

より効率的な墨塗りシステムの開発と評価

増渕 孝延^{†1,1} 小川 典子^{†1,2} 鹿志村 浩史^{†1,3} 石井 真之^{†1,4} 佐々木 良一^{†1,5}

† 東京電機大学工学部 〒101-8457 東京都千代田区神田錦町 2-2

E-mail: †1 {masubuchi, ogawa, kasimura, saneyuki}@isl.im.dendai.ac.jp

†2 sasaki@im.dendai.ac.jp,

あらまし 署名付き電子文書は、必ずしも署名したときそのまま利用されるとは限らない。例えば、行政文書が情報公開制度に基づき開示される際、そこに記述された個人情報や国家安全情報は、通常「墨塗り」された上で開示される。この一連の手続きを従来の電子署名技術を用いて実現しようとした場合、保管時に付与した署名が、その文書の一部を秘匿することによって検証できなくなる問題を「電子文書墨塗り問題」と呼び、これに対応可能な電子文書の真正性保証技術が提案されている。本稿では、既に提案されている真正性保証技術によって、文書を加工する際に生じるデータ量増加の改善および評価、さらに人に優しいユーザインタフェースの考察と実装を行った。

キーワード 電子署名, 情報公開, プライバシー保護

Development and evaluation of a more efficient electronic document SUMI coating system

Takanobu MASUBUCHI^{†1} Noriko OGAWA^{†2} Hiroshi KASHIMURA^{†3}

Saneyuki ISHI^{†4} and Ryoichi SASAKI^{†5}

†4 School of Engineering, Tokyo Denki University 2-2 Kandanishikicho, Chiyoda-ku, Tokyo,

101-8457 Japan

E-mail: †1 {masubuchi, ogawa, kasimura, saneyuki}@isl.im.dendai.ac.jp

†2 sasaki@im.dendai.ac.jp,

Abstract Digital signature does not allow any alteration of the document. However, “appropriate” alteration should be allowed for some signed document because of other security requirements etc. Disclosure of official information is a typical example of this. Sensitive information such as private information should be sanitized from the original digitally signed document when it is disclosed. “Digital document sanitizing problem” is the problem that signed document cannot be verified if some part of the signed document is concealed. The already proposed SUMI coating system which can solve digital document sanitizing problem. In this paper, we developed and evaluation of a more efficient electronic document SUMI coating system. Furthermore, we examine consideration of a user interface gentle to people and mount it on the above system.

Keyword digital signature, information disclosure, privacy issue

1. 背景と目的

電子文書の安全性を高める技術に電子署名技術がある。この技術は電子文書に対するいかなる改変も検知できるように設計されている為、不正者による改ざんから電子文書を守るということに関しては非常に有効である。しかし、電子文書の有効活用という観点から

は一切の加工が許されなくなるため逆に障害となる。

例えば、行政機関における情報公開の際に、プライバシー保護等の個人情報の削除であろうと改変が加えられたという点で、文書の真正性が確認できない。このように、従来の電子署名技術の利用では、「公開された文書の真正性保障」と「プライバシー情報の保護」と

いう2つの重要なセキュリティ要件を両立することができない。この問題を「電子文書墨塗り問題」と呼び、この問題に対する解決策として一部を秘匿した後であっても、元の文書との(秘匿部分以外の)同一性を保障可能な電子署名方式(以下 この方式を仮に「墨塗りシステム」と呼ぶ)が提案された。

