Vol.2014-DPS-158 No.12
Vol.2014-CSEC-64 No.12
2014/3/6

ヒステリシス署名における高速検証方式の提案

渡辺夏樹[†1]　安細康介[†1]　宮崎邦彦[†1]　熊谷洋子[†1]

近年，電子契約や電子入札など電子署名の利用範囲が拡大し，使用している公開鍵暗号の危殆化対策が課題となるほど長期的に電子データを保管するケースが増えたことから，ヒステリシス署名の利用が提言されている．

ヒステリシス署名では，過去の署名データを含む連鎖構造による署名データを生成し，この連鎖構造を辿ることで過去の電子データの真正性を検証する．連鎖構造が長くなるのに従い検証処理量が大きくなるため，現状ではバッチ処理による検証処理を行っているが，本来は外部機関による監査や訴訟などに対応してオンデマンドに検証処理を実施する事が望まれている．

本研究では，オンデマンドでの検証処理の実現に向け，連続的に連鎖構造を検証していた従来方式を改良し，間欠的に検証可能とすることで，検証を高速化する方式を提案する．

NATSUKI WATANABE[†1]　KOSUKE ANZAI[†1]
KUNIHIKO MIYAZAKI[†1]　YOKO KUMAGAI[†1]

1. はじめに

近年，金融・産業などをはじめとする様々な分野や，各省庁・地方自治体などの公共分野などにおいて，電子申請や電子契約の利用が拡大してきている．

紙ベースの申請や契約では，直筆の署名や押印等を行うことで作成者本人により作成されたものであること（真正の成立）を証明していたのに対し，電子申請や電子契約では，電子署名法を根拠として申請や契約に関わる電子データの作成時に，電子署名を施すことで上記真正の成立を証明することとしている[1].

一般に，電子申請や電子契約に関わる電子データは，電子化される以前と同等程度の期間，改ざんされることなく安全に保存しておく必要があるものと考えられる. 例えば，国税関係書類は，税法上 7 年，商法上 10 年の保存義務が定められており，商取引等の契約書類も刑法第 246 条の 2（電子計算機使用詐欺）等の公訴時効が刑事訴訟法第 250 条より 7 年と規定されていることから，少なくとも時効までの 7 年間，真正性が保証できる状態で保存することが求められる．このため，電子化された国税関係データや契約書類データ等に関しても同等の保存義務が生じると考えられる.

また，上記に挙げたような電子申請や電子契約に関わる各々の業務が不正なく適切に行われていることを示すため，監査証跡として業務実施内容のログデータを保管し，インシデント発生時の不正検査や定期的な監査を行うため，業務の正当性を示す証拠として示す必要性もあると考えられる．証拠として示すためには，業務実施内容のログデータの真正性をその都度検証可能であることも要求される.

上記のように，法律上の要請や業務実施内容の正当性提示の観点から，重要な電子データに対して長期間の真正性を担保しつつ, 安全に保管するニーズが高まってきている.

電子データの改ざんを防止し真正性を保証する技術として電子署名が考えられるが，電子データの真正性を長期的に保つためには，暗号の危殆化を考慮した電子証明書の有効期限切れに対応する必要があり，この課題を解決する技術としてヒステリシス署名[2][3]の利用が提言されている.

ヒステリシス署名は，電子署名を生成する際に一つ前に生成した電子署名のハッシュ値を計算して署名対象に追加することで，前後の電子署名間に関係を持たせる電子署名方式である．これにより，電子証明書の有効期限の切れた電子署名の検証においても，有効期限内の電子証明書で検証可能な電子署名から順々に電子署名の関係を確認していくことで，電子データの真正性を検証することが可能となる.

一方，電子契約や電子申請に伴うデータや業務実施内容のログデータ等の電子データの保管期間に比例して，データ数が膨大になると，従来のヒステリシス署名では，過去の電子データの真正性を検証するために膨大な量の電子署名の関係の確認を行う必要がある．このため，真正性の検証に相当の処理時間がかかり，現状ではバッチ処理等で処理を行う事が想定されている．しかし，本来は訴訟や第三者による監査等のタイミングにおいて，オンデマンドに真正性の検証を実施可能とすることが望まれる.

本研究では，ヒステリシス署名におけるオンデマンドでの真正性検証に耐えうる処理性能を実現することを目的に，電子署名間の連鎖構造を複数持つことで，従来連続した前後の関係しか確認できなかった方式を改良し，間欠的に関係を確認可能とすることで，真正性の検証を高速化する方式を提案する.

以降，第二章では，長期的な真正性の保証方法について要件を整理した上で従来のヒステリシス署名の課題について述べ，第三章では，本課題を解決する電子署名の真正性を高速に検証する新たなヒステリシス署名の提案方式について述べる. 第四章では, 提案方式の有効性について述べ，

†1 (株)日立製作所
Hitachi Ltd.

