論文 / 著書情報
Article / Book Information

論題(和文)	プライバシ保護を考慮したRFID方式の大規模システムへの適用
Title(English)	Application of privacy protection RFID methods to large-scale systems
著者(和文)	阿部正己, 尾形わかは
Authors(English)	Masaki ABE, Wakaha OGATA
出典(和文)	電子情報通信学会論文誌「基礎・境界：A」, Vol. J91-A, No. 3, pp. 399-410
Citation(English)	, Vol. J91-A, No. 3, pp. 399-410
発行日 / Pub. date	2008, 3
URL	http://search.ieice.org/
権利情報 / Copyright	本著作物の著作権は電子情報通信学会に帰属します。 Copyright (c) 2008 Institute of Electronics, Information and Communication Engineers.

論　文

プライバシー保護を考慮したRFID方式の大規模システムへの適用

阿部　正已[†*]　　尾形わかは[†a)]

Application of Privacy Protection RFID Methods to Large-Scale Systems

Masaki ABE[†*] and Wakaha OGATA[†a)]

あらまし　将来的に，更なるRFIDシステムの普及への障害となり得るプライバシー問題を解決する方法の一つとして，タグを内部更新し，出力を毎回可変にするOSK方式が大久保，鈴木，木下によって提案された．しかし，この方式はデータベース側の負荷が高く，スケーラビリティに課題があった．そこで，OSK方式よりスケーラビリティが高いOSK/AO方式がAvoine，Oechslinによって提案された．本論文では，オープンな環境での利用を前提として，OSK/AO方式の効率の再評価を行い，大規模システムでの利用を想定した場合に問題が生じることを指摘する．また，その問題点の解決策を二つ提案する．

キーワード　RFIDシステム，ハッシュチェーン方式，タイムメモリ・トレードオフ，OSK/AO方式

1. まえがき

近年急速に，RFID（Radio Frequency Identification）システムへの関心が高まり，物流におけるサプライチェーンマネジメントや，商品管理，履歴管理などでの活用が検討され，一部では既に実用化されている．RFIDシステムは主に，非接触のICであるRFタグ（以降，タグと呼ぶ）と，タグへID要求を出しタグからの返答を受けるリーダからなる．現状では，すべてのタグは自システムのリーダ以外からアクセスを受けないというクローズな環境を前提としたソリューションで用いられている．しかし今後は，ユビキタス社会の実現に向けて，消費者が購入する商品にタグを付け，購入後にも様々なサービスを提供する等，タグが自システムのリーダ以外からもアクセスを受けることもあるようなオープンな環境におけるソリューションへの適用が期待されている．

現在用いられているタグをオープンな環境におけるRFIDシステムで用いた場合，タグは常に一定のIDを発信しつづけるため，第三者が所有者に無断でタグのIDを読み取ることにより，以下に示すプライバシー問題が生じる．

1. IDから所持品を類推．
2. 固定のIDを追跡することで個人の行動追跡．

更に，コストの観点からタグに耐タンパ性を保証することは難しいことから，

3. タグの解析による行動追跡，本人特定

といった問題もある．これらは，将来的に，更なるRFIDシステムの普及への障害となり得る．

これらに対し，プライバシー保護を考慮したRFID方式が提案されている．

Hash lock方式[8]では，秘匿化（暗号化など）したIDをタグに格納することによって所持品の推測の問題を解決しているが，IDの追跡は可能である．Randomized hash lock方式[9]，YA-TRAP方式[10]やK-steps方式[11]では，タグからの出力を可変にすることにより，所持品の推測だけでなく，ID追跡の問題も解決している．しかし，出力はタグ内に保存される固定のIDや鍵などから計算されるため，タグを解析することにより，過去の行動追跡が可能となってしまう．

タグ解析による追跡をも防ぐ方式としては，大久保，鈴木，木下により提案された方式[1]がある．以降，この方式をOSK方式と呼ぶ(注1)．OSK方式はタグの物

† 東京工業大学，東京都
Tokyo Institute of Technology, 2–12–1 O-okayama, Meguro-ku, Tokyo, 152–8552 Japan
* 現在，(株) NTT データ
a) E-mail: wakaha@mot.titech.ac.jp

(注1)：Extended Hash-chain方式とも呼ばれる．

理的な解析によってメモリの中身を読み取ったとしても，タグの過去の送信履歴を推測できないというフォワードセキュア性も満たしているため，安全性の高いRFIDシステムといえる．

OSK方式はタグ数I，タグの最大更新回数Kに対して，ID認証ごとに，タグの認証をするデータベースでは$O(IK)$の計算を行う，若しくは，$O(IK)$のメモリを用意して事前に$O(IK)$の計算を必要とする．そのため，タグ数が大きいシステム等においては，データベースの負荷が高く，スケーラビリティに課題があった．

そこで，暗号解読などで使われるTime-memory Trade-off方式[3], [4]をこのOSK方式に適用させることで，メモリ量とデータベースでの計算量を調整できるスケーラビリティの高いOSK/AO方式がAvoine, Oechslinによって提案された[6], [7]．

本論文では，オープンな環境での利用を前提として，OSK方式及びOSK/AO方式の実行可能性を検討する．まず，OSK/AO方式の事前計算とID認証の詳細なアルゴリズムを用いて，計算量を見積もることにより，OSK/AO方式を大規模システムでの利用を想定した場合に問題が生じることを指摘する．また，その問題点の解決策を二つ提案する．一つは，OSK/AO方式を用いてID認証をする際に，複数回アクセスすることにより効率改善を行うものである．もう一つは，大久保，鈴木，木下によって提案されたOSK方式の変形[2]にTime-memory Trade-off方式を適用することで，効率を改善するものであり，複数回アクセスと組み合わせることで効率は大幅に改善できる．

2. RFIDシステムとOSK方式

2.1 RFIDシステム

RFIDシステムは，非接触なICであるRFタグ$Tag_i(0 \leq i < I)$，タグの情報を読み書きする装置リーダ/ライタ，タグの認証をするデータベースB，によって行われるプロトコルであり，Setup phaseとIdentification phaseの二つのphaseからなっている．Setup phaseはBとタグの初期化段階であり，Bは各タグTag_iにそれぞれ固有のIDであるID_iを割り当て，ライタを通じて書き込む．また，BもID_iを保存する．Identification phaseでは，リーダがタグにID取得要求を出し，タグから受け取ったIDをBに送り，BがID認証を行う．

2.2 OSK方式

OSK方式のプロトコルの詳細を示す前に概略を述べる．Setup phaseでは，Bは各タグに初期秘密情報をライタを通じて書き込む．また，Bも初期秘密情報を保存する．Identification phaseでは，リーダがタグと通信し，秘匿IDと呼ばれる情報を得る．タグは同時に保存している秘密情報の更新をする．リーダは秘匿をBへ送り，それを元にBがID認証を行う．