本研究では、既に提案された従来の方式¹⁾とCES²⁾の問題点を検討すると共に、解決策を提案し、より効率的な墨塗りシステムの開発と評価を行う。

2. 情報公開制度の墨塗りシステムの流れ

情報公開制度を例に墨塗りシステムの全体の流れを図1に示す。

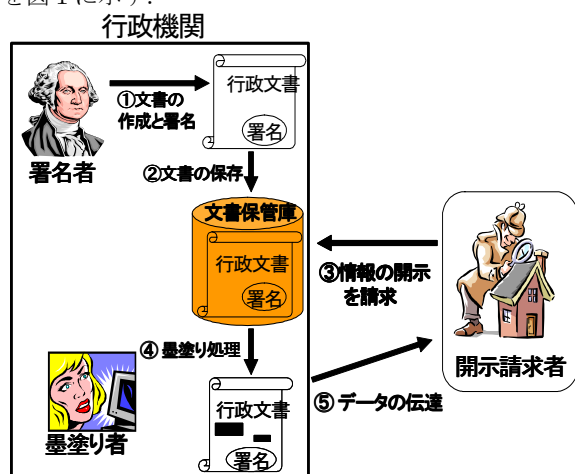


図1 墨塗りシステムの全体の流れ

本研究では、図1のように署名者、墨塗り者、開示請求者の3者それぞれの立場で問題の検証を行っていく。また、署名者、墨塗り者は同一の行政機関の人物、開示請求者を国民と想定した。

- 署名者 … 行政機関の長
- 墨塗り者 … 行政機関の職員
- 開示請求者 … 国民

3. 従来の墨塗り手順

ここでは、文献¹⁾にて提案された墨塗りシステムの方式3、方式4について解説する。

方式3

署名生成手順

- ① オリジナル文書を各構成要素(以下、ブロックと呼ぶ)に分割する。
- ② 各ブロックのデータのハッシュ値を算出し、算出されたN個のハッシュ値を結合したデータに対し、署名を生成。(1個の署名を生成)

- ③ 生成された署名(1個)と、オリジナル文書とからなるデータを署名付きオリジナル文書とする。

墨塗り手順

- ① 開示対象である署名付きオリジナル文書の中から、不開示情報を含む各ブロック(墨塗りされるブロック)を選択する。
- ② 選択された各ブロックのハッシュ値と、それ以外の各ブロックと、署名とからなるデータを開示情報とする。(墨塗りされたブロックはそのままハッシュ値を利用し、墨塗り以外のブロックについてはブロック自体を用いる)

検証手順

- ① 開示文書のうち、(墨塗り部分のハッシュ値ではない)元のデータ自体が与えられている各ブロックのハッシュ値を算出する。
- ② オリジナル文書作成者の公開鍵を用いて、開示文書の署名を検証し、ハッシュ値を算出する。
- ③ ①で算出されたハッシュ値と、開示文書に含まれている墨となっている部分のハッシュ値(合計N個生成される)からさらにハッシュ値を算出する。
- ④ ②で生成されたハッシュ値と③で生成されたハッシュ値を比較検証し、検証結果を出力する。

署名付き文書

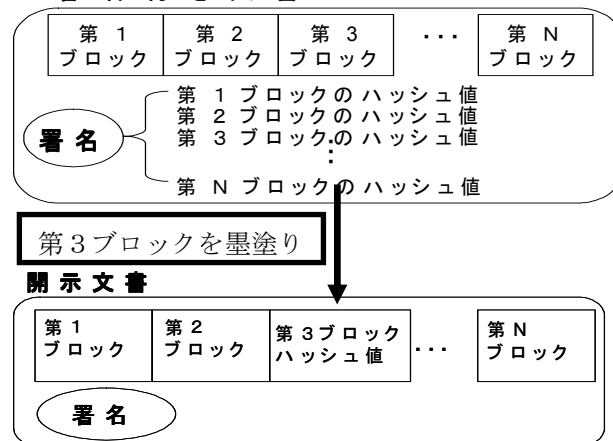


図2 従来(方式3)の墨塗りシステム

方式3では、開示文書に墨塗りされたブロックが存在しても墨塗り以外のブロックが変更されていなければ署名検証は成功するが、墨塗り以外の部分に変更が存在した場合には署名検証に失敗する。つまり、開示文書の真正性を確認するのは可能である。しかし、開示文書の前後の文脈から墨塗りブロックが推測される可能性があり、推測が正しいかどうかを推測したプロ