Vol.2014-DPS-158 No.12
Vol.2014-CSEC-64 No.12
2014/3/6

最後に第五章でまとめと今後の課題について述べる．

2. 長期的な真正性保証方法

本章では，長期的な真正性の保証が必要となる電子データの前提条件およびその電子データを保管し真正性を検証する上での要件と，本要件を満たすために従来のヒステリシス署名を適用した場合の課題について述べる．

2.1 電子データの前提条件と保管に関する要件

長期的に真正性の保証が必要となる電子データの前提条件として，以下を想定するものとする．

- （前提 1）電子データの保管期間は，電子計算機使用詐欺の公訴時効等の法律上の要請を考慮して 7 年間とする．
- （前提 2）対象とする電子データは，業務システム等の業務ログデータ等とし，連続した電子データが時系列で大量にあるものとする．
- （前提 3）第三者によるログデータの監査が行われる際に，オンデマンドで真正性の検証を行うこととする．

上記前提を置いた上で，実用的な真正性保証を行うための要件として，以下を満たすことが求められる．

- （要件 1）上記の長期に渡る保管（7 年間）の間，第三者に改ざんがないことを客観的に証明できること．
- （要件 2）電子署名生成時の処理量，処理時間が小さく，実行上問題がないこと．
- （要件 3）電子署名の付与によるデータ量の増加が，実行上問題がないこと．
- （要件 4）電子署名検証時の処理量，処理時間が小さく，オンデマンドでの真正性検証に支障が出ないこと．

要件 1 は，公訴時効までの 7 年間，第三者に対し対象となる電子データの改ざんがないことを客観的に証明することで，監査証跡や証拠として有効性を示すための要件である．

要件 2 は，対象とする電子データが，連続して大量に発生した際でも問題なく処理が可能であるための要件である．

要件 3 は，対象とする電子データに対し，真正性保証に必要となるデータ量が膨大（例えば対象データと同等のデータ量が必要）となってしまうと，対象データの副本をセキュアな環境に別途保管する対策に対しメリットが少なくなるため，データ量の増加はある程度に抑えることが望ましいことから決めた要件である．

要件 4 は，前提とおいた第三者によるオンデマンドの真正性検証のたびに検証処理時間が数分程度かかってしまっては，定期的な監査業務等に支障が出てしまうために必要となる要件である．

以下では，従来のヒステリシス署名を前提とする条件下で適用した際の課題について述べる．

2.2 従来のヒステリシス署名の適用と課題

2.2.1 ヒステリシス署名の概要

ヒステリシス署名とは，直前に生成された電子署名のハシュ値を取り込み，電子署名間に連鎖構造を持たせる署名である．これにより，有効期限が切れた電子証明書に対応する電子署名であっても，有効期限内の電子証明書により真正性が保証された電子署名から連鎖構造を辿ることで，結果として真正性を証明可能となる．

図 1 にヒステリシス署名の署名付与のイメージを，図 2 に署名検証のイメージを示す．

ヒステリシス署名を付与する際には，直前に生成された署名の情報が改ざんされていないことを確認したのち，ハッシュ化して付与する署名に埋め込むことで，連鎖構造を作成する．具体的に図 1 の電子データ LogFile15 を例に署名生成の手順を述べる．

（1-1）電子データ LogFile15 のハッシュ値 H(LogFile15)を計算
（1-2）有効期限内の電子証明書 Cert をもとに電子署名 Sig14 の検証を行い，検証した Sig14 を用いて H(LogFile14)と H(Hys13)の真正性を検証
（1-3）直前に生成された署名生成記録 Hys14 のハッシュ値 H(Hys14)を計算
（1-4）上記で計算した H(LogFile15)と H(Hys14)に対して電子署名 Sig15 を生成

