コンピュータセキュリティ 18−36
(2002. 7. 19)

# 実行ハードウェア確認タグ付きディジタル署名方式

宇根 正志[†*] 松本 勉[†‡]

†横浜国立大学大学院工学研究科 〒240-8501 横浜市保土ヶ谷区常盤台 79-5
‡横浜国立大学大学院環境情報研究院 〒240-8501 横浜市保土ヶ谷区常盤台 79-7
*日本銀行金融研究所 〒103-8660 東京都中央区日本橋本石町 2-1-1
E-mail: † une@mlab.jks.ynu.ac.jp, ‡ tsutomu@mlab.jks.ynu.ac.jp

**あらまし** ディジタル署名を安心して利用するためには，秘密鍵が漏洩した場合の対策について予め検討しておくことが重要である．本稿では，署名生成用ハードウェアから秘密鍵が漏洩したとしても，検証対象の署名が特定のハードウェアで生成されたか否かを確認可能なディジタル署名方式について検討する．計算が実行されたハードウェアの確認方法としては，松本・田中[12]が，物理的に複製困難なモジュールを利用した手法を提案している．本稿では，松本・田中[12]の手法をディジタル署名方式に適用したタグ付き署名方式について，そのセキュリティ要件を検討する．次に，秘密鍵漏洩への代表的な対策であるヒステリシス署名[9, 10]とフォワードセキュア署名方式[1, 2, 4, 5]を取り上げ，タグ付き署名方式との比較を行う．また，検討したセキュリティ要件を満たすディジタル署名方式の一実現形態を例示する．

**キーワード** 耐クローンモジュール，ディジタル署名，秘密鍵漏洩，ハードウェア

# A Digital Signature Scheme that can Verify Signing Hardware

Masashi UNE[†*] and Tsutomu MATSUMOTO[†‡]

† Graduate School of Engineering, Yokohama National University,
79-5 Tokiwadai, Hodogaya,Yokohama, 240-8501 Japan
‡ Graduate School of Environment and Information Sciences, Yokohama National University,
79-7 Tokiwadai, Hodogaya,Yokohama, 240-8501 Japan
*Institute for Monetary and Economic Studies, Bank of Japan,
2-1-1 Nihonbashi-Hongokucho, Chuo, Tokyo, 103-8660 Japan
E-mail: † une@mlab.jks.ynu.ac.jp, ‡ tsutomu@mlab.jks.ynu.ac.jp

**Abstract** In order to use digital signature securely, it is important to consider countermeasures against the compromise of a signing key. This paper discusses a digital signature scheme that can verify signing hardware even though the signing key was compromised. To verify hardware that carried out a computational process, Matsumoto and Tanaka[12] proposed a method that is based on a clone resistant module. In this paper, we apply the method of Matsumoto and Tanaka[12] to a digital signature scheme and consider its security requirements. Next, we compare the signature scheme that employs the method of Matsumoto and Tanaka[12] with the hysteresis signature[9, 10] and the forward-secure digital signature scheme[1, 2, 4, 5] from viewpoints of their effects and environments. We also show a protocol of the digital signature scheme that fulfills the security requirements.

**Keyword** clone resistant module, compromise of a signing key, digital signature, hardware

## 1. はじめに

ディジタル署名は，公開鍵暗号技術を利用してデータの一貫性確認やデータ作成者の本人確認を実現する技術である．わが国では 2001 年 4 月に「電子署名及び認証業務に関する法律」が施行され，ディジタル署名が一定条件の下で手書きの署名や押印と同等の法的効果をもつこととなったこともあり，ディジタル署名が一層注目を集めている．

ディジタル署名を安全に利用するためには，署名生成用の秘密鍵が第三者に漏洩しないように管理することが最も重要である．秘密鍵は IC カード等のハードウェアに格納されるケースが多い．しかし，タイミング解析や電力解析等の各種攻撃が提案されており[3]，IC カード等から秘密鍵が漏洩する可能性は否定できない[3, 6, 8]．また，署名方式やその実装方法にセキュリティ上の欠陥があった場合，公開鍵等の公開情報から秘密鍵が効率的に推定される可能性もある．秘密鍵が漏洩すると，その秘密鍵を用いて署名が偽造されるおそれがある．

ハードウェアに格納された秘密情報が漏洩した場合の対策として，松本・田中[12]は，暗号処理が特定のハードウェアで実行されたか否かを確認する方式（実行ハードウェア確認タグ方式）を提案している．実行ハードウェア確認タグ方式では，各ハードウェアを特定するためのデータと暗号処理結果のデータを結び付ける「実行ハードウェア確認タグ（以下，単にタグという）」と呼ばれるデータが生成され，タグを用いた検証によって暗号処理が実行されたハードウェアを確認する．各ハードウェアを特定するためのデータは「耐クローンモジュール」の出力によって生成される．

松本・田中[12]は，実行ハードウェア確認タグ方式をディジタル署名に応用可能であるとしているが，具体的な要件や実現方法を示していない．本稿では，実行ハードウェア確認タグ方式をディジタル署名に応用し，秘密鍵が漏洩しても署名生成が実行されたハードウェアを確認可能な「実行ハードウェア確認タグ付き署名方式（以下，タグ付き署名方式という）」の想定環境やセキュリティ要件を検討する．その上で，秘密鍵が漏洩した場合の代表的な対応策としてヒステリシス署名[9, 10]とフォワードセキュア署名方式[1, 2, 4, 5]を取り上げ，想定環境やセキュリティ要件に関してタグ付き署名方式と比較する．最後に，タグ付き署名方式の一実現形態を例示し，その安全性について考察する．