ックのハッシュ値を計算し、開示されたハッシュ値と一致するかどうかを調べることで確認可能である。この確認手段は墨塗り部分の情報を推定しようと試みる攻撃者が、開示文書自体の推定結果の正当性を保証する手段として利用できてしまうので、各ブロックのハッシュ値の推測を困難にする為に、各ブロックに乱数を付加すると、仮に同一文字のブロックであろうと乱数によって、異なるハッシュ値が生成されるので、墨塗り部分の推測が困難になる方式4が提案されている。

方式4

署名生成手順

- ① オリジナル文書をN個のブロックに分割する。
- ② N個のブロックそれぞれに対して生成された乱数を結合したデータ(以下、乱数付きブロックと呼ぶ)を生成する。
- ③ 各乱数付きブロックのハッシュ値を算出し、算出されたN個のハッシュ値を結合したデータに対し、署名を生成(1個の署名を生成)。
- ④ 生成された署名(1個)とN個の乱数付きブロックとからなるデータを署名付きオリジナル文書とする。

墨塗り手順

- ① 開示対象である署名付きオリジナル文書の中から不開示情報を含む各ブロックを選択する。
- ② ①で選択された各乱数付きブロックのハッシュ値と、それ以外の各乱数付きブロックと、署名とからなるデータを開示文書とする。

検証手順

- ① 開示文書のうち、(墨塗り部分のハッシュ値ではない)元のデータ自体が与えられている各乱数付きブロックのハッシュ値を算出する。
- ② オリジナル文書作成者の公開鍵を用いて、開示文書の署名を検証し、ハッシュ値を算出する。
- ③ ①で算出されたハッシュ値と、開示文書に含まれている墨となっている部分のハッシュ値(合計N個生成される)からさらにハッシュ値を算出する。
- ④ ②で生成されたハッシュ値と③で生成されたハッシュ値を比較検証し、検証結果を出力する。

4. 従来の墨塗り方式の問題点

(問題1) 署名後の文書に墨を塗れる箇所は、墨塗り

者に墨を「塗る」「塗らない」を委ねられている。

(問題2) 墨となる部分の推測を防ぐために全てのブロックに乱数を付加するので、データ量が通常の文書よりも数倍に増加してしまう。

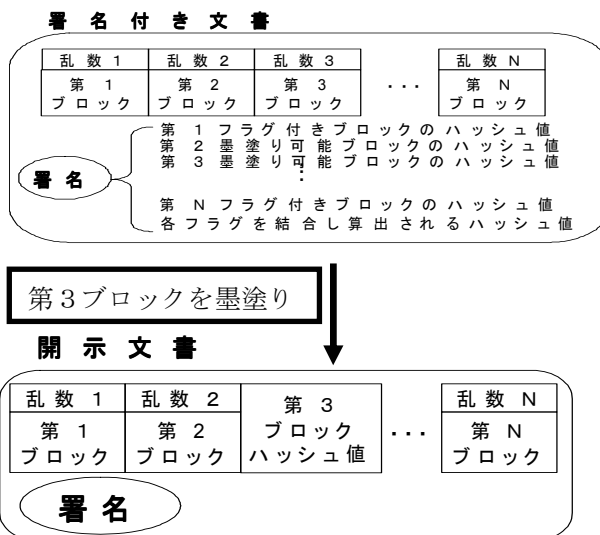


図3 従来(方式4)の墨塗りシステム

5. 問題解決のアプローチ

- ① (問題1)に対するアプローチ: 署名者が墨塗り不可部分、可能部分、必須部分を区別し、指定する。
- ② (問題2)に対するアプローチ: 署名者と墨塗り者が用いる乱数は、乱数生成用の種(Seed)を設定後、乱数を生成し、乱数付きで保存せず、Seedを保存する。
- ③ (問題2)に対するアプローチ: 署名者によって墨塗り処理が禁止されている箇所である墨塗り不可部分には乱数を付加しない。