以下で，文中で用いる記号の定義を行う．

I	:	Bの管理するタグ数
K	:	タグの最大更新回数
s_i^0	:	Tag_iとBが最初に保持する初期秘密情報
s_i^k	:	k回更新された後にTag_iが保持している秘密情報$(0 \leq k < K)$
r_i^k	:	k回更新を行った後のTag_iの秘匿ID
$\|s\|$	:	s_i^kとr_i^kのビット長
G, H	:	$\|s\|$ビット出力のハッシュ関数

以下にOSK方式のプロトコルを示す．

[Setup phase]

1. 各$i(0 \leq i < I)$に対し，BはID_iとランダムなs_i^0を生成し，s_i^0をライタを通じてTag_iに書き込む．
2. Bは，ID_iとs_i^0を関連づけて保存する．

[Identification phase]（図1参照）

1. リーダは，$Tag_{i'}$にID取得要求を出す．
2. $Tag_{i'}$は，現在保持している秘密情報$s_{i'}^{k'}(0 \leq k' < K)$から秘匿ID $r_{i'}^{k'} = G(s_{i'}^{k'})$を生成する．
3. $Tag_{i'}$は$r_{i'}^{k'}$を，リーダを通じてBに送信する．
4. $Tag_{i'}$は，秘密情報を$s_{i'}^{k'+1} = H(s_{i'}^{k'})$によって更新する．
5. Bは，すべてのi, kに対して，s_i^0を用いて$G(H^k(s_i^0))$を算出し，受信した$r_{i'}^{k'}$と比較する．
6. Bは，$r_{i'}^{k'} = G(H^k(s_i^0))$となる$i$が存在した場合，$ID_i$をリーダに送信する．存在しなければ認

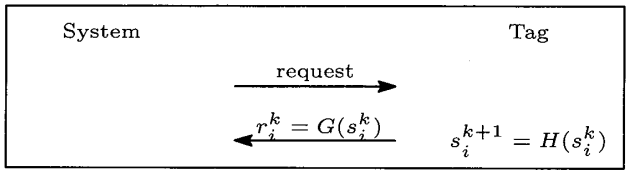


図1 OSK方式のIdentification phase
Fig. 1 Identification phase of OSK scheme.

証失敗とする．

OSK方式では，Identification phaseですべてのi,kに対してs_i^0を計算するため，ID認証に時間がかかる．大きいメモリが用意できる場合，事前にlookup tableを作成することで，Identification phaseを高速に実行できる．今後は，ID認証に完全探索を用いる方式をOSK/完全探索，lookup tableを用いる方式をOSK/lookup tableと書くことにする．OSK/lookup tableの詳細を以下に示す（OSK/完全探索からの追加分，変更点のみ示す）．

[Setup phase]

2. Bは，すべての秘匿ID $r_i^k = G(H^k(s_i^0))(0 \leq i < I, 0 \leq k < K)$を$s_i^0$から求め，それらと$ID_i$を関連づけるため$(r_i^k, i)$のようなペアを生成する．
3. 上記のペアをr_i^kに関してソートしてtableとして保存する．

[Identification phase]

5. Bは，受信した$r_{i'}^{k'}$と一致するものが，事前に生成したtable内に存在するか二分探索を用いて比較する．
6. Bは，table内に$r_{i'}^{k'} = r_i^k$となる(r_i^k, i)がある場合，ID_iをリーダに送信する．一致するものがtable内になければ認証失敗とする．

OSK/完全探索及びOSK/Lookup tableにおける事前計算量，ID認証の最悪計算量及び必要メモリは，表1のとおりである．ただし，計算量はBにおける計算量とし，特に注釈のない場合はハッシュ関数の適応回数で評価する．また，必要メモリはBにおいて必要なメモリ量で，ID_iを保存するためのメモリ量は除く（以降，同様）．

2.3 OSK2方式

OSK方式には，効率改善のためのいくつかの変形方式が提案されている．ここでは，大久保，鈴木，木下によって提案されたOSK2方式[2]について説明する．

OSK2方式は，OSK方式のIdenitification phaseにおいて，タグからリーダへ秘匿IDとともに現在のタグの更新回数を送るようにし，各タグ固有の秘密情報に加えて，内部更新用の各タグ共通のコモンシードを導入したものである．

以下で，OSK2方式で用いる記号の定義を行う．

s_i ： Tag_iとBが保持する秘密情報
cs_0： すべてのタグとBが最初に保持するコモンシード
cs_k： k回更新されたタグが保持しているコモンシード$(0 \leq k < K)$
r_i^k ： k回更新した後のTag_iの秘匿ID（$|s_i| = |cs_k| = |r_i^k|$）

以下に完全探索を用いるOSK2方式であるOSK2/完全探索のプロトコルを示す．

[Setup phase]

1. Bはランダムなcs_0を生成する．
2. 各$i(0 \leq i < I)$に対し，ID_iとランダムなs_iを生成し，s_i, cs_0及び更新回数0をライタを通じてTag_iに書き込む．
3. Bは，ID_iとs_iを関連づけて保存し，また，cs_0も保存する．

[Identification phase]（図2参照）

1. リーダは，$Tag_{i'}$にID取得要求を出す．
2. $Tag_{i'}$は，現在保持しているシード$cs_{k'}(0 \leq k' < K)$と秘密情報$s_{i'}$から，秘匿ID $r_{i'}^{k'} = G(s_{i'} \| cs_{k'})$を生成する．
3. $Tag_{i'}$は$(r_{i'}^{k'}, k')$を，リーダを通じてBに送信する．
4. $Tag_{i'}$は，シードを$cs_{k'+1} = H(cs_{k'})$のように更新し，更新回数も$k' \leftarrow k' + 1$とする．
5. $(r_{i'}^{k'}, k')$を受信したBは，$cs_{k'} = H^{k'}(cs_0)$を計算し，すべてのiに対して$G(s_i \| cs_{k'})$を算出し，$r_{i'}^{k'}$と比較する．
6. $r_{i'}^{k'} = G(s_i \| cs_{k'})$となる$i$が存在した場合，$ID_i$

表1 OSK方式の効率
Table 1 Efficiency of OSK scheme.

	メモリ量	最悪計算量	事前計算量
完全探索	$I\|s\|$	$2IK$	0
Lookup table	$IK(\|s\| + \log_2 I)$	$\log_2 IK$ *	$2IK$

* 二分探索の回数

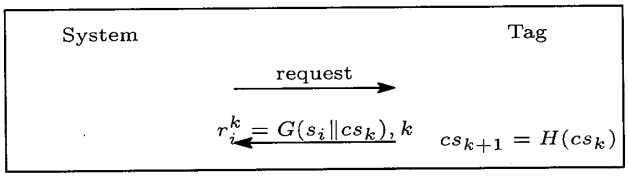


図2 OSK2方式のIdentification phase
Fig. 2 Identification phase of OSK2 scheme.

表 2 OSK2 方式の効率
Table 2 Efficiency of OSK2 scheme.