本稿の構成は次のとおりである．まず，2 において実行ハードウェア確認タグ方式の想定環境・目的や概要等について説明し，3 においてその方式をディジタル署名に適用する場合の想定環境やセキュリティ要件について検討する．4 においては，タグ付き署名方式をヒステリシス署名やフォワードセキュア署名方式と比較し，5 ではタグ付き署名方式の一実現形態を示し，その安全性について考察する．最後に 6 では今後の課題を整理する．

## 2. 実行ハードウェア確認タグ方式

### 2.1 想定環境・目的

松本・田中[12]が提案した実行ハードウェア確認タグ方式の最大の特徴は，暗号処理が実行されたハードウェアを確認するために耐クローンモジュールの利用を想定している点である．耐クローンモジュールは，入出力をもつハードウェアのモジュールであり，「耐クローンモジュール F が実現するランダム関数を func(F)と表すと，func(F)を実現する F 以外のモジュールを作製困難である」という性質を有するものとして定義される[12]．また，func(F)の出力を得るには F に入力を与えてその出力を観測するという方法しかなく，入出力集合のサイズが非常に大きいため，入出力の関係全体を示す表を作成するには膨大な時間と記憶領域を要する．現時点では耐クローンモジュールが利用可能となっている訳ではないが，例えば，各個体に固有でランダムなパターンが埋め込まれた半導体，特殊繊維入りのプラスチック，特殊加工が施されたホログラムなどが候補と考えられる[11]．

松本・田中[12]では，実行ハードウェア確認タグ方式の利用環境について次の想定が置かれている．

- **想定 1（耐クローンモジュールの入出力の入手）**：攻撃者は，耐クローンモジュールの入出力の一部を入手する．ただし，攻撃者が得る入出力集合のサイズは耐クローンモジュールの入出力集合全体のサイズに比べて極めて小さく，ある入力 x がランダムに与えられたときに，攻撃者が得た入出力集合から x に対応する出力 y を求めることは困難である．
- **想定 2（耐クローンモジュールとハードウェア）**：耐クローンモジュールの機能を損なうことなく，モジュールをハードウェアから分離することは困難である．
- **想定 3（ハードウェアからのデータの漏洩）**：攻撃者はハードウェア内部の秘密情報を入手する．
- **想定 4（ハードウェアの持ち主）**：ハードウェアの正当な持ち主は不正な行為を行わない．
- **想定 5（通信データの傍受）**：攻撃者は署名や暗号文等の生成時に送受信されるデータを入手する．

実行ハードウェア確認タグ方式の目的は，上記の想定 1～5 が満たされる状況の下で，「署名や暗号文等が特定のハードウェアにおいて生成されたことを確認可能にする」ことと設定されている．

### 2.2 方式の概要[12]

#### 2.2.1 エンティティ

実行ハードウェア確認タグ方式は，管理者（A），ハードウェア保持者（H），タグ・署名付きデータの検証を A に依頼する検証者（V）の 3 つのエンティティから構成される．A は，暗号処理用ハードウェア（Z）を準備して H に提供するとともに，十分に大きな記憶容量を有するデータベース DB1，DB2 を安全に管理し，V の依頼によってタグを用いた検証を行う．

### 2.2.2 記号の定義

本稿で用いる記号は以下のとおり定義する．

- $ID_A$：管理者 A の識別子
- $ID_H$：ハードウェア所持者 H の識別子
- f：耐クローンモジュールの入出力関係を表す関数
- $x_i$：耐クローンモジュールの入力として用いられるデータの候補
- $y_i$ (=$f(x_i)$)：$x_i$ に対する耐クローンモジュールの出力
- $T_H$:ハードウェア所持者 H のハードウェアに格納されている耐クローンモジュールの入出力のテーブル（管理者が DB1 において保管）
- h：衝突困難な一方向性ハッシュ関数
- SD(M)：データ M に対して暗号処理を実行した結果のデータ（例えば，M に対する署名付きデータ）
- Tag(M)：M に対するタグ
- SDT(M)：M に対するタグ付きデータ

### 2.2.3 準備

管理者 A とハードウェア所持者 H は，以下の手順でハードウェアを準備する．

- **準備** 1：A は，H が使用するハードウェア Z に組み込む耐クローンモジュール F を選択する．
- **準備** 2：A は，F の入力の集合からランダムに E 個の要素を選択し，それらを $x_i$ (i=1, 2, ..., E) とする．A は，$x_i$ (i=1, 2, ..., E) を F に入力して出力 $f(x_i)$を求め，テーブル $T_H$ を作成する．A は($ID_H$, $T_H$)を DB1 に記録し，秘密に管理する．
- **準備** 3：A は，F を Z に組み込み，H に Z を渡す．