6. 提案方式

署名生成手順

- ① 文書を1文字ごとにN個のブロックに分割し、各ブロックに対し墨塗り不可部分、墨塗り可能部分、墨塗り必須部分を示すフラグをそれぞれ指定し、付加する。
- ② Seedを指定して乱数を生成し、Seedを保存する。
- ③ 墨塗り可能部分、墨塗り必須部分と指定した各ブロックにのみ②で生成された乱数を付加する。(以下このように乱数を付加したブロックを「墨塗り可能ブロック」「墨塗り必須ブロック」と呼ぶ。)
- ④ ③で指定した各墨塗り可能ブロックと各墨塗り

必須ブロックとそれ以外の墨塗り不可部分のフラグ付きブロック(それぞれのブロックの合計N個)のハッシュ値を算出する。

- ⑤ 算出されたN個のハッシュ値を結合したデータに署名者の秘密鍵で署名を生成する。
- ⑥ オリジナル文書, 生成された署名, 各ブロックのフラグの集合からなるデータを署名付きオリジナル文書とする。

提案方式の署名付きオリジナル文書

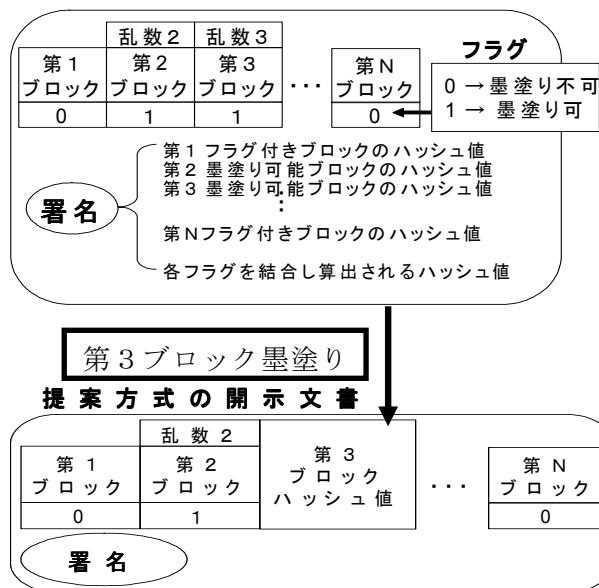


図4 提案方式の墨塗りシステム

墨塗り手順

- ① 開示対象の署名付きオリジナル文書を署名生成手順と同様に文書をフラグ付きブロックに分割し, Seedを指定して乱数を各ブロックに付加する。
- ② 生成された墨塗り必須ブロックを墨塗り処理する。
- ③ 生成された墨塗り可能ブロックの中から不開示情報を含むブロックを選択し, 墨塗り処理する。
- ④ 各墨塗りブロックのハッシュ値(以下「墨塗り部分」と呼ぶ)と, それ以外の各墨塗り可能ブロックおよび墨塗り不可部分のブロックのハッシュ値と, 署名からなるデータを開示文書とする。

検証手順

- ① オリジナル文書作成者の公開鍵を用いて, 開示文書の署名を検証し, ハッシュ値(Hとする)を算出する。
- ② 開示文書のうち, 各墨塗り部分以外のブロックのハッシュ値を算出する。
- ③ ②で算出されたハッシュ値と墨塗り部分のハッシュ値(合計N個生成される)をからさらにハッシュ値を算出する(H'とする)。

④ ①で生成したHと③で生成したH'を比較検証し, 検証結果を出力する。

7. ユーザインタフェースの考察と実装

我々は, 実際に JAVA 言語を用いて墨塗りシステムを用いるそれぞれのユーザの視点でユーザインタフェースを考察した。墨塗りシステムにおける一連の処理と共に実行画面(図5~10に示す)より解説する。