上記の署名生成手順を順次電子データに実施していくことで，署名の連鎖構造を構築することができる．

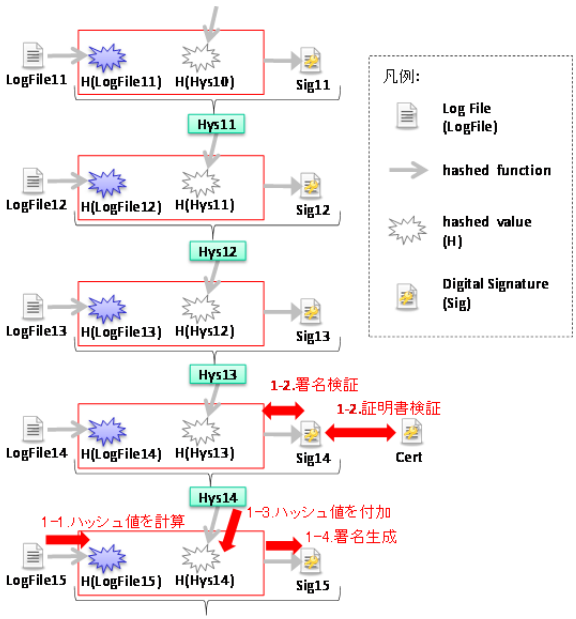

図 1 従来のヒステリシス署名方式の署名生成

上記のように生成されたヒステリシス署名を検証する際には，署名生成時に構築した各電子署名間の連鎖構造を順に辿ることにより，電子証明書の有効期限が切れた過去の

電子署名の真正性を検証することが可能となる．具体的に図 2 の電子データ LogFile11 を例に署名検証の手順を述べる．

（2-1）有効期限内の電子証明書 Cert をもとに電子署名 Sig15 の検証を行い H(LogFile15)と H(Hys14)の真正性を検証（電子署名 Sig11～Sig14 に対応する電子証明書の有効期限が切れている想定とする）

（2-2）署名生成記録 Hys14（ハッシュ値 H(LogFile14)，ハッシュ値 H(Hys13)，電子署名 Sig14）のハッシュ値を H(Hys14)と比較し，真正性を検証

（2-3～2-5）（2-2）と同様にハッシュ値を比較し，署名生成記録 Hys13,Hys12,Hys11 の真正性を検証

（2-6）LogFile11 から計算したハッシュ値を H(LogFile11)と比較した上で，真正性が検証された電子署名 Sig11 により H(LogFile11)と H(Hys10)を署名検証することで LogFile11 の真正性を検証

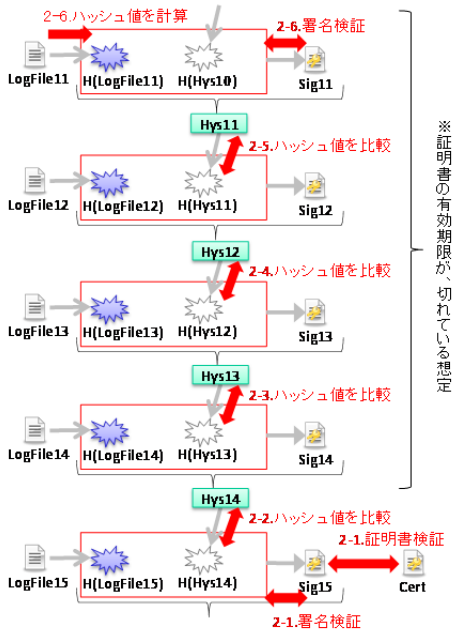

図 2 従来のヒステリシス署名方式の署名検証

上記のようにハッシュ値の比較を順次行うことで，ヒステリシス署名の連鎖構造を辿って電子証明書の有効期限が切れた電子署名であっても真正性を検証可能とし，検証対象の電子データの真正性を保証することが出来る．

2.2.2 ヒステリシス署名の課題

上記のように，署名付与，検証を行うため，ヒステリシス署名は電子証明書の有効期間が切れた電子署名に対しても，有効期間内の電子証明書に対応した署名から連鎖構造を順に辿ることで，真正性の検証が可能になる．すなわち，（要件 1）を満たすことが可能である．

（要件 2）の電子署名生成時の処理量，処理時間については，通常の電子署名と比較して，直前の電子署名の検証処理及び署名生成記録のハッシュ値計算処理が追加で行われることとなるが，署名対象となる電子データのサイズが 4.1 節で後述する前提条件（30Mbyte 程度）のようにある程度大きい場合には電子データのハッシュ値 H(LogFile)を生成する処理が全体処理のほとんどを占め，上記追加で行われる処理量，処理時間の増加は，実行上問題のある程度にはならないものと考えられる．

（要件 3）の電子署名の付与によるデータ量の増加については，通常の電子署名と比較し，各署名対象に一つ前の署名生成記録のハッシュ値を保持する分が追加される．署名対象とする電子データや電子署名のデータ量に対し，増加分のハッシュサイズは SHA256 を用いた場合でも 32 バイトのため，実行上問題のある程度にはならないものと考えられる．

（要件 4）の電子署名の検証時の処理量，処理時間については，通常の電子署名と比較し，連鎖構造を構成する署名数分のハッシュ値の比較が追加で行われるため，連鎖構造が膨大になった場合には相当の処理時間がかかる事となる．4.1 節で後述する前提条件のように，7 年間分の電子データを約 370 万件，ハッシュ値 1 回の比較に 1ms 要すると仮定すると，1 回の検証処理に約 1 時間（3,680s）要するため，検証をオンデマンドに実行することは困難であると考えられる．

