秘密分散法を応用した電子透かし
—分散情報としてのバースマークの利用—

M2007MM030 棚瀬真臣
指導教員：真野芳久

1 はじめに

近年、ソフトウェアにおける著作権侵害が大きな問題となっており、プロテクション技術が重要となってきている。ソフトウェアの盗用を防ぐことを目的としたプロテクション技術の一つとして、電子透かしと呼ばれる技術が存在する。しかし、ソフトウェアに対する電子透かしには、画像などへの電子透かしよりも電子透かしが発見されやすい、電子透かしを埋め込むことでソフトウェアの処理時間やサイズが大きくなる、などの欠点がある。また、ソフトウェアの部分的な盗用を防ぐために 1 つのソフトウェアに複数のソフトウェア透かしを埋め込むことも考えられ、それらの欠点を改善することは、より重要となってきている。

そこで本研究では、1 つのソフトウェアに複数のソフトウェア透かしを埋め込む状況で、ソフトウェア透かしの欠点を改善する手法として、秘密分散法とバースマークを用いる手法を提案し、具体例を用いて実現可能だと示す。

2 関連技術

2.1 電子透かし

電子透かしは、デジタルコンテンツ中に特定の情報を埋め込む技術のことである。デジタルコンテンツに電子透かしを埋め込むことにより、コンテンツの不正な改ざんを検出できる、コンテンツに盗用が起こった時に著作権を主張することができる、などのセキュリティの向上が期待できる。特に、ソフトウェアに対する電子透かしのことをソフトウェア透かし (以下透かしと略記) と呼ぶ。

2.2 バースマーク

バースマークは、プログラムから抽出した特徴あるいは、その特徴の一致をもってプログラム盗用の可能性を識別する技術である。透かしが予めプログラムに埋め込んでおいた意図的な情報を用いることに対して、バースマークはプログラムに表れた特徴を用いる。

2.3 秘密分散法

秘密分散法は、Shamir[4] と Blakley[5] によって、それぞれ独自に発表された暗号化方法である。秘密分散法では、秘密情報を複数の分散情報に分散する。また、決められた数片の分散情報を集めないと秘密情報は復元されない。

2.3.1 (k,n) 完全秘密分散法

秘密情報 S の分散情報 $W_1, W_2, ..., W_n$ が次の 2 条件を満たすとき (k,n) 完全秘密分散法という。

- 任意の k 個以上の分散情報から S が正しく復元できる。
- 任意の $k-1$ 個以下の分散情報からは S の情報が全く得られない。

(k,n) 完全秘密分散法において、任意の分散情報 W_j の情報量 $|W_j|$ は、秘密情報 S の情報量 $|S|$ に対して $|W_j| \geq |S|$ となることが知られている。

2.3.2 (k,L,n) ランプ型秘密分散法 [6]

秘密情報 S の分散情報 $W_1, W_2, ..., W_n$ が次の 3 条件を満たすとき (k,L,n) ランプ型秘密分散法という。

- 任意の k 個以上の分散情報から S が正しく復元できる。
- 任意の $k-1$ 個から $k-L+1$ 個の間の分散情報からは、段階的に S の情報が得られる。
- 任意の $k-L$ 個以下の分散情報からは S の情報が全く得られない。

(k,L,n) ランプ型秘密分散法において、任意の分散情報 W_j の情報量 $|W_j|$ は、秘密情報 S の情報量 $|S|$ に対して $|W_j| \geq \frac{|S|}{L}$ となることが知られている。また、$L=1$ の時完全秘密分散法となる。

3 透かしへの秘密分散法の応用

本章では、秘密分散法を透かしに応用する手法を述べる。透かしに秘密分散法を用いることで、透かし情報を解読することが難しくなる。

透かしではこれまで図 1 に示すような埋め込み (Embed)、抽出 (Recognize) モデルが一般的に用いられているが、本研究では秘密分散法を応用するために、このモデルを拡張して利用する。