### 2.2.4 計算実行，タグ生成

データ M に対して暗号処理を実行するとともに，タグ Tag(M)を生成する手順は以下のとおりである（図 1 参照）．

- **生成処理** 1：ハードウェア所持者 H は，自分の識別子 $ID_H$ を管理者 A に送り，耐クローンモジュール F の入力 q=($q_1$, $q_2$, ..., $q_n$)の送付を要求する．
- **生成処理** 2：A は，テーブル $T_H$ から $q_i$ (i=1, 2, ..., n)を選択して H に送る．A は，送信情報[$ID_H$, q, $t_1$]をデータベース DB2 に記録する．A が選択する q は毎回異なる．$t_1$ は，管理者が q を送った日時データである．
- **生成処理** 3：H は，q= ($q_1$, $q_2$, ..., $q_n$)を F に入力して出力 $a_i$=$f(q_i)$ (i=1, 2, ..., n)を求めるほか，暗号処理結果のデータ SD(M)を得る．次に，H はタグ Tag(M)を次のように生成する．

$$Tag(M)=h(a_1\|a_2\| \ldots \|a_n\|SD(M)\|t_2)$$

- **生成処理** 4：H は，タグ付きデータ SDT(M)を次のように生成する．ただし，$t_2$ はハードウェア所持者が Tag(M)を生成した日時データである．

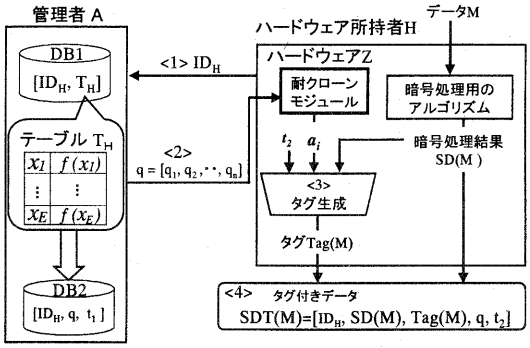


**図1 実行ハードウェア確認タグ方式における タグの生成手順**

$$SDT(M) = [ID_H, SD(M), Tag(M), q, t_2]$$

### 2.2.5 タグ検証

検証者 V からの依頼により，管理者 A は SDT(M)がハードウェア Z から生成されたか否かを次の手順で検証する．

- **検証処理** 1：V は，A に SDT(M)=[$ID_H$, SD(M), Tag(M), q, $t_2$] を渡す．
- **検証処理** 2：A は，DB2 から H に対応した q と SDT(M)内の q を比較する．一致しなければ SDT(M)が不正なものと判定する．
- **検証処理** 3：A は，DB1 の情報を利用して $a'_i$=$f(q_i)$ (i=1, 2, ..., n)を求め，Tag(M)を生成する．すなわち，A は以下の計算を行う．

$$Tag'(M)=h(a'_1\|a'_2\|\ldots\|a'_n\|SD(M)\|t_2)$$

- **検証処理** 4：A は，Tag'(M)と SDT(M)内の Tag(M)を比較する．一致しなければ SDT(M)が不正なものと判定し，一致すれば，A は，SDT(M)が Z において生成されたと判定する．

## 2.3 実行ハードウェア確認タグ方式に対する攻撃

松本・田中[12]は，実行ハードウェア確認タグ方式に対する攻撃として，ハードウェアの複製とタグの偽造を想定している．松本・田中[13]は，これらの攻撃に対する安全性に関して考察を行い，実行ハードウェア確認タグ方式がこれらの攻撃に対して安全であることを確認している．

# 3. タグ付き署名方式

## 3.1 想定環境・攻撃・目的

実行ハードウェア確認タグ方式をディジタル署名に利用する場合に想定される環境や攻撃について検討する．タグ付き署名方式では，耐クローンモジュールが組み込まれたハードウェアに署名生成用秘密鍵が格納され，データ M に対するディジタル署名およびタグがその内部で生成される．

このタグ付き署名方式の利用環境に関する想定として，

2.1 節の想定 1〜5 を置く．ここで，想定する攻撃形態は以下のとおりとする．

・**署名やタグの偽造**：攻撃者（ハードウェアの正当な持ち主以外のエンティティ）が，管理者によるタグを用いた検証において検知されないように，署名やタグを偽造する．

以上を踏まえて，タグ付き署名方式の目的を次のように定める．

・**タグ付き署名方式の目的**：想定 1〜5 が満たされる状況の下で，署名が生成されたハードウェアを確認可能にすることによって，署名やタグの偽造を検知可能にする．

### 3.2 タグ付き署名方式のセキュリティ要件

3.1 で設定した攻撃に対して安全性を確保するためのセキュリティ要件について考察する．

秘密鍵が漏洩すると，署名自体の検証によって署名の偽造を検知することが不可能となることは明らかである．したがって，タグ付き署名方式では，タグを用いた検証によって署名やタグの偽造を検知する必要がある．

ハードウェア所持者は当然自分のハードウェアを厳重に管理すると考えられる．こうした状況の下で，攻撃者がタグ付きの署名を偽造する方法として，次の 2 つが考えられる．第一に，ハードウェア所持者によるハードウェアの管理が適切に行われており，攻撃者は当該ハードウェアを用いることなく想定 1〜5 の下でタグ付きの署名を偽造する，というタイプが考えられる．第二に，ハードウェアの厳重な管理にもかかわらず，攻撃者が何らかの方法で当該ハードウェアを盗用し，想定 1〜5 の下でタグ付きの署名を偽造する，というタイプが考えられる．

まず，攻撃者が，第一のタイプの攻撃を実行する場合を考える．こうした攻撃に関して，松本・田中[13]では，攻撃者が，耐クローンモジュールの入出力の一部を得ることができるものの，実際にタグ付きデータの生成に利用された耐クローンモジュールの入出力を得ることはできないとの想定の下で安全性の評価を行っている．これに対して，本稿では，攻撃者にとってより有利な状況を想定した上でセキュリティ要件を設定することとする．すなわち，攻撃者が，管理者とハードウェア所持者との間で交信されるデータを盗聴し，管理者からハードウェア所持者に送られる耐クローンモジュールの入力を入手するとともに，その入力に対応する耐クローンモジュールの出力も入手できた，という場合を想定する．

