocation Scheme with Minimal Storage at Receivers”, Asiacrypt’02, pp.433-450, 2002.

[4] 野島, 楫, “落とし戸付き一方向性関数を利用した木構造鍵管理方式”, SCIS’03，pp.131-136, 2003.

[5] 菊地, “Rabin Tree とブロードキャスト暗号への応用”, ISEC’03，pp.9-12, 2003.