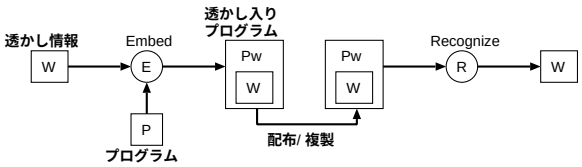

図 1 従来の透かし埋め込み抽出モデル

本方法では秘密分散法を透かしに応用し、図 2 に示すモデルを用いる。透かし埋め込みでは、まず秘密情報 S を n 個の分散情報 $W_1, ..., W_n$ に分散する。それらの分散情報を、透かし埋め込み機能 E でプログラム P に埋め込み、透かし入りプログラム P_W を生成する。また、透かし入りプログラム P_W から透かし抽出機能 R で k 個の分散情報 $W_{j_1}, ..., W_{j_k}$ を抽出し、秘密情報 S を復元する。

ここで、プログラムサイズが埋め込み情報の h 倍増えるとすると、透かしを n 個埋め込む場合のプログラムサイズの増加量は、「$n \times h \times$ 埋め込み情報量」となる。よっ

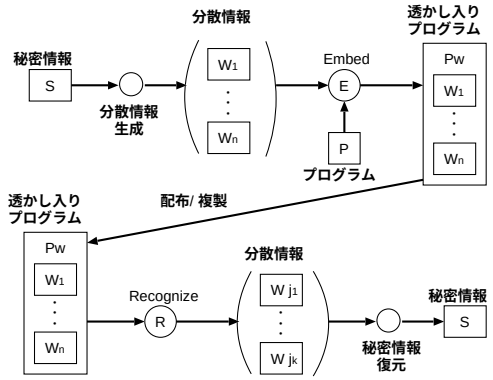

図 2 秘密分散法を応用した透かし埋め込み抽出モデル

て、完全秘密分散法での分散情報 W_j を n 個埋め込んだ場合のプログラムサイズの増加量は $h\sum|W_j| \geq hn|S|$ となり、最善で $hn|S|$ となる。また、ランプ型秘密分散法での分散情報 W_j を n 個埋め込んだ場合のプログラムサイズの増加量は $h\sum|W_j| \geq hn\frac{|S|}{L}$ となり、最善で、$hn\frac{|S|}{L}$ となる。

4 分散情報としてのバースマーク

本章では、3 章で述べた秘密分散法を応用した透かし埋め込み手法の cost を改善するために、透かしの一部を埋め込み対象のバースマークで置き換える手法を提案する。

4.1 提案手法の概要

図 3 に示すように、1 個の分散情報 W_i をバースマーク $BM(P)$ で置き換えるとする。$BM(P)$ で置き換えられた分散情報 W_i は透かしとして埋め込む必要はなくなり、分散情報 W_i の抽出はバースマークを抽出することで可能となる。結果として、1 個の分散情報を透かしとして埋め込む必要がなくなり、透かし埋め込み cost は分散情報 1 個分少なくなる。

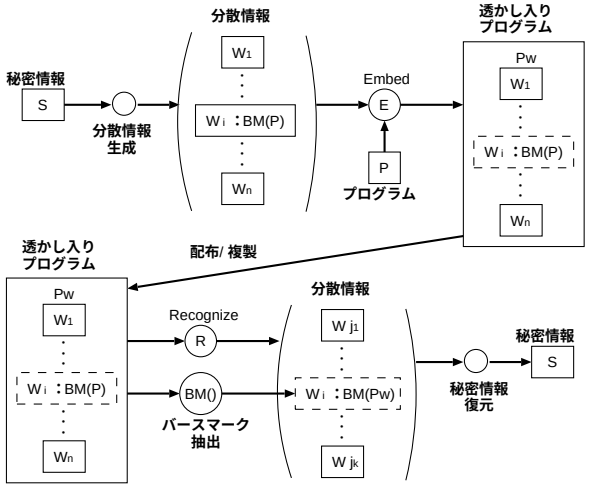

図 3 バースマークを利用する透かし埋め込み抽出モデル

問題点として、透かしを埋め込むことでバースマークが変わる可能性があることが挙げられる。透かし埋め込みの際はプログラム P から抽出したバースマーク $BM(P)$ を分散情報 W_i として用いるのに対して、透かし抽出の際は、透かし入りプログラム P_W から抽出したバースマーク $BM(P_W)$ を W_i として用いている。よって、本方法でバースマークを用いるには $BM(P) = BM(P_W) = W_i$ である必要がある。バースマークを変えないで透かしを埋め込む簡単な方法として、透かしの埋め込み場所とバースマークの抽出場所を変える方法が挙げられる。