上記により，従来のヒステリシス署名は，（要件 1）～（要件 3）は満たすものの，（要件 4）の真正性検証をオンデマンドで行うために必要となる検証の高速性が無いことが課題となる．

3. ヒステリシス署名の高速検証方式の提案

本章では，2.2.2 で示した従来のヒステリシスの課題を解決するヒステリシス署名の高速検証方式について提案する．

3.1 解決方針

提案する高速検証方式は，以下の 2 つの方針により，ヒステリシス署名による真正性検証を高速化する．

- （解決方針 1）署名対象に p(p≧2)個前の署名ハッシュ値を追加
- （解決方針 2）署名対象に（解決方針 1）の署名ハッシュ値を複数追加

上記方針により署名生成時に複数の連鎖構造を作成することで，検証時にはこの連鎖構造を間欠的に辿ることを可能にし，ハッシュ値を比較する回数を減少することで，検証時間を短縮させる．以下，解決方針それぞれを具体的に説明する．

(1) 署名対象に p(p≧2)個前の署名ハッシュ値を追加

従来のヒステリシス署名では 1 個前の署名生成記録のハ

Vol.2014-DPS-158 No.12
Vol.2014-CSEC-64 No.12
2014/3/6

ッシュ値のみを付与して電子署名をしていたのに対し，p(p≧2)個前の署名生成記録のハッシュ値を追加した上で電子署名を施す方針とする．具体的に図 3 の電子データ LogFile15 を例に p=2 の場合の署名生成の手順を述べる．

（3-1）電子データ LogFile15 のハッシュ値 H(LogFile15)を計算

（3-2）有効期限内の電子証明書 Cert をもとに電子署名 Sig14 の検証を行い H(LogFile14)と H(Hys13),H(Hys12)の真正性を検証

（3-3）直前に生成された署名生成記録 Hys14 のハッシュ値 H(Hys14)を計算

（3-4）上記で計算した H(LogFile15)と H(Hys14)，また，真正性検証済みの H(Hys13)に対して電子署名 Sig15 を生成

上記の署名生成手順を順次電子データに実施していくことで，1 個前の署名生成記録からなる連鎖構造と，p 個前の署名生成記録からなる連鎖構造を構築する事ができる．

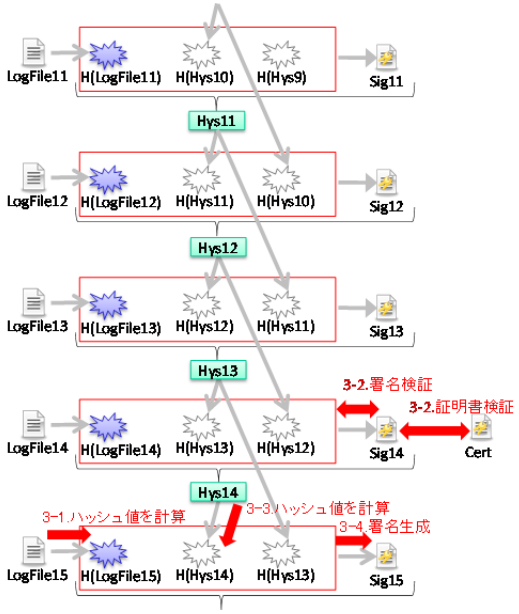

図 3 解決方針(1) 署名生成

上記のように生成されたヒステリシス署名の検証の際には，署名生成時に構築した連鎖構造を，従来 1 個ずつ辿っていたのに対し p 個ずつ辿ることが出来るため，ハッシュ値の比較回数を従来方式の 1/p 回にする事が可能となる．具体的に図 4 の電子データ LogFile11 を例に，署名検証の手順を述べる．

（4-1）有効期限内の電子証明書 Cert をもとに電子署名 Sig15 の検証を行い H(LogFile15)と H(Hys14)と H(Hys13)の真正性を検証（電子署名 Sig11～Sig14 に対応する電子証明書の有効期限が切れている想定とする）

（4-2）署名生成記録 Hys13（ハッシュ値 H(LogFile13)，ハッシュ値 H(Hys12)，ハッシュ値 H(Hys11)，電子署名 Sig13）のハッシュ値を H(Hys13)と比較し，真正性を検証

（4-3）（4-2）と同様にハッシュ値を比較し，署名生成記録 Hys11 の真正性を検証

（4-4）LogFile11 から計算したハッシュ値を H(LogFile11)と比較した上で，真正性が検証された電子署名 Sig11 により H(LogFile11)と H(Hys10)と H(Hys9）を署名検証することで LogFile11 の真正性を検証