	メモリ量	最悪計算量	事前計算量
完全探索	$I\|s\|+\|s\|$	$I+K$	0
Lookup table	$IK(\|s\|+\log_2 I)$	$\log_2 I$ *	IK

* 二分探索の回数

をリーダに送信する．存在しなければ認証失敗とする．

OSK2 方式も，Lookup table を用いることで ID 認証の計算量を減らすことができる．OSK2/lookup table では，Setup phase において，すべての秘匿 ID r_i^k を計算しておき，ペア (r_i^k, i) を k の値ごとにグループ分けし，各グループ内で r_i^k の値に関してソートしておく．

OSK2/完全探索及び OSK2/Lookup table における事前計算量，ID 認証の最悪計算量及び必要メモリは，表 2 のとおりである．

3. OSK/AO 方式

OSK/完全探索では，必要なメモリ量は小さいが ID 認証の際に時間がかかり，OSK/Lookup table では，ID 認証は高速だが必要なメモリ量が大きくなるため，利用者の環境による柔軟な適用ができず，タグ数の多い大規模システムでの運用に問題が生じる可能性がある．

そこで，暗号解読などで使われる Time-memory Trade-off 方式 [3], [4] をこの ID 認証に適用させることで，メモリ量と ID 認証の時間を調整できる OSK/AO 方式が提案された [6], [7].

3.1 Time-memory Trade-off

Time-memory Trade-off 方式は暗号解読における鍵探索の一手法として，Hellman によって提案され [3]，Oechslin によってその改良版が提案された [4]．この方式は，完全探索，Lookup table といった従来の探索法と異なり，使用するメモリ量を利用者が決定でき，それに応じて，事前計算量や鍵探索の計算量が決定される．ただし，確率的アルゴリズムであり，成功確率は使用するメモリ量に依存している．以下に，鍵空間の大きさが N であるときの暗号鍵探索を，各探索法を適用し実行する場合の必要リソース量を示す (Time-memory Trade-off 方式の場合は，典型的な値を示す)．

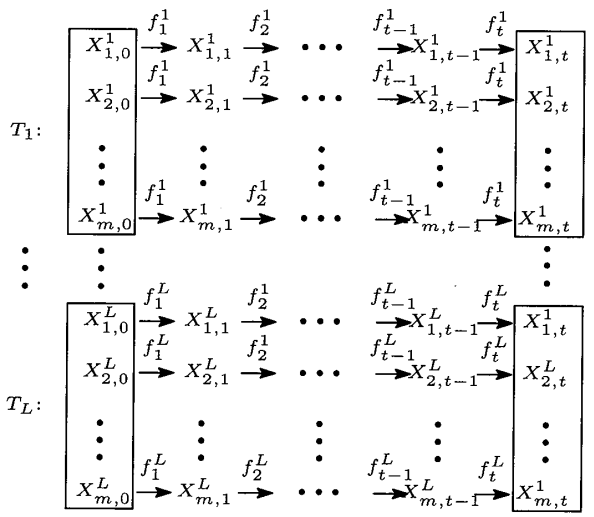


図 3 (perfect) rainbow table
Fig. 3 (Perfect) rainbow table.

	メモリ量	計算量	事前計算量	成功確率
完全探索	$O(1)$	$O(N)$	$O(1)$	1
Lookup table	$O(N)$	$O(1)$	$O(N)$	1
Time-memory Trade-off	$O(N^{\frac{2}{3}})$	$O(N^{\frac{2}{3}})$	$O(N)$	0.5

この表から分かるように，Time-memory Trade-off 方式は，メモリ量，計算量に関して完全探索と Lookup table の中間値をとることに成功している．

[3] で提案された Time-memory Trade-off 方式では，事前計算において，図 3 のような長さ t の鍵の chain を m 本含む table を L 個生成し，生成した table の最初の列 $(X_{1,0}^1, \ldots X_{m,0}^L)$ と最後の列 $(X_{1,t}^1, \ldots X_{m,t}^L)$ のみ保存することによりメモリ量を節約している．ただし，table T_l の生成には，関数 $f^l = f_1^l = \cdots = f_t^l$ を用いる．[4] では perfect rainbow table を用いた方式が提案された．Rainbow table は，図 3 のように各 table の各列ごとに異なる tL 種類の関数 f を利用し作成される．更に，perfect rainbow table は，rainbow table の各 table の各列の値がすべて異なるように構成されたもので，同じメモリ量に対して成功確率が高くなる．

3.2 OSK/AO 方式

Avoine と Oechslin は，perfect rainbow table を用いた Time-memory Trade-off 方式を OSK 方式の ID 認証に適用し，OSK/AO 方式を構成した [6]．以下で，OSK/AO 方式で用いる記号の定義を行う．

P : 1 回の Identification phase での成功確率．
x : Perfect rainbow table 以外に B が保存する

情報の量を示すパラメータ．ただし

$$1 \leq x \leq K. \tag{1}$$

t ： Setup phaseで生成されるchainの長さ．

m ： Setup phaseで生成されるtable一つ当りのchainの本数．ただし次の上限がある[5]．

$$m \leq \frac{2IK}{t+2}$$

L ： Setup phaseで生成されるtableの個数．ただし次の下限がある[5]．

$$L \geq \left\lceil \frac{-\ln(1-P)}{2} \right\rceil$$

以下にOSK/AO方式のプロトコルを示す．

[Setup phase]

1. 各$i(0 \leq i < I)$に対し，BはID_iとランダムなs_i^0を生成し，ライタを通じてTag_iにs_i^0を書き込む．
2. (**Preprocessing1**) $K_\mathrm{d} = \left\lfloor \frac{K}{x} \right\rfloor$とする．$B$は，$s_i^0$から$k = K_\mathrm{d}, 2K_\mathrm{d}, \ldots, (x-1)K_\mathrm{d}$に対する$s_i^k$を計算し，これらを保存する．
3. (**Preprocessing2**) Bは，$(i,k) \in Z_I \times Z_K$を鍵とし，図3のようなperfect rainbow tableを作成する．ただし，適切な$R : \{0,1\}^{|s|} \mapsto (i',k') \in Z_I \times Z_K$に対して

$$F : (i,k) \mapsto r_i^k = G(H^k(s_i^0))$$
$$f : (i,k) \mapsto R(F(i,k))$$

とする．

[Identification phase]

1. リーダは，$Tag_{i'}$にID取得要求を出す．
2. $Tag_{i'}$は，現在保持している秘密情報$s_{i'}^{k'}(0 \leq k' < K)$から，秘匿ID $r_{i'}^{k'} = G(s_{i'}^{k'})$を生成する．
3. $Tag_{i'}$は$r_{i'}^{k'}$を，リーダを通じてBに送信する．
4. $Tag_{i'}$は，秘密情報を$s_{i'}^{k'+1} = H(s_{i'}^{k'})$によって更新する．
5. (**Identification**) $r_{i'}^{k'}$を受け取ったBは，Time-memory Trade-off方式の探索を行い，$r_{i'}^{k'} = r_i^k$となる(i,k)のペアを求める．ペアを出力できないときは認証失敗とする．
6. Bは，求めた(i,k)から，ID_iをリーダに送信する．