このような場合においてもタグ付きの署名の偽造に対応するためには，以下の要件を満たすことが求められる．

・**要件 1**：攻撃者が実際にタグ付きの署名生成に用いられた耐クローンモジュールの入出力を入手できた状況において，管理者から配布されたハードウェアを使用しないでタグ付きの署名が偽造されたならば，管理者はそのタグを用いた検証において異常を検知する．

次に，攻撃者が，第二のタイプの攻撃を実行し，正当なハードウェアを使用して既定の手続に沿ってタグ付きの署名を偽造した場合，管理者は，タグを用いた検証において偽造の事実を検知することは不可能である．こうした事態を回避するために，タグ付き署名方式には次の要件を満たすことが求められる．

・**要件 2**：管理者から配布されたハードウェアを使用してタグ付きの署名を生成する場合，タグ付きの署名の生成者がそのハードウェアの正当な所持者ではないならば，タグ付き署名方式における既定の手続に沿ってタグ付きの署名を偽造することは困難である．

タグ付きの署名の偽造に対して安全なタグ付き署名方式を実現するためには，以上の 2 つの要件を満足するように，タグの生成方法や検証方法等を設計することが必要である．ただし，これらの要件を満足する手段としては様々なものが想定され，どのような手段を採用すべきかは，タグ付き署名方式が適用されるアプリケーションの要件等に依存すると考えられる．

## 4. 既存技術との比較

秘密鍵が漏洩した場合の代表的な対応策であるヒステリシス署名とフォワードセキュア署名方式を取り上げ，タグ付き署名方式との比較を行う．

### 4.1 ヒステリシス署名

ヒステリシス署名は，署名生成の履歴を署名生成者でさえも偽造困難な形態で保管し，「秘密鍵が漏洩しても，署名生成履歴を用いた検証によって，過去に生成されたとみなされる署名の偽造を検出する」ことを目的とする署名方式である[9, 10]．

ヒステリシス署名は，署名生成の際に，既存の署名等を署名生成記録として署名に埋め込むという手法（チェイニング署名，chaining signature）を採用する[10]．署名生成履歴はすべての署名の署名生成記録によって構成される．したがって，過去に生成された署名を偽造するためにはそれ以降に生成されたすべての署名も署名生成履歴と整合的に偽造する必要があるため，署名の偽造が困難になると考えられる．また，過去の署名の偽造を一層困難にする手段として，異なる利用者の署名生成履歴を互いに綴り合わせる「履歴交差」と呼ばれる方法も提案されている[10]．

ヒステリシス署名を利用する場合の主な想定は以下のとおりである[9, 10]．

・**署名生成履歴の管理**：署名生成履歴は欠損のない完全な状態で保管され，それを第三者が確認可能である．
・**署名生成の形態**：正当な秘密鍵の使用者でも，署名生成履歴に記録されないような手段を用いてヒステリシス署名を適正に生成することは不可能である．

ヒステリシス署名の一実現形態として，IC カード等の耐

表　秘密鍵が漏洩した場合の主な対応策の比較

| | タグ付き署名方式 | ヒステリシス署名 | フォワードセキュア署名方式 |
|---|---|---|---|
| **実現方法** | 耐クローンモジュールの出力や署名等から生成されたデータ（EV に対応）を管理者が安全に保管し，そのデータを使って実行ハードウェアを確認することによって検証する． | 署名生成履歴を安全に管理し，検証対象の署名の情報が署名生成履歴に存在するかを検証する． | 更新後の秘密鍵から更新前の秘密鍵を入手困難なように秘密鍵を更新する． |
| **効果** | 署名の偽造検知 | 署名の偽造検知 | 署名の偽造検知 |
| **秘密鍵の漏洩形態** | あらゆるタイプの秘密鍵の漏洩が想定されている． | あらゆるタイプの秘密鍵の漏洩が想定されている． | 署名方式の欠陥による漏洩は想定されていない． |
| **主な想定** | ・耐クローンモジュールを利用可能である．<br>・管理者は，データベースを安全に管理し，攻撃者との結託等の不正行為を行わない．<br>・ハードウェアの正当な持ち主は攻撃者との結託等不正行為を行わない． | ・署名生成履歴は完全な状態で保管され，それを第三者が確認可能である．<br>・秘密鍵の正当な持ち主であっても署名生成履歴に記録されない形態で署名を生成することはできない． | ・更新後の古い秘密鍵は廃棄され，攻撃者が入手できない．<br>・秘密鍵の正当な持ち主は，秘密鍵が漏洩した場合に直ちにその事実を検知する． |
| **署名生成時のデータ交信** | 管理者と署名生成者との間でデータの更新が必要． | 基本的には不要．履歴交差の際に適宜データ交信が必要． | 不要． |

タンパーハードウェア内部で署名や署名生成履歴の生成・管理を行うという形態が想定されている[9].

### 4.2　フォワードセキュア署名方式

フォワードセキュア署名方式は，秘密鍵を短い期間（例えば1日）で更新し，使用期間が満了した秘密鍵をその都度更新・廃棄することによって，「現在使用している秘密鍵が漏洩しても，その秘密鍵の使用が開始される前の時点で生成されたとみられる署名の偽造を検出可能にする」ことを目的とする署名方式である[2, 4].