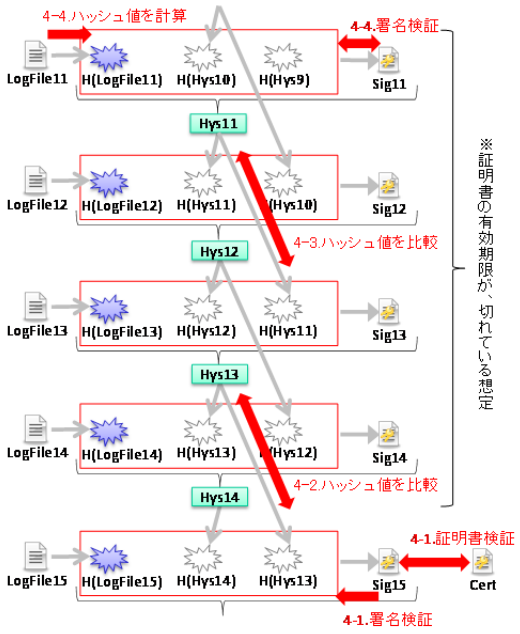

図 4 解決方針(1) 署名検証

上記のように，生成時に p 個前の署名生成記録を含めて複数の連鎖構造を構築し，検証時に p 個前連鎖構造を辿ることで検証回数を削減し，真正性検証を高速化する．

(2) 署名対象に複数の署名生成記録のハッシュ値を追加

従来，1 個前の電子署名のハッシュ値を 1 つのみ付与して電子署名をしていたのに対し，$2 \leqq p_1 < p_2 < p_3 < \cdots < p_n$ を満たす p_1 個前～p_n 個前の署名生成記録のハッシュ値を複数（n 個）追加した上で電子署名を施す方針とする．具体的に，図 5 の電子データ LogFile15 を例に $p_1=2$，$p_2=4$，n=2 の場合の署名生成の手順を述べる．

（5-1）電子データ LogFile15 のハッシュ値 H(LogFile15)を計算

（5-2）有効期限内の電子証明書 Cert をもとに電子署名 Sig14 の検証を行い H(LogFile14)と H(Hys13)，H(Hys12)，H(Hys10)の真正性を検証

（5-3）署名生成記録 Hys13 のハッシュ値と，上記で真正性を検証した署名生成記録 Hys13 のハッシュ値 H(Hys13)とを比較し，署名生成記録 Hys13 に含まれる H(Hys11)の真正性を検証

（5-4）直前に生成された電子データ LogFile14 のハッシュ

Vol.2014-DPS-158 No.12
Vol.2014-CSEC-64 No.12
2014/3/6

値 H(LogFile14), 電子データ LogFile13 の署名生成記録のハッシュ値 H(Hys13)，電子データ LogFile12 の署名生成記録のハッシュ値 H(Hys12)，電子データ LogFile10 の署名生成記録のハッシュ値 H(Hys10)に対して署名(Sig14)し，上記 4 つのハッシュ値および Sig14 の全体（署名生成記録 Hys14）に対するハッシュ値 H(Hys14)を計算

（5-5）上記で計算した H(LogFile15)と H(Hys14)，また，真正性検証済みの H(Hys13), H(hys11)に対して電子署名 Sig15 を生成

上記の署名生成手順を順次電子データに実施していくことで，1 個前の署名生成記録からなる連鎖構造と，$2 \leqq p_1 < p_2 < p_3 < \cdots < p_n$ を満たす p_1 個前～p_n 個前の n 個の署名生成記録からなる連鎖構造を構築する事ができる．

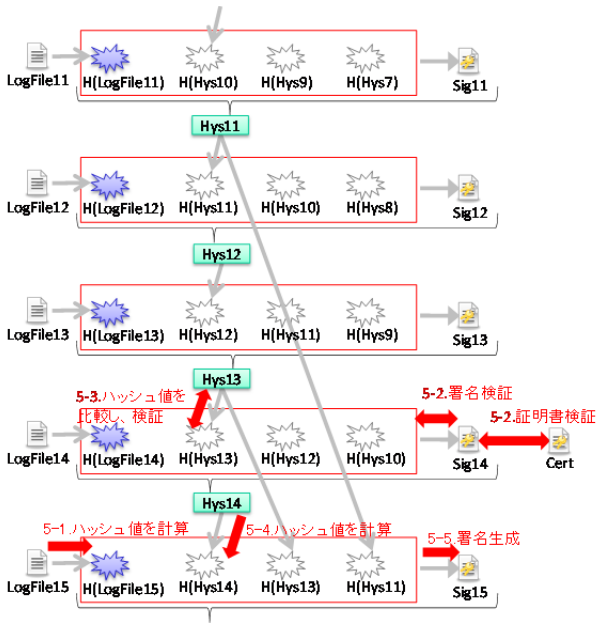

図 5 解決方針(2)　署名生成

上記のように生成されたヒステリシス署名の検証の際には，署名生成時に構築した複数の連鎖構造（p_1～p_n）のうち，検証対象の電子データまで辿ることが可能な最大の p_i を順次選択しながら署名生成履歴を辿ることで高速に検証可能となる．