パラメータx, m, t, Lを自由に設定することで，必要とされるメモリ量M_{AO}，ID認証の計算量，認証の成功確率Pが決まる．Avoineらは，$I, K, P, |s|$及びBが準備できるメモリ量M_{AO}が与えられ，rainbow tableの数Lを最小としたとき，ID認証の計算量を

$$T_{\mathrm{ave}}^{AO} = \frac{2K^3(\log IK)^2\gamma}{x(\frac{M_{AO}}{I|s|} - x)^2|s|^2} \tag{2}$$

と見積もっている．ただし，これはStep 5における平均の計算量であり，γは成功確率Pに依存する係数で，例えば，$P = 0.999$のとき$\gamma = 8$である．またAvoineらはこの平均計算量T_{ave}^{AO}を最小にするようにパラメータを最適化し，

$$x = \frac{M_{AO}}{3I|s|}$$

のとき，最小値

$$T_{\mathrm{ave}}^{AO} \approx \frac{3^3 I^3 K^3 (\log_2 IK)^2 |s| \gamma}{2M_{AO}^3}$$

をとることを示している．

事前計算量T_{pre}^{AO}については，成功確率をほぼ1とするためにはL個のrainbow table内にIK通り以上の値$R(r_i^k)$が存在する必要があり，一つの$r_i^k = G(H^k(S_i^0))$を計算するためには平均$K/2$回のハッシュ計算が必要であるため，およそ$IK^2/2$回程度のハッシュ計算が必要となるとし，以下のように評価されている．

$$T_{\mathrm{pre}}^{AO} \approx IK^2/2 \tag{3}$$

4. OSK/AO方式の計算量の再評価

本章では，オープンな環境での利用を前提として，OSK/AO方式の効率の再評価を行う．

まず，Avoineらは ID認証の平均計算量T_{ave}^{AO}を最小化するようにパラメータを決定していたが，本研究では，ID認証の最悪計算量T_{worst}^{AO}を最小化することを考える．T_{worst}^{AO}は，1回のIdentification phaseで認証プロセスが終了するまでにかかる最悪の計算量であり，認証が失敗する場合にあたる．オープンな環境を想定した場合，リーダが自システム以外のタグから情報を受け取ることが大いに考えられ，その場合は必ずID認証が失敗するまで計算することになる．したがってT_{worst}^{AO}が大きい場合，システムが円滑に機能しなくなる危険性があり，システムを評価する上で最悪

計算量 T^{AO}_{worst} が重要となると考える．ただし，Avoine らと同様に，$I, K, P, |s|$ 及び M_{AO} が与えられるものとして最適化をする．

次に，事前計算量 T^{AO}_{pre} の再評価を行う．前章で示したように，Avoine らは，T^{AO}_{pre} を式 (3) と評価しているが，この見積りは以下の 3 点で不正確であると考える．

- Preprocessing1 の計算量を考慮していない．
- Preprocessing1 で計算した s^k_i を利用することにより，rainbow table 作成のための計算量が大幅に削減できることを考慮していない．
- Perfect rainbow table を作成する際に，衝突が起きて計算をし直す確率を考慮していない．

そこで本章では，詳細のアルゴリズムを示すことにより T^{AO}_{pre} をより厳密に評価する．

4.1 最悪計算量の導出と最適化

まず，最悪計算量 T^{AO}_{worst} を付録 **1.** に示すアルゴリズム Identification(AO) に基づいて示す．

Identification(AO) において最も計算量が大きくなるのは，table 内の各列を順に検索したときに毎回 (i,k) の候補が見つかるが false alarm [3] が生じ，最後まで認証が成功しないときである．

関数 f の計算を 1 回実行する際に，ハッシュ関数は最悪 $\frac{K}{x}$ 回実行されることに注意すると，T^{AO}_{worst} は以下で表せる．

$$T^{AO}_{\mathrm{worst}} = t(t-1)L\frac{K}{x} \approx t^2 L\frac{K}{x} \tag{4}$$

一方，パラメータ m, t, L, x と必要なメモリ量 M_{AO}，成功確率 P は，以下の関係を満たす．

$$M_{AO} = I|s|x + 2(\log_2 IK)mL \tag{5}$$

$$P = 1 - \left(1 - \frac{m}{IK}\right)^{tL} \approx 1 - e^{-tL\frac{m}{IK}} \tag{6}$$

M_{AO} と P が与えられるので，これらを用いて式 (4) から t を消去すると，T^{AO}_{worst} はパラメータ x, L によって(注2)

$$T^{AO}_{\mathrm{worst}} = \frac{2^2 K^3 L(\log_2 IK)^2(-\ln(1-P))^2}{x(\frac{M_{AO}}{I|s|} - x)^2|s|^2}$$

で表される．これは式 (2) の $L\times$const 倍となっており，T^{AO}_{worst} を最小にする x, L は，平均計算量を最小化する場合と同様，

$$x = \frac{M_{AO}}{3I|s|}$$

$$L = \left\lceil \frac{-\ln(1-P)}{2} \right\rceil$$

であることが分かる．

このとき，m, t は $mt \approx 2IK$ を満たし，

$$m = \frac{M_{AO}}{3\lceil(-\ln(1-P))/2\rceil(\log_2 IK)}$$

$$t = \frac{3(-\ln(1-P))IK(\log_2 IK)}{M_{AO}}$$

となる．このパラメータを用いた場合の最悪計算量は，

$$T^{AO}_{\mathrm{worst}} \approx \frac{3^3 I^3 K^3 (\log_2 IK)^2 |s| (-\ln(1-P))^3}{2M^3_{AO}}$$

で表される（e は自然対数の底）．

4.2 事前計算量の導出

次に，事前計算量 T^{AO}_{pre} を付録 **1.** に示すアルゴリズム Preprocessing1(AO)，Preprocessing2(AO) に基づいて示す．

まず Preprocessing1(AO) の計算量 T^{AO}_{pre1} は，

$$T^{AO}_{\mathrm{pre1}} = I(K-1)\frac{x-1}{x} \approx IK\frac{x-1}{x} \tag{7}$$

と表せる．次に，Preprocessing2(AO) において，p 本目の chain を作成しているときに他の chain と衝突する確率は，以下のように求められる．

$$1 - \left(1 - \frac{p-1}{IK}\right)^t \approx 1 - e^{-t\frac{p-1}{IK}}$$

よって，Preprosessing2 に要する計算量 T^{AO}_{pre2} は，

$$\begin{aligned} T^{AO}_{\mathrm{pre2}} &\approx \frac{1}{2}\left(1 + \frac{K}{x}\right) L \sum_{p=1}^{m} t e^{t\frac{p-1}{IK}} \\ &= \frac{1}{2}\left(1 + \frac{K}{x}\right) Lt \frac{e^{\frac{mt}{IK}} - 1}{e^{\frac{t}{IK}} - 1} \\ &\approx \frac{1}{2}\left(1 + \frac{K}{x}\right) IKL(e^{\frac{mt}{IK}} - 1) \end{aligned} \tag{8}$$