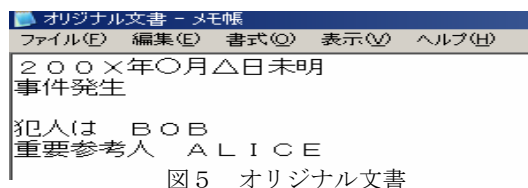


図5 オリジナル文書

例えば, 図5のようなオリジナル文書を署名者が作成したとする。

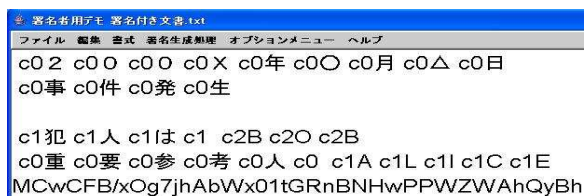


図6 署名後の実行画面

図5のオリジナル文書を一文字ずつのブロックに分割し, 全てのブロックにそれぞれフラグを付加する。図6におけるc0, c1, c2はそれぞれフラグを表している。c0は墨塗り不可部分, c1は墨塗り可能部分, c2は墨塗り必須部分を表している。図6の4行目の“犯人は”の部分と5行目の“ALICE”の部分を墨塗り可能部分とし, さらに4行目の“BOB”を墨塗り必須部分とする。次にSeedをセットして乱数を生成し, 墨塗り可能部分と必須部分にのみ乱数を付加する。墨塗り不可部分には墨塗り処理が禁止されているため乱数を付加しない。最後に署名者の秘密鍵で署名を生成し, 署名を付加する。

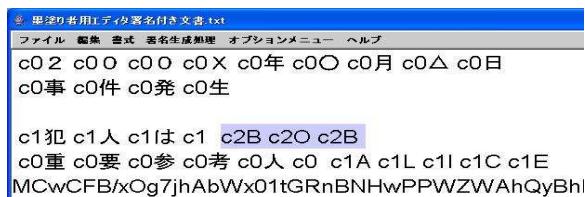


図7 墨塗り前の選択確認(実行画面)

墨塗り者の処理として, 不開示部分に墨塗り処理をする。図7の4行目の“BOB”の部分は署名者により墨塗り必須部分と指定されているので墨塗りする。

墨塗り処理を行う前に墨塗り箇所を確認ができるように選択部分を表示するように設計した。

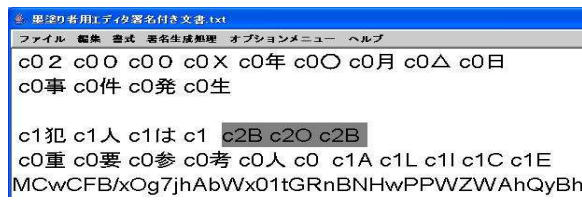


図 8 墨塗り前の選択（実行画面）

図 8 は墨塗り箇所の確認後、実際に墨塗りするが、墨塗りする“BOB”を“***”とパスワード入力をしたときのように表示、または図 8 のように黒くするだけでは墨塗り箇所の文字数の推測が可能となる。

そこで、可読性も考慮し、図 9 のように墨塗り部分を【■】と表示するように設計した。

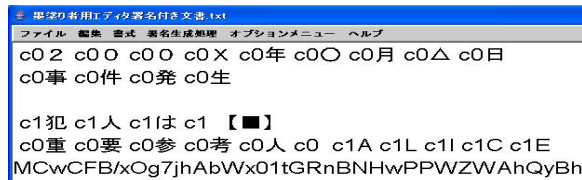


図 9 墨塗り後（実行画面）

また、墨塗り不可部分は墨塗りを禁止しているため、自由に墨を塗ろうとする攻撃者の攻撃を防ぐと共に誤って墨を塗ってしまうという誤操作の防止にもなる。



図 10 署名検証（実行画面）

図 10 は開示請求者が受け取った一部開示決定の開示文書の表示画面である。選択されている部分は署名者の秘密鍵によって生成されたハッシュ値である。



図 11 二つのハッシュ値の比較（実行画面）

図 11 では初めに署名を署名者の公開鍵で復号を試み、ペア鍵が一致している場合のみハッシュ値を算出する。ペア鍵の一致を確認後、自動的に墨塗り部分のハッシュ値と、墨塗り部分以外のハッシュ値からハッシュ値を算出し、二つのハッシュ値が等しいかを人の