4.2 藤井らの (k,n) しきい値法 [7] を用いた構成法

藤井らの (k,n) しきい値法は、生成行列 G と、秘密情報 S を $(n-1)$ 個に分割した分割情報 S_i と、$(k-1)(n-1)$ 個の乱数成分 $R_{d,e}$ からなるベクトル U を用いて分散情報 W を生成する。

$$W = (R_{1,1}, ..., R_{k-1,n-1}, S_1, ..., S_{n-1})G$$

本方法では、乱数成分 R の代わりに、秘密情報 S と $k-1$ 個の分散情報 W から一意に決まる代用情報 R' を用いる。この R' を用いて分散情報を生成することで、他の分散情報が定まる。また、本手法は $k-1$ 個のバースマークを分散情報として使うことが可能であり、サイズ増加量は最善で $h\{n-(k-1)\}|S|$ となる。

4.2.1 $(2,3)$ しきい値法での構成

本節では、$k=2, n=3$ の場合での構成法を述べる。まず、秘密情報 S と代用情報 R' からなるベクトル U と生成行列 G から分散情報 $W_1 = BM\ (P)$ となるような分散情報 W_1, W_2, W_3 を生成するとする。

$$(W_1, W_2, W_3) = (R'_1, R'_2, S_1, S_2)G$$

ここで仮に $BM\ (P) = 101011, S = 111000$ として、分散情報 W_1 を生成する処理 $W_1 = UG_1$ から代用情報 R' を求める。また、分散情報 W_1 は 2 個 $(n-1$ 個$)$ に分割され、$E(0)$ は $2 \times 2(n-1 \times n-1)$ の単位行列となる。

$$(101,011) = (R'_1, R'_2, 111,000)\left[\frac{E(0)}{E(0)}\right]$$
$$(101,011) = (R'_1, R'_2)\left[\ E(0)\ \right] \oplus (111,000)\left[\ E(0)\ \right]$$
$$(R'_1, R'_2) = (101,011) \oplus (111,000)$$
$$(R'_1, R'_2) = (010,011)$$

となり、秘密情報 S と分散情報 W_1 から代用情報 R' が決まる。このようにして求めた代用情報 R' を用いて分散情報を生成することで、分散情報 W_2, W_3 が定まる。

4.2.2 (k,n) しきい値法での構成

$k-1$ 個の分散情報 W、秘密情報 S と代用情報 R' からなるベクトル U、生成行列 G を以下のように定める。

$$W = (W_{i_1}, ..., W_{i_{k-1}}) = (BM_{i_1}(P), ..., BM_{i_{k-1}}(P))$$
$$U = (R'_{1,1}, ..., R'_{k-1,n-1}, S_1, ..., S_{n-1})$$
$$G = (G_{i_1}, .., G_{i_{k-1}})$$

分散情報を生成する処理は $W = UG$ と表すことができる。生成行列 G で秘密情報 S に関係する部分を G_s、代用情報 R' に関係する部分を G_r とすると

$$W = UG = U \left[\frac{G_r}{G_s} \right]$$
$$W = (R'_{1,1}, ..., R'_{k-1,n-1})G_r \oplus (S_1, ..., S_{n-1})G_s$$
$$W \oplus (S_1, ..., S_{n-1})G_s = (R'_{1,1}, ..., R'_{k-1,n-1})G_r$$

とすることができる。ここで、$W \oplus (S_1, ..., S_{n-1})G_s = W_s$ と表し、G_r の逆行列 ${G_r}^{-1}$ を求める。また、G_r は $(k-1)(n-1) \times (k-1)(n-1)$ であり、逆行列 ${G_r}^{-1}$ が存在する。

$$W_s = (R'_{1,1}, ..., R'_{k-1,n-1})G_r$$
$$W_s {G_r}^{-1} = (R'_{1,1}, ..., R'_{k-1,n-1})$$

となり、秘密情報 S と $k-1$ 個の分散情報 W から代用情報 R' が定まる。また、このようにして求めた代用情報 R' を用いて分散情報を生成することで、他の分散情報が定まる。