となる（$t/IK \ll 1$ を用いた）．

一般に M_{AO} の適用範囲（**5.** 参照）において $T^{AO}_{\mathrm{pre1}} \ll T^{AO}_{\mathrm{pre2}}$ であることを考慮すると，事前計算量は

$$T^{AO}_{\mathrm{pre}} \approx \frac{1}{2}\left(1 + \frac{K}{x}\right) IKL(e^{\frac{mt}{IK}} - 1)$$

である．したがって最適パラメータにおける事前計算量は

(注2)：従来研究では L は最小値をとるものとしているが，ここでは可変のパラメータとしている．

表3 OSK/AO方式の効率（小規模システム）
Table 3 Efficiency of OSK/AO scheme. (Small-scale system)

	メモリ量	最悪計算時間	事前計算時間	成功確率
完全探索	16 KB	0.13 s	0	1
OSK/AO(*)	64 KB	28.3 s	5.1 minutes	0.999
	256 KB	0.44 s	1.3 minutes	
	1 MB	6.9 ms	20.0 s	
	4 MB	0.11 ms	5.6 s	
Lookup table	8.8 MB	fast	0.13 s	1

(*) 来研究の見積もりによると事前計算量はメモリ量に依存せず16 s.

$$T_{\mathrm{pre}}^{AO} \approx 3.2IK\left(1+\frac{3IK|s|}{M_{AO}}\right)\left\lceil\frac{-\ln(1-P)}{2}\right\rceil \tag{9}$$

で与えられる．

Avoineらの見積り（式(3)）では，事前計算量はタグ数Iと最大更新回数Kのみに依存する．しかし詳細に見積もることにより（式(9)），ID認証に必要とされる成功確率Pが1に近いほど事前計算量が多くなること，準備できるメモリ量が大きいほど事前計算量が少なくてすむことが明確となった．

5. OSK方式とOSK/AO方式の比較

本章では，タグ数Iが約1000個，約100万個，約1億個の三つの規模のRFIDシステムを想定し，それぞれにおいてOSK/完全探索，OSK/lookup table, OSK/AOの比較を行う．

本研究では，下記の値を用いるとする．

- 最大更新回数：$K = 2^9$（約500回）(注3)
- 秘密情報のビット長：$|s| = 64\left\lceil\frac{4\log_2 IK}{64}\right\rceil$ (注4)
- 成功確率：$P = 0.999$（OSK/AOのみ）(注5)

また，OSK/AO方式で使用するメモリ量M_{AO}は，OSK/完全探索で要するメモリ量より多く，OSK/Lookup tableで要するより少ない範囲内とする．

計算時間は，1秒当りハッシュ関数H, Gを2^{23}回計算可能として計算する．

(a) 小規模システムへ適用する場合

$I = 2^{10}$（約1000個）の小規模システムへ適用した場合の計算時間は表3となる．この程度の規模のシステムの場合，どの方式でも用いることができ，成功確率が100 %である完全探索またはLookup tableを用いることを推奨する．

(b) 中規模システムへ適用する場合

$I = 2^{20}$（約100万個）の中規模システムの場合，

表4 OSK/AO方式の効率（中規模システム）
Table 4 Efficiency of OSK/AO scheme. (Medium-scale system)

	メモリ量	最悪計算時間	事前計算時間	成功確率
完全探索	16 MB	128 s	0	1
OSK/AO(*)	64 MB	66.0 s	3.7 days	0.999
	256 MB	1.0 s	22.0 hours	
	1 GB	16.1 ms	5.7 hours	
	4 GB	0.25 ms	1.6 hours	
Lookup table	9.5 GB	fast	128 s	1

(*) 従来研究の見積もりによると事前計算量はメモリ量に依存せず4.6 hours.

表5 OSK/AO方式の効率（大規模システム）
Table 5 Efficiency of OSK/AO scheme. (Large-scale system)

	メモリ量	最悪計算時間	事前計算時間	成功確率
完全探索	2.3 GB	3.4 hours	0	1
OSK/AO(*)	16 GB	7.9 s	197.5 days	0.999
	64 GB	0.12 s	50.1 days	
	256 GB	1.92 ms	13.2 days	
Lookup table	1.3 TB	fast	3.4 hours	1

(*) 従来研究の見積もりによると事前計算量はメモリ量に依存せず18.2 days.

表4から，OSK/完全探索は，最悪計算時間が大きく現実的ではないが，OSK/Lookup table及びOSK/AOは十分適応可能であることが分かる．したがって，この場合も成功確率が100 %であるOSK/Lookup tableを用いるのがよいと考えられる．

(c) 大規模システムへ適用する場合

$I = 3\cdot 2^{25}$（約1億個）への大規模システムへの適用を考えた場合，表5より，OSK/完全探索やOSK/Lookup tableでは実現が困難であることが分かる．また，OSK/AOの場合はメモリ量，最悪計算時間は実用可能な値になっているが，事前計算時間が膨大になっている．したがって，大量のメモリを用意できない場合や事前計算時間を十分用意できない場合には，大規模なRFIDシステムの実現は困難であると考えられる．

(注3)：データベースBによって定期的にSetup phaseが実行されない場合，各タグの更新回数はリセットされないため，最大更新回数Kは非常に大きな値をとる必要があり，現実的でない．したがって，定期的なSetup phaseの実行が必要であり，そのSetup phase実行間隔にタグがリーダによってアクセスされるであろう回数をKに設定する．
(注4)：秘匿IDの衝突が起こらないために，$|s|$はあまり小さくすることはできない．$|s| \geq 4\log_2 IK$が望ましいとされる[6]．またタグ容量として区切りをよくするため，64ビット単位とした．
(注5)：$P = 1$とした場合，パラメータt, Lが無限大に大きくなってしまうため，Pをできる限り1に近い値とすることで対応する．

6. OSK/AO 方式の効率改善策の提案

本章では，OSK/AO 方式の効率改善策を提案する．

6.1 アイデア

これまで，タグの ID 認証の成功確率は，1 回の Identification phase での成功確率 P で評価してきた．P には非常に高い成功確率が要求され，その結果，**5.** で示したように事前計算量が膨大となった．

改善方法として，ID 認証に失敗した場合には再びタグにアクセスし，異なる秘匿 ID を受け取り，再度 ID 認証を実行することを考える．アクセスの最大回数を $w(\geq 1)$ とした場合，実際の ID 認証成功確率 P' は，

$$P' = 1 - (1-P)^w$$

となるため，P' を 0.999 とした場合でも 1 回の Identification phase の成功確率 P は比較的低い値でよく，計算量，メモリ量を削減することができる．

6.2 パラメータの決定

ID 認証の最悪計算量を最小化するためのパラメータ決定は，**4.** とほぼ同様に行うが，パラメータ w が追加されるため，以下の変更が必要である．

まず，期待される ID 認証の成功確率を P' としたとき，1 回の Identicfication phase の成功確率として

$$P = 1 - (1-P')^{\frac{1}{w}} \tag{10}$$

を用いる．

また，実際のタグの ID 認証の最大回数を K' としたとき，タグの最大更新回数 K は K' より大きく設定する必要がある．ID 認証が成功するまでのタグへのアクセス数の期待値を v としたとき，$K = K'v$ とすればよい．v は以下で与えられる．

$$\begin{aligned} v &= P + 2P(1-P) + \cdots + w(1-P)^w \\ &= \frac{1-(1-P)^w}{P} = \frac{P'}{1-(1-P')^{\frac{1}{w}}} \end{aligned}$$