目にも分かりやすく表示するように設計した。

8. データの比較と評価

今回は独自に設定した文字数のデータを表 1 に示す。

表 1 検証を行ったデータ

すべて全角文字の場合	文字数	データ量(バイト)
500 文字に設定したテキスト	500	1,000
1,000 文字に設定したテキスト	1,000	2,000
A4 用紙 2 枚分	2,092	4,032
3,000 文字に設定したテキスト	3,000	6,000
5,000 文字に設定したテキスト	5,000	10,000
10,000 文字に設定したテキスト	10,000	20,000

表 1 の“A4 用紙 2 枚分”とは無作為に 10 個選択した全角文字の論文を A4 用紙 2 枚分のみ抽出し、それぞれ文字数を計測し平均値を算出したデータである。

また、図 12 は東京都の情報公開制度の運用状況である。請求件数（1 年間の全請求件数）に対して、部分開示である一部開示決定の文書は請求件数の約 35%ほどの割合である。

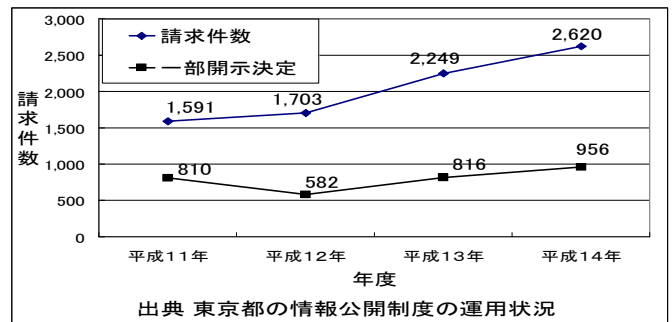


図 12 東京都の開示請求の状況

表 1 の検証データと図 12 のデータを用いて、比較・検証を表 2 の 3 つの各方式でそれぞれ行った。

表 2 3 つの各方式

方式 No.	方式名	Seed	フラグ
①	従来方式	x	x
②	提案方式 フラグ無し	○	x
③	提案方式	○	○

図 13 は署名者が署名生成し、署名付きで保存する文書（以下、保存時データ量と呼ぶ）のデータ量を示す。

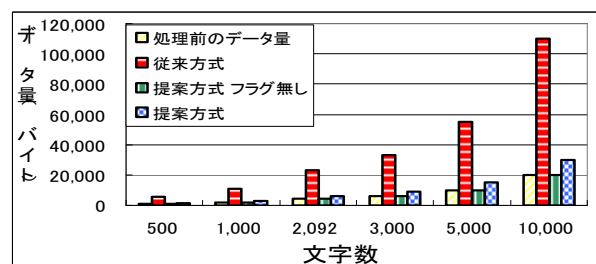


図 13 各方式の保存時データ量の比較

図 13 より提案方式の保存時のデータ量は従来方式と比べると大幅に減少していることがわかる。これは保存するときに従来方式では全てのブロックに乱数を付加して保存していることにに対し、提案方式では Seed のみを別途保存しているためである。