具体的に図 6 の電子データ LogFile11 を例に，署名検証の絵手順を述べる．

（6-1）有効期限内の電子証明書 Cert をもとに電子署名 Sig15 の検証を行い H(LogFile15)と H(Hys14)と H(Hys13)と H(Hys11)の真正性を検証（電子署名 Sig11～Sig14 に対応する電子証明書の有効期限が切れている想定とする）

（6-2）署名生成記録 Hys11（ハッシュ値 H(LogFile11)，ハッシュ値 H(Hys10)，ハッシュ値 H(Hys9)，ハッシュ値 H(Hys7)，電子署名 Sig11）のハッシュ値を H(Hys11)と比較し，真正性を検証

（6-3）LogFile11 から計算したハッシュ値を H(LogFile11)と比較した上で，真正性が検証された電子署名 Sig11 により H(LogFile11)と H(Hys10）と H(Hys9）と H(Hys7）を署名検証することで LogFile11 の真正性を検証

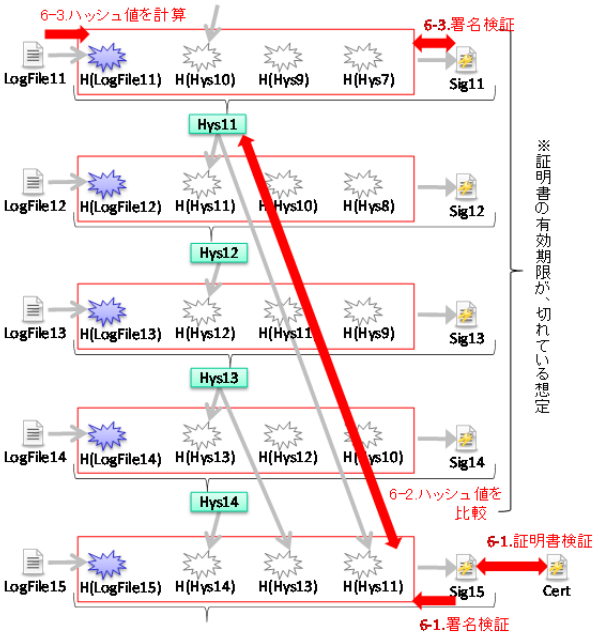

図 6　解決方針(2)　署名検証

上記のようにすることで，（解決方針１）と比較してさらに検証回数を削減し，真正性検証を高速化することが可能となる．

このとき，$2 \leqq p_1 < p_2 < p_3 < \cdots < p_n$ をどのような間隔とし，n をいくつとするかは，署名対象のデータの生成頻度や保存年数，証明書の有効期限等の様々な状況を考慮して，決定する事が望まれる．

また（解決方針１）（解決方針２）では，署名生成時に署名生成記録のハッシュ値を含める際に，複数の署名生成記録が改ざんされていないことを確認する必要があるため，従来のヒステリシス署名に比べ署名生成にかかる時間が増加することを考慮する必要がある．

3.2 実現方式

真正性を検証する対象として前提としたすべてのログデータは定期的に出力され，時系列順にソートされていること，すべてのログデータは，訴訟や監査の際に検証対象となる可能性があるという特徴がある．$2 \leqq p_1 < p_2 < p_3 < \cdots < p_n$ の決定の際には上記特徴を考慮し，ソート済みのデータを効率的に検索可能な二分木探索の考え方を応用する．

つまり，ヒステリシス署名の生成時に 2^1 個前～2^i 個前～2^n 個前の署名生成記録のハッシュ値 n 個を従来のヒステリシス署名に追加する．検証時には 2^i 個前の署名生成記録のうち各々取り得る最大のものを利用して検証を行う．2^1 個前～2^i 個前～2^n 個前の署名生成記録が追加されたヒステリシス署名の署名生成記録のイメージを図 7 に，署名生成

Vol.2014-DPS-158 No.12
Vol.2014-CSEC-64 No.12
2014/3/6

記録の連鎖構造と検証のイメージを図 8 に示す．

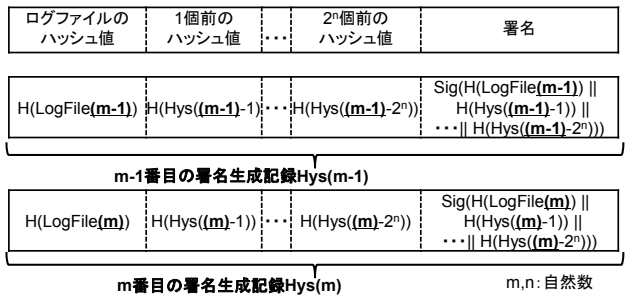

図 7 提案の署名生成記録のイメージ

図 7 に示す通り，全ての署名生成記録は，ログデータのハッシュ値と 1 個前の署名記録のハッシュ値に加えて，2^1 個前～2^i 個前～2^n 個前の署名生成記録のハッシュ値を含み，さらに，それら全てを対象にした署名が付与される．