ID 認証の最悪計算量は，1 回の Identicitation phase の最悪計算量である T^{AO}_{worst} を用いることができない．1 回の ID 認証につき最大 w 回 Identification phase が失敗するため，$T'^{AO}_{\mathrm{worst}} = wT^{AO}_{\mathrm{worst}}$ を ID 認証の最悪計算量とする．

このような条件のもと，適用する RFID システムによって決定された P', I, K', M_{AO} に対して，最悪計算量 T'^{AO}_{worst}，事前計算量 T^{AO}_{pre} がシステムの期待する値になるように w の値を決定すればよい．

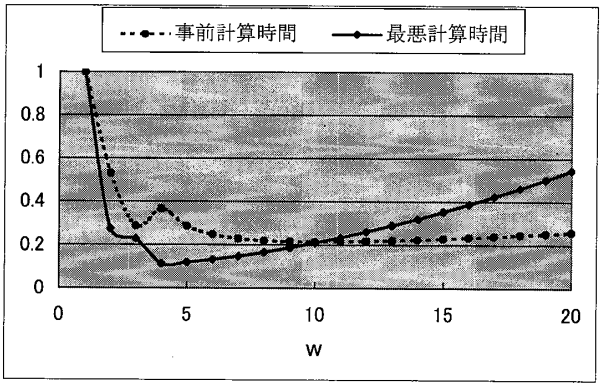


図 4 OSK/AO 方式の繰返しアクセスによる改善
Fig. 4 Improvement of OSK/AO scheme by iterating access.

6.3 効率の評価

5. の大規模システムの数値例の場合において(注6)，$1 < w \leq 20$ として事前計算量と最悪計算量を見積もった結果を図 4 に示す．なお，効率の改善効果を示す縦軸は $w = 1$（繰返しを許さない場合）の計算量を基準にしてグラフ化し，使用するメモリ量 M_{AO} に依存しない値となっている．図 4 より，$w = 10$ のとき最悪計算量と事前計算量がともに約 $\frac{1}{5}$ になっており，効率が改善されているのが分かる．

ただし，この改善策では，リーダとタグ間で複数回の通信をする場合があり，計算時間に比べて通信時間の時間が大きい場合，ID 認証に要する時間は短縮することができない．

7. OSK 方式の変形に対する Time-memory Trade-off 方式適用の提案

本章では，前章とは別の効率改善策として，perfect rainbow table を用いた Time-memory Trade-off 方式を OSK 方式の変形である OSK2 方式の ID 認証に適用した OSK2/AO 方式を提案する．

7.1 OSK2/AO 方式

OSK2/AO 方式を以下に示す．ただし，OSK2/完全探索からの差分のみを示す．

[Setup phase]

3. (**Preprocessing1**) $K_{\mathrm{d}} = \left\lfloor \frac{K}{x} \right\rfloor$ とする．B は，cs_0 から $k = K_{\mathrm{d}}, 2K_{\mathrm{d}}, \ldots, (x-1)K_{\mathrm{d}}$ に対する cs_k を計算し，これらを保存する．
4. (**Preprocessing2**) B は，各 $k(0 \leq k < K)$ に対し，$i \in Z_I$ を鍵とする図 3 のような L 個の

(注6)：$P' = 0.999, I = 3 \cdot 2^{25}, K' = 2^9$ とする．

perfect rainbow tableを作成する（したがって，KL個のtableが作成される）．ただし，適切な$R:\{0,1\}^{|s|}\mapsto i'\in Z_I$に対して

$$F:i\mapsto r_i^k=G(s_i\|cs_k)$$
$$f:i\mapsto R(F(i))$$

とする．

[Identification phase]

5. (**Identification**) $(k',r_{i'}^{k'})$を受け取ったBは，Time-memory Trade-off方式の探索を行い，$r_{i'}^{k'}=r_i^{k'}$となるiを求める．iを出力できないときは認証失敗とする．
6. Bは，ID_iをリーダに送信する．

7.2 最悪計算量，事前計算量の導出

最悪計算量$T_{\rm worst}^{AO2}$と事前計算量$T_{\rm pre}^{AO2}$を付録**2.**に示すアルゴリズムPreprocessing1(AO2)，Preprocessing2(AO2)，Identification(AO2)に基づいて示す．

まず，Identification(AO2)において，最も計算時間が長くなるのは毎回false alarm [3]が生じ，最後まで認証が成功しないときである．関数fの計算を1回実行する際に，ハッシュ関数は最悪$\frac{K}{x}$回実行されることに注意すると，$T_{\rm worst}^{AO2}$は以下のように表せる．

$$T_{\rm worst}^{AO2}=t(t-1)L\approx t^2L \tag{11}$$

次に，$T_{\rm pre1}^{AO2}$は，

$$T_{\rm pre1}^{AO2}=x-1\approx x$$

と表せる．Preprocessing2(AO2)において，p本目のchainを作成しているときに他のchainと衝突する確率は，以下のように求められる．

$$1-\left(1-\frac{p-1}{I}\right)^t\approx 1-e^{-t\frac{p-1}{I}}$$

よって，$T_{\rm pre2}^{AO2}$は，

$$\begin{aligned}T_{\rm pre2}^{AO2}&\approx\frac{1}{2}\left(1+\frac{K}{x}\right)KL\sum_{p=1}^{m}te^{t\frac{p-1}{I}}\\&=\frac{1}{2}\left(1+\frac{K}{x}\right)KLt\,\frac{e^{\frac{mt}{I}}-1}{e^{\frac{t}{I}}-1}\\&\approx\frac{1}{2}\left(1+\frac{K}{x}\right)IKL(e^{\frac{mt}{I}}-1)\end{aligned}$$

となる．$x\le K$であるから，$T_{\rm pre}^{AO2}=T_{\rm pre1}^{AO2}+T_{\rm pre2}^{AO2}\approx T_{\rm pre2}^{AO2}$として考える．

必要なメモリ量，成功確率Pは

$$M_{AO2}=I|s|+x|s|+2(\log_2 I)mKL$$
$$P=1-\left(1-\frac{m}{I}\right)^{tL}\approx 1-e^{-tL\frac{m}{I}} \tag{12}$$

である．ただし，$x\le K\ll I$と仮定すれば

$$M_{AO2}\approx I|s|+2(\log_2 I)mKL \tag{13}$$

としてよい．

M_{AO2}及びPが与えられたとき，式(13)，(12)からmLとtが

$$mL=\frac{M_{AO2}-I|s|}{2(\log_2 I)K} \tag{14}$$

$$t=\frac{2IK(-\ln(1-P))(\log_2 I)}{M_{AO2}-I|s|} \tag{15}$$

と定まる．xは自由に選ぶことができるため，計算量を最小化する観点から最大の

$$x=K \tag{16}$$

とすればよい．このとき，

$$T_{\rm pre2}^{AO2}\approx IKL(e^{\frac{mt}{I}}-1)=IKL(e^{(-\ln(1-P))/L}-1). \tag{17}$$