これまでに様々なフォワードセキュア署名方式の実現方法[1, 4, 5]が提案されているが，いずれも秘密鍵は一方向性関数を用いて更新され，最新の秘密鍵を入手してもそれ以前の秘密鍵を入手困難にするという仕組みを採用している[1, 4, 5]. フォワードセキュア署名方式の主な想定環境は以下のとおりである[4].

- **秘密鍵の漏洩形態**：秘密鍵は，ハードウェア等から破壊・非破壊解析によって外部に漏洩する．署名アルゴリズムの実装方法にはセキュリティ上の欠陥がなく，公開鍵等の公開情報から秘密鍵が漏洩することはない．
- **秘密鍵の廃棄**：使用期間が満了した秘密鍵は直ちに廃棄され，廃棄後の古い秘密鍵を第三者が入手することは困難である．

フォワードセキュア署名方式の有効性に関する研究として，高橋・洲崎・松本[7]の研究成果が注目される．高橋らは，フォワードセキュア署名方式の提案論文において「秘密鍵の正当な持ち主が秘密鍵の漏洩を直ちに検知できるか否かが明確にされていない」点を指摘し，秘密鍵の漏洩が直ちに検知されない場合には，どの時点より前の署名が引き続き信頼できるかを識別不可能になり，フォワードセキュア署名方式の有効性が失われる可能性を示唆している．すなわち，フォワードセキュア署名方式を実装する場合には，秘密鍵の漏洩を直ちに検知するための対策が必要であるといえる．

### 4.3　3つの方式の比較（表参照）

秘密鍵が漏洩した場合の対策の方法は3つの方式において異なっているものの，最終的に期待される効果は，いずれの方式においても署名の偽造を検知することとなっている．

想定されている秘密鍵の漏洩形態をみると，タグ付き署名方式とヒステリシス署名では，秘密鍵漏洩形態に特段の制限がない．これに対してフォワードセキュア署名方式は，署名方式自体の欠陥による秘密鍵の漏洩は想定されていない．このため，タグ付き署名方式とヒステリシス署名は，対応可能な秘密鍵漏洩形態がフォワードセキュア署名方式に比べて広いといえる．

各方式の実現に必要な想定に関しては，タグ付き署名方式では，耐クローンモジュールが利用可能であること，データベースの管理等を適切に実行する管理者が利用可能であること等が挙げられる．ヒステリシス署名では，署名生成履歴が欠損のない完全な状態で保管され，その事実を第三者が確認可能であることや，正当な署名生成者でさえも署名生成履歴に記録されない形態で署名を生成することができないようにしておくことが挙げられる．これらの方式では，形態は異なるが，一種の「信頼できる第三者（Trusted Third Party）」によってデータを安全に保管することが前提となっているといえる．一方，フォワードセキュア署名方式では，更新前の秘密鍵が確実に廃棄されることが必要となるほか，秘密鍵が漏洩した場合にはそれを直ちに検知することが必要である．しかし，信頼できる第三者を利用することは必ずしも必要となっていない．したがって，信頼できる第三者の利用可能性という観点では，フォワードセキュア署名方式は実現性に関するハードルが比較的低いといえる．

また，署名生成時における他のエンティティとのデータ交信については，タグ付き署名方式では耐クローンモジュールの入力の送付が必要となる．ヒステリシス署名や

フォワードセキュア署名方式では，基本的には他のエンティティとデータ交信を行わないで署名生成が行われる．

## 5. タグ付き署名方式の一実現形態

本節では，3 において検討したセキュリティ要件を満足するタグ付き署名方式の一実現形態を説明する．タグ付き署名方式には様々な形態のものが考えられるが，ここではその一例として，タグ付き署名方式の具体的なプロトコルを説明する．以下で特段説明していない基本的な枠組みは 2 で説明したものと同一である．

### 5.1 追加的な記号の定義

2.2.2 で定義した記号に追加して以下の記号を定義する．

- EV：タグ検証用データ（ハードウェア所持者が生成して管理者が保管）
- $SKE_A$, $PKE_A$：管理者 A の守秘用の秘密鍵と公開鍵
- $SKS_A$, $PKS_A$：管理者 A の署名用の秘密鍵と公開鍵
- $S_A(M)$：データ M に対する管理者 A の署名
- $Cert(PKS_A)$：管理者 A の署名生成用秘密鍵 $PKS_A$ に対する公開鍵証明書
- $PD_H$：ハードウェア Z の正当な所持者 H の本人確認用パスワード（サイズは各アプリケーションのセキュリティ要件によって設定されるとする）

以下で説明するプロトコルでは，耐クローンモジュールの入力として 2 種類のデータ $q=(q_1, q_2, \ldots, q_n)$，$q^*=(q^*_1, q^*_2, \ldots, q^*_n)$が管理者からハードウェア所持者に送られる．これらの入力に対応する耐クローンモジュールの出力をそれぞれ $a=(a_1, a_2, \ldots, a_n)$，$a^*=(a^*_1, a^*_2, \ldots, a^*_n)$とする．また，タグ Tag(M)とタグ付きデータ SDT(M)については，次のように生成されるものとする．

$$Tag(M)=[ID_H, ID_A, h(a^*_1 \| a^*_2 \| \ldots \| a^*_n), t_1]$$
$$SDT(M) = [SD(M), Tag(M)]$$