以上のように署名生成記録の連鎖構造を構築した際の，1 個前，2^1 個前，2^2 個前の署名生成記録との関係のイメージを図 8-(1)に示す．実線は各々1 個前との関係，破線は各々 2^1 個前との関係，一点破線は 2^2 個前との関係を示したものである．

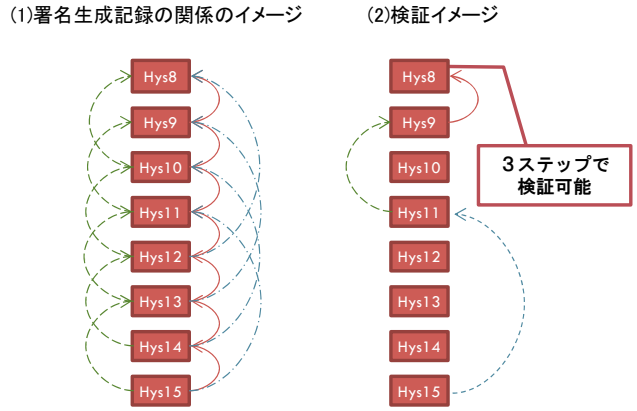

図 8 実現方式の署名生成記録の関係と検証イメージ

図 8-(2)では，有効期限内の証明書で検証可能な Hys15 を用いて Hys8 を検証する際に，二分木探索の考え方を応用し，複数の連鎖構造のうち検証回数が少なくなる連鎖構造を辿って検証するイメージを示す．まず，Hys15 の保持する 2^i 個前の署名生成記録で Hys8 がハッシュ値比較により検証できるか（Hys15 に H(Hys8)が含まれるか）を確認する．Hys15 の 2^2 個前が Hys11，2^3 個前が Hys7 であることから Hys15 に Hys8 は含まれていないため，Hys15 に含まれる署名生成記録のハッシュ値うち，Hys8 にもっとも近く Hys8 より大きい署名生成記録である 2^2 個前の H(Hys11)を選択し，ハッシュ値比較により，Hys11 を検証する．

次に，検証済みの Hys11 を 上記の Hys15 と同様に処理する．つまり Hys11 に含まれる署名生成記録のハッシュ値から 2^1 個前の H(Hys9)を選択してハッシュ値比較により Hys9 を検証する．このように検証していくことで，Hys15→Hys11→Hys9→Hys8 の順番にハッシュ値比較による検証が可能となり，従来のヒステリシス署名ではハッシュ値の比較を Hys8～14 の 7 回していたのに対し，提案した方式ではハッシュ値の比較が 3 回に減少する事がわかる．

このように 2 分探索を応用して検証することで，有効期限内の証明書で検証可能な署名生成記録と，検証対象のログデータに対応する署名生成記録との間に x 個の署名生成記録がある場合，従来はハッシュ値比較を x 回していたのに対し，最大でも $\log_2(x)$回のハッシュ値の比較回数で検証可能となる．

4. 有効性検証

4.1 有効性検証を行う上での前提条件

有効性検証に際して，真正性の検証対象とする電子データ（ログデータ）の前提条件およびその他の諸条件を以下のように仮定する．

ログデータは，1 分に 1 回ログローテンションして約 30Mbyte のログファイルが作成されることとし，すべてのログファイルに電子署名を付与することとする．すなわち，7 年間で約 370 万（＝1(file／分）×60(分)×24(時間)×365(日)×7(年)）のログファイル及び，それぞれのログファイルの真正性を保証するため，各々のログファイルに提案するヒステリシス署名が作成される．

以上の約 370 万のログファイルを高速に検証するため，2^1 個前～2^i 個前～2^{21} 個前の署名生成記録のハッシュ値を全ての署名生成記録に含める．n=21 としたのは，2^{22} が約 410 万であり，（要件 1）で想定される 7 年分のログファイル数約 370 万を超えるためである．電子署名の一般的サイズはヘッダ情報等も含めると約 2kbyte ほどになるが，上記追加情報を含めるため 1 つのログファイルに付加されるヒステリシス署名は増加することが想定される．

また，署名生成記録の比較処理にかかる必要な時間は，検証処理を行うマシンスペックや並列計算の工夫等によりまちまちになると考えられるが，ここでは 1 回の比較処理に約 1ms 程度を要するものと仮定する．