7.3 最悪計算量による最適化

$T_{\rm worst}^{AO2}$を最小化するためには，Lを最小とすればよく

$$L_{\rm mim}=\left\lceil\frac{-\ln(1-P)}{2}\right\rceil$$

となる．

$I=3\cdot 2^{25}$（約1億個）の大規模システムへの適用を想定し，$K=2^9$（約500回），$|s|=192\,bit$, $P=0.999$とし，1秒当りhashを2^{23}回計算できるとして，いくつかのM_{AO2}に対して各ID認証方式で必要とされるメモリ量と計算時間を見積もった結果を表6に示す．これにより，OSK2/AO方式はOSK/AO方式に比べ，大幅に効率が改善され，大規模システムへの適用を想定した場合においても現実的な値となることが分かった．

7.4 事前計算量による最適化

OSK/AO方式では，事前計算量が非常に大きくなった．また，表6の最悪計算時間はかなり小さい値と

表 6 OSK2/AO の効率（最悪計算量による最適化）
Table 6 Efficiency of OSK2/AO scheme. (Minimizing the worst computation cost)

	メモリ量	最悪計算時間	事前計算時間	成功確率
完全探索	2.25 GByte	12 s	0	1
OSK2/AO	16 GByte	12.2 ms	1.3 day	0.999
	64 GByte	0.61 ms		
	256 GByte	35.9 μs		
Lookup table	1.3 TByte	fast	1.7 hour	1

表 7 OSK2/AO の効率（事前計算量による最適化）
Table 7 Efficiency of OSK2/AO scheme. (Minimizing the precomputation cost)

	メモリ量	最悪計算時間	事前計算時間	成功確率
OSK2/AO	16 GByte	0.1 s	12.8 hour	0.999
	64 GByte		11.8 hour	
	256 GByte		11.8 hour	

なっており，最悪計算時間を最小とする必要がない場合もあると考えられる．そこで，許容最悪計算量 $\bar{T}^{AO2}_{\rm worst}$ を与え，$T^{AO2}_{\rm worst} \leq \bar{T}^{AO2}_{\rm worst}$ という条件で事前計算量を最小化することを考える．

式 (14)，(15) を満たしつつ事前計算量 $T^{AO2}_{\rm pre}$ を最小化するためには，L は大きい方がよい．したがって，

$$L_{\max} = \left\lceil \bar{T}^{AO2}_{\rm worst}/t^2 \right\rceil$$

とすればよい．ただし，t は式 (15) で与えられる値．

これまでと同様の大規模システムへの適用を想定した場合のメモリ量と計算時間を表 7 に示す．なお，許容最悪計算量は 2^{20}（約 0.1 s）とした．表から，最悪計算量を許容範囲内に押さえつつ，事前計算量を軽減させることができることが分かる．また，最悪計算量を約 0.1 秒以内とした場合には，多くのメモリ量を使用しても事前計算量はほぼ変化がないことから，63 GByte 程度のメモリを準備すれば十分であることが分かる．

7.5 安全性の考察

OSK2/AO 方式は，OSK/AO 方式に比べ少ない計算量で ID 認証ができることが分かったが，安全性が低下していることに留意する必要がある．

第 1 に，OSK2 方式ではタグが現在の更新回数を出力するため，プライバシー保護の観点から望ましくない可能性がある．ただし，複数回アクセスを許す拡張をした場合，更新回数は，ID 認証の回数とは一致しないため，この問題は軽減されると思われる．

第 2 に，すべてのタグで共通のシードを用いているため，一つのタグからシードが解析された場合，他のタグのフォワードセキュア性を満たさなくなる．具体的なアタックの手法を以下に示す．

1. タグ Tag_i からシード cs_k を解析する．
2. 現在の更新回数が $k'(>k)$ であるタグ $Tag_{i'}(i' \neq i)$ から秘密情報 $s_{i'}$ を解析する．
3. 1., 2. の解析結果から，タグ $Tag_{i'}$ の過去の送信履歴を推測することができる．

8. むすび

本論文では，OSK/AO 方式の事前計算と ID 認証の具体的なアルゴリズムに基づいて効率の再評価を行い，OSK/完全探索，OSK/Lookup table と比較した．その結果，大規模システムでの利用を想定した場合，事前計算時間が膨大となってしまうという問題が生じることが分かった．

この問題点に対し，二つの解決策を提案した．一つは，ID 認証の際に繰り返しアクセスをするものであり，もう一つは，OSK 方式の変形である OSK2 方式に perfect rainbow table を用いた Time-memory Trade-off 方式を適用するものである．

前者の改善策によって，事前計算時間を 1/5 程度まで軽減することができる．ただし，タグ–リーダ–データベース間の通信回数が増えるため，通信時間が大きい場合にはあまり効果的ではない．

一方，後者の改善策は通信時間による制限がなく効果を発する．また，通信時間を無視できる場合は，繰返しアクセスによる改善を組み合わせることで，大きな効率改善が可能である．ただし，特殊な環境においてはフォアワードセキュリティを満たさないなど，安全性が若干低いことも指摘した．

文　献

[1] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Cryptographic approach to "privacy-friendly" tags," RFID Privacy Workshop, MIT, Massachusetts, USA, Nov. 2003.

[2] M. Ohkubo, K. Suzuki, and S. Kinoshita, "Hash-chain based forward-secure privacy protection scheme for low-cost RFID," Proc. SCIS2004, pp.719–724, 2004.

[3] M. Hellman, "A cryptanalytic time-memory trade off," IEEE Trans. Inf. Theory, vol.IT-26, no.4, pp.401–406, 1980.

[4] P. Oechslin, "Making a faster cryptanalytic time-memory trade-off," Advances in Cryptology, Proc. Crypto 2003, LNCS vol.2729, pp.617–630, Springer-Verlag, 2003.

[5] G. Avoine, P. Junod, and P. Oechslin, "Time-memory

trade-offs: False alarms detection using checkpoints," Progress in Cryptology, Proc. Indocrypt 2005, LNCS vol.3797, pp.183–196, Springer-Verlag, 2005.

[6] G. Avoine and P. Oechslin, "A scalable and provably secure hash based RFID protocol," International Workshop on Pervasive Computing and Communication Security. PerSec 2005, pp.110–114, Hawaii, USA, March 2005.

[7] G. Avoine, E. Dysli, and P. Oechslin, "Reducing time complexity in RFID systems," Selected Areas in Cryptography, SAC 2005, LNCS vol.3897, pp.291–306, Springer-Verlag, 2005.

[8] S.A. Weis, Security and Privacy in Radio-Frequency Identification Devices, Masters thesis, MIT, May 2003.

[9] S.A. Weis, S.E. Sarma, R.L. Rivest, and D.W. Engels, "Security and privacy aspects of low-cost radio frequency identification systems," Security in Pervasive Computing, LNCS, vol.2802, pp.201–212, Springer-Verlag, 2004.