図 14 は墨塗りが開示請求者に渡す開示文書（以下、開示時データ量と呼ぶ）のデータ量を示す。

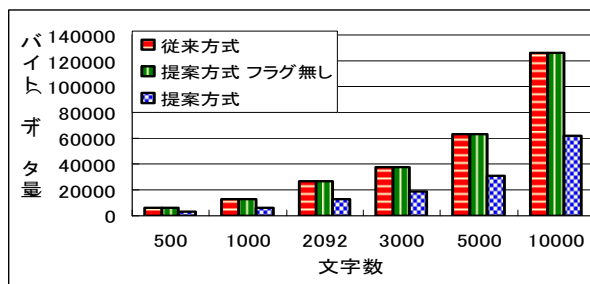


図 14 各方式の開示時データ量の比較

図 14 より“従来方式”と“提案方式のフラグ無し方式”はデータ量が等しくなる。また、フラグ付き方式である“提案方式”のデータ量は少ない。

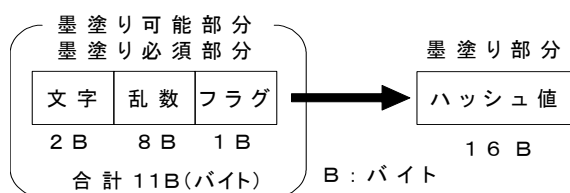


図 15 墨塗り前と墨塗り後のブロックの変化

提案方式では、図 15 のようにブロックを構成した。墨塗り可能部分、必須部分は墨を塗ることにより 11 バイトから 16 バイト（墨塗り部分）に置き換わるため墨塗り部分の割合が高いとデータ量が増加する。また、墨塗り不可部分には乱数を付加しないため不可部分の割合が高いとデータ量の増加を抑えられる。

図 14 は、墨塗り不可部分の割合を 80% と墨となる部分を 20% として計測したデータである。

墨となる部分と墨塗り不可部分の割合をそれぞれの方式で計測した結果を図 16 に示す。

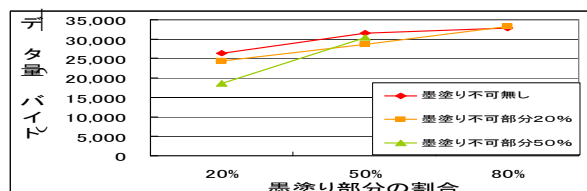


図 16 墨部分と墨塗り不可部分の割合のデータ量

図 16 より墨塗り部分、墨塗り不可部分を共に全体の 20% に設定した時にデータ量は逆転した。

9. まとめ

本研究では、既に提案された従来の方式の問題点を検討し、より効率的な墨塗りシステムの開発と評価を行った。さらに署名者、墨塗りが開示請求者それぞれのユーザの視点で、人に優しいユーザインタフェースの考察と実装を行った。

我々の提案した方式では、Seed を指定した乱数生成と保存、墨塗り不可と可能部分とを分けることにより効率性を実現した。また、墨塗り対象となる文書の墨塗り不可部分の割合が多いほどデータ量を削ることが可能である。計測したデータは独自に設定したデータに基づき計測したので、システムを用いる環境によってデータ量は異なる。特に、文書の作成者である署名者が墨塗り不可部分や墨塗り必須部分を指定する必要が生じた時には、提案方式が非常に有効である。

10. 今後の課題

今回は、ブロックの分割を 1 文字ごとに分割した。これは、不開示ブロックに開示部分が含まれないようにする為である。墨塗り部分のブロックは墨塗り以外のブロックとの比較によりブロック数は推測可能である。つまり墨塗り部分の文字数が推測可能になってしまう。そのため今後は、ブロックの分割方法を改善する必要がある。品詞ごとに分割、文書の区切り記号を指定して正規表現による分割、もしくは XML のようなあらかじめ構造化された文書の構成要素を用いたブロック分割することにより文字数の推測を防ぐ。

また、我々の提案方式では、署名者の負担が大きくなってしまった。1 つのオリジナル文書に署名者、墨塗りが一人ずつ担当するので、今後は XML 署名 (Enveloped 署名) を用いて、署名者が複数人、文献³⁾のように墨塗りが複数人であるという形式にも対応できるシステム作りが今後の課題である。

文献

- 宮崎 邦彦, 洲崎 誠一, 岩村 充, 松本 勉, 佐々木良一, 吉浦 裕, “電子文書墨塗り問題”, 信学技法 ISEC2003-20, 61-67, 2003
- R.Steinfeld, L.Bull, and Y.Zheng, "Content Extraction Signatures", ICISC 2001, 285-304, 2001
- 宮崎 邦彦, 岩村 充, 松本 勉, 佐々木 良一, 吉浦 裕, “開示条件を制御可能な電子文書墨塗り技術”, SCIS2004, 515-520, 2004