4.2 ログデータを対象とした有効性検証

ログデータを対象に提案の高速検証方式を適用した場合の有効性検証を，2 章で定義した要件を満たすかの確認によって行う．（要件 1）は提案方式がヒステリシス署名と同様に連鎖構造を保持しているため満たしている．（要件 2）～（要件 4）についての有効性を以下で詳細を述べる．

（要件 2）の電子署名の生成時の処理量，処理時間については，従来のヒステリシス署名と比べて追加で必要となる 2^1 個前～2^i 個前～2^{21} 個前の署名生成記録のハッシュ値を計算し，署名生成記録が改ざんされていない事を確認する処理が負荷とならない事を確認する．

実現方式において，図 5 の処理(5-3)に当たる処理が n=21 の場合最大で 21 回必要となる．21 回のハッシュ値比較にかかる追加処理，の大半を占めるのはハッシュ化対象である署名生成記録からハッシュ値を計算する処理だと考えられ，そのハッシュ化対象の署名生成記録サイズは合計で 2.7Kbyte（要件 3 に後述）の 21 倍である，56.7Kbyte である．これは元々の署名対象のログデータ 30Mbyte より十分小さいため，ログデータのハッシュ値 H(LogFile)を生成する処理量，処理時間に対して，追加処理による増加分は，実行上問題のある程度にはならないものと考えられる．

（要件 3）電子署名の付与によるデータ量の増加については，2^1 個前～2^i 個前～2^{21} 個前の署名生成記録のハッシュ値が追加される事が保存の際に実行上問題がない事を確認する．前提条件で整理したとおり，従来のヒステリシス署名に比較し 21 個の署名生成記録のハッシュ値を追加で含めることになる．SHA256 を用いた場合，ハッシュ値を一つ追加する毎に 32byte 必要になるため，1 つのヒステリシス署名につき 672byte 追加され，合計で約 2.7Kbyte となる．署名対象となるログファイルのサイズが約 30Mbyte の場合，保存の追加コストは 0.9%程度に収まるため，実行上問題とはならないと考える．

（要件 4）の署名検証時の処理量については，署名生成記録のハッシュ値の連鎖構造を辿る際のハッシュ値の比較回数が少なくなり，オンデマンドでの真正性検証に支障がないことを確認する．

図 8 で示した通り，有効期限内の証明書で検証可能な署名生成記録と，検証対象のログデータに対応する署名生成記録との間に x 個の署名生成記録がある場合，提案方式では最大でも $\log_2(x)$回のハッシュ値の比較回数で検証可能となる．つまり，7 年分の署名の個数である約 370 万を x とすると，$\log_2(x)$は約 21.8 になり，証明書の有効期限が切れていないと考えられる最新の署名生成記録から，最大でも 21 回のハッシュ値比較により，7 年前のどの署名生成記録を検証することが可能である．

ハッシュ値 1 回の比較に 1ms 要すると想定すると，1 回の検証処理を 21ms で実行可能なため，検証をオンデマンドに実行することも実行上問題がない．

以上, 実現方式でログデータに署名を行うことによって，要件 1 から要件 4 を満たすことを示した．

5. まとめおよび今後の課題

本報告では，電子データの真正性を保証する技術として従来のヒステリシス署名を適用する場合の課題を挙げ，それを解決する高速検証方式を提案した上で，提案方式の有効性を示した．

従来のヒステリシス署名を適用する場合の課題は，署名対象ファイル数が大量になると，有効期限が切れた真正性の検証に膨大な時間かかり，オンデマンドでの検証が困難となることである．この問題を解決する方法として，（解決方針 1）p(p≧2)個前の署名生成記録のハッシュ値を含める方法，および（解決方針 2）複数の署名生成記録のハッシュ値を含める方法，を取ることで高速に真正性を検証可能とする方式を提案した．提案した方式は，オンデマンドでの真正性検証にも支障なく高速に実行することが可能であり，その他の真正性検証に求められる要件（署名生成時の処理量，処理時間，およびデータ量の増加が実行上問題とならないこと）も満たすことを確認することで，本提案方式の有効性を示した．

今後の課題としては，現状定性的な評価に留まっている提案方式の有効性評価を，提案した方式を実装して実機による評価を行うことで，定量的に示していく必要があると考える．

参考文献

1) 独立財団法人 情報処理推進機構, 暗号の危殆化に関する調査報告書,2005
http://www.ipa.go.jp/security/fy16/reports/crypt_compromize/
2) 洲崎誠一, 松本勉: 電子署名アリバイ実現機構－ヒステリシス署名と履歴交差, 情報処理学会論文誌, Vol.43, No.8, pp.2381-2393(2002)
3) 芦野佑樹, 佐々木良一: セキュリティデバイスとヒステリシス署名を用いたデジタルフォレンジックシステムの提案と評価, 情報処理学会論文誌, Vol.49, No.2, pp.999-1009(2008)