### 5.2 準備

2.2.3 の準備 1～3 に以下の準備 4，5 を追加する．

- **準備** 4：ハードウェア所持者 H はハードウェア内部で署名用の鍵ペア $SKS_H$，$PKS_H$ を生成する．
- **準備** 5：H は，署名生成者の本人確認用パスワード $PD_H$ を決定して A に秘密に通知する．A は，$PD_H$ を$(ID_H, T_H)$とともに DB1 に記録し，これらを秘密に管理する．

### 5.3 タグ・署名付きデータの生成

データ M に対するタグ・署名付きデータ SDT(M)は以下の手順で生成する（図 2 参照）．16 の生成処理のうち，生成処理 2～6 はハードウェアの確認を行う処理に対応し，生成処理 7～10 がハードウェアの持ち主の確認を行う処理に対応する．また，残りの生成処理 11～16 がタグ・署名付きデータの生成処理に対応する．

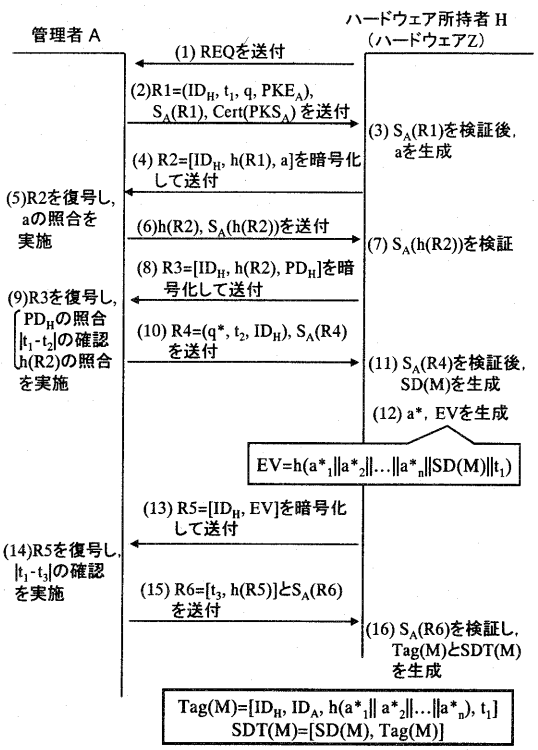


**図2 タグ・署名付きデータの生成手順**

- **生成処理** 1：ハードウェア所持者 H は，耐クローンモジュール F の入力データの送付を要求するデータ REQ（$ID_H$ 等から構成）を A に送る．
- **生成処理** 2：管理者 A は，REQ の受信日時データ $t_1$ を DB2 に記録し，$T_H$ から $q=(q_1, q_2, \ldots, q_n)$を選択して $R1=(ID_H, t_1, q, PKE_A)$，$S_A(R1)$，$Cert(PKS_A)$を生成し，H に送る．
- **生成処理** 3：H は $S_A(R1)$を検証し，q に対する F の出力 $a_i=f(q_i)$ (i=1, 2, …, n)を生成する．
- **生成処理** 4：H は，$R2= [ID_H, h(R1), a]$を $PKE_A$ で暗号化して A に送る．ただし，$a= (a_1, a_2, \ldots, a_n)$とする．
- **生成処理** 5：A は，R2 を復号し，a が q に対応していることを確認する．対応していない場合，A は生成処理を中断する．
- **生成処理** 6：A は，$PD_H$ の送付を要求するデータとして，h(R2)と $S_A(h(R2))$を送る．
- **生成処理** 7：H は $S_A(h(R2))$を検証する．
- **生成処理** 8：H は $R3= [ID_H, h(R2), PD_H]$を $PKE_A$ で暗号化して A に送る．
- **生成処理** 9：A は，R3 を復号し，
  <1>$PD_H$ が DB1 に記録されているものと一致すること，
  <2>R3 の受信日時データ $t_2$ と $t_1$ の時間差が一定値以内

であること，

<3>R2 のハッシュ値と R3 に含まれる h(R2)が一致すること，

の 3 点を確認する．いずれかが成功しない場合，A は H に対して R3 を暗号化して再送するように通知する．一定回数以上連続して失敗すれば，生成処理を中断し，当該ハードウェア Z の使用を差し止める．

- **生成処理** 10：A は，$T_H$から選択した $q^*_i$ (i = 1, 2, ..., n)と $t_2$ を DB2 に記録した上で，R4=($q^*$, $t_2$, $ID_H$)と $S_A$(R4)を H に送る．ただし，$q^*= (q^*_1, q^*_2, ..., q^*_n)$である．
- **生成処理** 11：H は，$S_A$(R4)を検証後，SD(M)を生成する．
- **生成処理** 12：H は，$q^*= (q^*_1, q^*_2, ..., q^*_n)$に対する F の出力 $a^*_i=f(q^*_i)$ (i=1, 2, ..., n)を求め，タグ検証用データ EV を，$EV=h(a^*_1\|a^*_2\| ...\|a^*_n\|SD(M)\|t_1)$として生成する．
- **生成処理** 13：H は，R5=[$ID_H$, h(R4), EV]を $PKE_A$ で暗号化して A に送る．
- **生成処理** 14：A は，R5 を復号し，

<1>R5 の受信日時データ $t_3$ と $t_1$ の時間差が一定値以内であること，

<2>R4 のハッシュ値と R5 に含まれる h(R4)が一致すること，

の 2 点を確認する．いずれかが成功しない場合，A は R5 を暗号化したデータを再送するよう要求する．一定回数以上連続して成功しない場合には，生成処理を中断し，Z の使用を差し止める．