[10] G. Tsudik, "YA-TRAP: Yet another trivial RFID authentication protocol," Fourth IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOMW '06), pp.640–643, 2006.

[11] T. Nohara, S. Inoue, K. Baba, and H. Yasuura, "Quantitative evaluation of unlinkable frequency schemes," Workshop on Privacy in the Electronic Society (WPES), pp.55–60, 2005.

付　　録

1. OSK/AO方式の具体的なアルゴリズム

OSK/AO方式の計算量の評価に使用する具体的なアルゴリズムを示す.

Preprocessing1(AO)

Input:H, K, I, $s_i^0(0 \le i < I)$, x
Output: $K_{\mathrm{d}}, \hat{s}_i^q(0 \le q < x, 0 \le i < I)$

$K_{\mathrm{d}} \leftarrow \left\lfloor \frac{K}{x} \right\rfloor$
for i=0 to $I-1$ **do**
  $s \leftarrow s_i^0, \hat{s}_i^0 \leftarrow s$
  for q=1 to $x-1$ **do**
    for z=1 to K_{d} **do**
      $s \leftarrow H(s)$
    end for
    $\hat{s}_i^q \leftarrow s$
  end for
end for

Preprocessing2(AO)

Input: G, H, I, K, m, t, L, K_{d}, $\hat{s}_i^q(0 \le q < x, 0 \le i < I)$
Output: F, R_j^l, f_j^l, $T_l(1 \le l < L, 1 \le j < t)$

define function F as $F : (i,k) \mapsto G(H^{(k \bmod K_{\mathrm{d}})}(\hat{s}_i^{k \% K_{\mathrm{d}}}))$
for l=1 to L **do**
  for j=1 to t **do**
    pick a random reduction function R_j^l and define function f_j^l as $f_j^l : (i,k) \mapsto R_j^l(F(i,k)) \in Z_I \times Z_K$
  end for
  for p=1 to m **do**
    repeat
      repeat
        pick $i' \in Z_I, k' \in Z_K$ at random
      until T_l contains no (*,(i', k')) entry
      $(i,k) \leftarrow (i',k')$
      for j=1 to t **do**
        $(i,k) \leftarrow f_j^l(i,k)$
      end for
    until T_l contains no ((i,k),*)entry
    insert((i,k), (i',k')) in table T_l
  end for
end for

Identification(AO)

Input: $R_j^l, f_j^l, F, T_l(1 \le l \le L, 1 \le j \le t), y$
Output: (i',k') or $\perp$ /* $y = r_{i'}^{k'}$ */

for l=1 to L **do**
  for j=t down to 1 **do**
    $(i,k) \leftarrow R_j^l(y)$
    for p=$j+1$ to t **do**
      $(i,k) \leftarrow f_p^l(i,k)$
    end for
    if T_l contains ((i,k),*)entry **then**
      get ((i,k), (i',k'))entry from T_l
      For u=1 to j-1 **do**
        $(i',k') \leftarrow f_u^l(i',k')$
      end for
      $r \leftarrow F(i',k')$
      if $r = y$ **then**
        return (i',k') /* succeed */
      end if
    end if
  end for
end for
return $\perp$ /* fail */

2. OSK2/AO方式の具体的なアルゴリズム

OSK2/AO方式の計算量の評価に使用する具体的なアルゴリズムを示す.

Preprocessing1(AO2)

Input:H, K, cs_0, x
Output: $K_{\mathrm{d}}, \hat{cs}_q(0 \le q < x)$

$K_{\mathrm{d}} \leftarrow \left\lfloor \frac{K}{x} \right\rfloor$
$cs \leftarrow cs_0, \hat{cs}_0 \leftarrow cs$
for q=1 to $x-1$ **do**
  for z=1 to K_{d} **do**
    $cs \leftarrow H(cs)$
  end for
  $\hat{cs}_q \leftarrow cs$
end for

Preprocessing2(AO2)

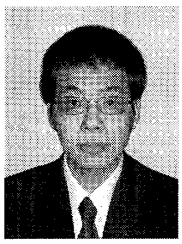
Input: G, H, I, K, m, t, L, x, $s_i(0 \le i < I)$, K_{d}, $\hat{cs}_q(0 \le q < x)$
Output: F, $R_j^{l,k}$, $f_j^{l,k}$,

$T_{l,k}(1 \le l \le L, 1 \le j \le t, 0 \le k < K)$
1: define function F as
  $F : (i,k) \mapsto G(s_i \| (H^{(k \bmod K_d)}(\hat{cs}_{k\%K_d})))$
2: **for** k=0 to $K-1$ **do**
3: **for** l=1 to L **do**
4: **for** j=1 to t **do**
5: pick a random function $R_j^{l,k}$
  and define function $f_j^{l,k}$ as
  $f_j^{l,k} : i \mapsto R_j^{l,k}(F(i,k)) \in Z_I$
6: **end for**
7: **for** p=1 to m **do**
8: **repeat**
9: **repeat**
10: pick $i' \in Z_I$ at random
11: **until** $T_{l,k}$ contains no (*,i') entry
12: $i \leftarrow i'$
13: **for** j=1 to t **do**
14: $i \leftarrow f_j^{l,k}(i)$
15: **end for**
16: **until** $T_{l,k}$ contains no (i,*)entry
17: insert(i, i') in table $T_{l,k}$
18: **end for**
19: **end for**
20: **end for**

Identification(AO2)

Input: F, $R_j^{l,k}$, $f_j^{l,k}$, $T_{l,k}$,
  $(1 \le l \le L, 1 \le j \le t, 0 \le k < K)$, y, k'
Output: i' or $\perp$ /* $y = r_{i'}^{k'}$ */
1: **if** $k' \ge K$ **then**
2: **return** $\perp$ /* fail */
3: **for** l=1 to L **do**
4: **for** j=t down to 1 **do**
5: $i \leftarrow R_j^{l,k'}(y)$
6: **for** p=$j+1$ to t **do**
7: $i \leftarrow f_p^{l,k'}(i)$
8: **end for**
9: **if** $T_{l,k'}$ contains (i,*)entry **then**
10: get (i, i')entry from $T_{l,k'}$
11: **for** u=1 to $j-1$ **do**
12: $i' \leftarrow f_u^{l,k'k}(i')$
13: **end for**
14: $r \leftarrow F(i')$
15: **if** $r = y$ **then**
16: **return** i' /* succeed */
17: **end if**
18: **end if**
19: **end for**
20: **end for**
21: **return** $\perp$ /* fail */

(平成 19 年 6 月 5 日受付，10 月 1 日再受付)



阿部　正己

平 17 東工大・工・情報工学卒．平 19 同大大学院修士課程了．同年（株）NTT データ入社．現在，IT 基盤構築に従事．

尾形わかは　（正員）

平 3 東工大修士課程了．平 6 同博士課程了．平 7 姫路工大助手，平 12 東工大理財工学研究センター助教授．現在，同大イノベーションマネジメント研究科准教授．情報セキュリティ，暗号理論及び暗号プロトコルの研究に従事．