- **生成処理** 15：A は，EV と $t_3$ を DB2 に記録し，R6=[$t_3$, h(R4)] と $S_A$(R6)を H に送る．
- **生成処理** 16：H は，$S_A$(R6)を検証し，タグ Tag(M)とタグ付きデータ SDT(M)を以下のとおり生成する．

$$Tag(M) = [ID_H, ID_A, h(a^*_1\|a^*_2\| ...\|a^*_n), t_1]$$
$$SDT(M) = [SD(M), Tag(M)]$$

以上の結果，A は DB2 に($ID_H$, $t_1$, $q^*$, EV)を保管する．

## 5.4 タグの検証

SDT(M)の検証は次の手順で実行される．

- **検証処理** 1：管理者 A は，Tag(M)から($ID_H$, $t_1$)を取り出し，DB2 から($ID_H$, $t_1$)に対応する $q^*$を検索する．該当する $q^*$ が存在しない場合，Tag(M)が偽造されたと判定する．
- **検証処理** 2：A は，$a^{*\prime}_i=f(q^*_i)$ (i=1, 2, ..., n)を求めた後，$h(a^{*\prime}_1\|a^{*\prime}_2\| ...\|a^{*\prime}_n)$ を計算し，Tag(M)内の $h(a^*_1\|a^*_2\| ...\|a^*_n)$と照合する．一致しない場合，SDT(M)が正しいハードウェア Z で生成されていないと判定する．
- **検証処理** 3：A は，SDT(M)内の SD(M)を用いて EV'= $h(a^{*\prime}_1\|a^{*\prime}_2\| ...\|a^{*\prime}_n\|$ SD(M) $\|$ $t_1$)を生成し，DB2 に記録されている EV と照合する．一致しない場合，署名やタグが偽造されたと判定する．

3 つの検証処理が成功した場合，A は「SDT(M)が正しいハードウェアにおいてその正当な所持者によって生成されたものである」と判定し，検証者に通知する．

## 5.5 ハードウェア盗難時およびパスワード漏洩時の対応

ハードウェア所有者 H は，自分のハードウェア Z およびパスワード $PD_H$ を厳重に管理し，第三者が入手することのないように努める．ハードウェア所有者が，ハードウェアの盗難やパスワード漏洩に気づいた場合には，直ちに管理者に通知する．これらの通知を受けた管理者は直ちにハードウェアの使用を差し止める．

## 5.6 安全性に関する考察

### 5.6.1 タグ付き署名の偽造検知（要件 1）

まず，要件 1 が満足されていることを確認する．攻撃者は，ハードウェア Z の正当な所持者 H の秘密鍵を入手した上で，自分で選択したデータ M'に対するタグ・署名付きデータ SDT(M') (=[SD(M'), Tag(M')]) を偽造する．この場合，管理者がタグ付きの署名の偽造を検知することを示す．

**(1) 検証処理 1，2**

攻撃者は，ハードウェア Z に格納されていた H の秘密鍵を入手しているので，署名検証において検知不可能な形態で SD(M')を偽造することができる．したがって，攻撃者は，タグを用いた検証の際に異常が検知されないように SD(M')や Tag(M')を偽造することを試みる．

攻撃者は，まず，Tag(M')を偽造する際に，検証処理 1 が成功するように REQ の受信日時データ $t_1$ を決める必要がある．そのためには，攻撃者は実際に交信された R1 内の $t_1$ を利用する必要があるが，「攻撃者は A と H の交信データを盗聴する」（想定 5）と想定していることから，盗聴によって R1 内の $t_1$ や R4 内の $q^*$を入手し，それを攻撃に利用することができる．

以下では，正規の手続で既に生成されたタグ付きの署名に関するデータで，攻撃に利用されるものの記号をイタリックで表記する．まず，攻撃者が盗聴によって入手する R1 内の日時データを $t_1$，R4 内の耐クローンモジュールへの入力を $q^*$とする．$t_1$ や $q^*$によって生成されたタグ付き署名に対応する署名対象データを $M$ とし，タグ検証用データを $EV$ (=$h(a^*_1\|a^*_2\|...\|a^*_n\|SD(M)\|t_1)$) とする．

タグ付き署名方式では，「攻撃者は少数の入力に対する耐クローンモジュールの出力を入手する」（想定 1）と想定していることから，攻撃者は $q^*$に対する耐クローンモジュールの出力 $a^*_i = f(q^*_i)$ (i=1, 2, ..., n)を入手することができる．そこで，攻撃者は，この $a^*$を用いて Tag(M') = [$ID_H$, $ID_A$, $h(a^*_1\|a^*_2\|...\|a^*_n), t_1$]を偽造する．

この結果，検証処理 1 に関しては，$t_1$ に対応する $q^*$が DB2 に保管されていることから，検証処理 1 において偽造を検知することはできない．

また，検証処理 2 に関しても，攻撃者が Tag(M')の偽造の際に用いた $a^*$が $q^*$に対応していることから，$h(a^*_1\|a^*_2\| ...\|a^*_n)$の照合も成功する．したがって検証処理 2 においても偽造を検知することはできない．

(2) 検証処理 3

タグ付きの署名の偽造を検証処理 1，2 において検知することはできないものの，以下で示すように検証処理 3 において検知可能である．

管理者 A は，DB2 を検索して $t_I$ に対応する $EV$ を入手した後，検証者から得た SD(M')を用いてタグ検証用データ $EV'=h(a^*_1\|a^*_2\|\ldots\|a^*_n\|SD(M')\|t_I)$ を生成し，EV' と $EV = h(a^*_1\|a^*_2\|\ldots\|a^*_n\|SD(M)\|t_I)$ を照合する．

こうした検証において異常を検知されないようにするためには，攻撃者は，予め EV' と $EV$ が等しくなるように（SD($M$)とは異なる）SD(M')を偽造しておく必要がある．しかし，h は衝突困難なハッシュ関数であるため，そうした条件を満たす M'を選択することは困難である．

### 5.5.2 ハードウェアの盗用に対する安全性（要件 2）

要件 2 が満足されていることを確認する．攻撃者は，ハードウェア Z の正当な所持者 H から Z を盗用し，H になりすまして既定の手続に沿ってタグ付きの署名の偽造すると想定する．

攻撃者が既定の手続に沿ってタグ付きの署名を偽造するためには，生成処理 8 において H のパスワード $PD_H$ を H や h(R2)とともに暗号化して管理者 A に送る必要がある．したがって，攻撃の正否は攻撃者が $PD_H$ を入手可能か否かに依存することとなる．しかし，ここでは，ハードウェア所有者が $PD_H$ を厳格に管理していることを想定しているため，攻撃者は正しい $PD_H$ を管理者 A に送ることができず，A は生成処理 9 における $PD_H$ の照合によって異常を検知することができる．また，万一 Z が盗用され，かつ，$PD_H$ が攻撃者に漏洩したとしても，直ちにハードウェア Z の使用が差し止められるならば，攻撃を実行することはできなくなると考えられる．

## 6. おわりに

本稿では，松本・田中[12]が提案した実行ハードウェア確認タグ方式をディジタル署名に応用した「実行ハードウェア確認タグ付きディジタル署名方式」について検討を行った．まず，実行ハードウェア確認タグ方式を署名に適用する場合の想定環境を整理し，セキュリティ要件について検討した．また，ヒステリシス署名とフォワードセキュア署名方式との比較を行ったほか，タグ付き署名方式の一実現形態を示し，その安全性について考察を行った．

今後は，より詳細な安全性評価を行うほか，署名生成・検証時に必要となるコストに関して他の既存技術と比較を行う方針である．また，実際に耐クローンモジュールとしてどのようなものが利用できるかについても検討を行う方針である．

**謝辞** 本研究は，一部分，文部科学省科学研究費補助金・特定領域研究 13224040（松本勉）の支援を受けて行われた．

## 参 考 文 献

[1]Abdalla, M., and L. Reyzin, "A New Forward-Secure Digital Signature Scheme," Proceedings of ASIACRYPT 2000, LNCS 1976, pp.116-129, Springer-Verlag, 2000.

[2]Anderson, R., "Two Remarks on Public-Key Cryptology, " Manuscript, 2000.

[3]Anderson, R., Security Engineering – A Guide to Building Dependable Distributed Systems, Wiley Computer Publishing, John Wiley & Sons, Inc., 2001.

[4]Bellare, M., and S. K. Miner, "A Forward-Secure Digital Signature Scheme," Proceedings of CRYPTO '99, LNCS 1666, pp.431-448, Springer-Verlag, August 1999.

[5]Itkis, G., and L. Reyzin, "Forward-Secure Signatures with Optimal Signing and Verifying," Proceedings of CRYPTO 2001, LNCS 2139, pp.332-354, Springer-Verlag, 2001.

[6] 小森旭・松浦幹太・須藤修，「PKI に基づく C/S 型アプリケーションの安全性分析と証拠性評価」，コンピュータセキュリティシンポジウム 2001 論文集，情報処理学会シンポジウムシリーズ Vol.2001，No.15，pp.319-324，情報処理学会，2001 年 10 月

[7]高橋知史・洲崎誠一・松本勉，「Forward-Secure Digital Signature は役に立つか」，2002 年暗号と情報セキュリティシンポジウム予稿集，pp.837-842，電子情報通信学会，2002 年 1 月

[8]洲崎誠一・松本勉，「電子署名の偽造に関する一考察」，コンピュータセキュリティシンポジウム 2001 論文集，pp.211-216，情報処理学会，2001 年 10 月

[9]洲崎誠一・宮崎邦彦・宝木和夫・松本勉，「暗号ブレイク対応電子署名アリバイ実現機構（その 2）—詳細方式—」，情報処理学会研究報告 2000-CSEC-8，pp.18-23，情報処理学会，2000 年 3 月

[10]松本勉・岩村充・佐々木良一・松木武，「暗号ブレイク対応電子署名アリバイ実現機構（その 1）—コンセプトと概要—」，情報処理学会研究報告 2000-CSEC-8，pp.13-17，情報処理学会，2000 年 3 月

[11]松本勉・久保田浩美・井上拓也・鴨志田昭輝・林修一・井上信吾・清水健介，「耐クローン性に基づく認証方式の提案」，1997 年暗号と情報セキュリティシンポジウム予稿集，SCIS'97-19C，電子情報通信学会，1997 年 1 月

[12]松本勉・田中直樹，「計算の実行ハードウェアを確認する方法」，コンピュータセキュリティシンポジウム 2000 論文集，pp.199-204，情報処理学会，2000 年 10 月

[13]松本勉・田中直樹，「計算実行ハードウェアの物理的仮定に基づく認証」，2001 年暗号と情報セキュリティシンポジウム予稿集，pp.613-618，電子情報通信学会，2001 年 1 月