4.3 (k, L, n) しきい値法での構成法

藤井らの (k, n) しきい値法 [7] を、(k, L, n) ランプ型に拡張し用いる。秘密情報 S を $L(n-1)$ 個に分割した分割情報 $S_{i,j}$ と、$(k-L)(n-1)$ 個の乱数成分 $R_{d,e}$ からなるベクトル U を用いて分散符号化を行うことで、(k, L, n) ランプ型に拡張することが可能である。

$$W = (R_{1,1}, ..., R_{k-L,n-1}, S_{1,1}, ..., S_{L,n-1})G \quad (1)$$

また、代用情報 R'(秘密情報 S と $k-L$ 個の分散情報 W から一意に決まる) を用いて分散情報を生成することで、$k-L$ 個のバースマークを分散情報として使うことが可能であり、サイズ増加量は最善で $h\{n-(k-L)\}\frac{|S|}{L}$ となる。

$$W = (W_{i_1}, ..., W_{i_{k-L}}) = (BM_{i_1}(P), ..., BM_{i_{k-L}}(P))$$
$$U = (R'_{1,1}, ..., R'_{k-L,n-1}, S_{1,1}, ..., S_{L,n-1})$$
$$G = (G_{i_1}, .., G_{i_{k-L}})$$

$$W = UG = U \left[\frac{G_r}{G_s} \right] \quad (2)$$
$$W = (R'_{1,1}, ..., R'_{k-L,n-1})G_r \oplus (S_{1,1}, ..., S_{L,n-1})G_s \quad (3)$$
$$W \oplus (S_{1,1}, ..., S_{L,n-1})G_s = (R'_{1,1}, ..., R'_{k-L,n-1})G_r \quad (4)$$
$$W_s = (R'_{1,1}, ..., R'_{k-L,n-1})G_r \quad (5)$$
$$W_s {G_r}^{-1} = (R'_{1,1}, ..., R'_{k-L,n-1}) \quad (6)$$

5 分散情報の保管

提案手法の cost と resilience を改善する手法として、分散情報を保管することが挙げられる。n 個の分散情報全てを埋め込むのではなく、任意の d 個 $(d \leq n)$ の分散情報だけを埋め込み、残り $(n-d)$ 個を保管する。

5.1 resilience

ソフトウェア P'_W から $k-1$ 個以下の分散情報しか抽出できない場合を考える。秘密情報 S の復元には、k 個の分散情報が必要となるので、P'_W からの分散情報だけでは、秘密情報 S を復元できない。しかし分散情報を保管することで、対象から抽出できた分散情報が $k-(n-d)$ 以上ならば、保管情報を用いることで秘密情報 S の復元が可能となる。

5.2 cost

4 章で述べた手法でのサイズ増加量は、$h\{n-(k-L)\}\frac{|S|}{L} = h\{|S| + (n-k)\frac{|S|}{L}\}$ となり、最善でも $h|S|$ は増加する。しかし分散情報を保管した場合、サイズ増加量は $hd\frac{|S|}{L}$ となる。さらに、e 個 $(e \leq d, e \leq k-L)$ をバースマークで置き換えることでサイズ増加量は $h(d-e)\frac{|S|}{L}$ となり、$d = e$ の時に最善で 0 とすることができる。

5.3 問題点

ソフトウェア P'_W から $k-L$ 個以下の分散情報しか抽出できない状況を考える。$k-L$ 個以下の分散情報からは秘密情報 S に関する情報が全く得ることができず、秘密情報 S は L 個の後付け情報 W' で任意の情報 S' に変えることが可能である。よって、保管した分散情報 W が後付けの情報 W' ではないと証明できる必要がある。$W \neq W'$ を証明する一つの方法として、保管する情報 W を信頼できる機関に預ける方法が挙げられる。

6 実験

3 章と 4 章で述べた提案手法が実際に実現可能な手法だと示し、性能を確認するために透かし埋め込みによる実行時間とサイズの増加量に関する実験を行った。

6.1 実験システム

実験システムは、透かし埋め込みと透かし抽出の 2 つの機能に分けることができる。実験システムの略図を図 4, 図 5 に示す。さらに、図 4 に示す透かし埋め込みシステムはバースマークを利用するか否かで動きが変わる。バースマークを利用する時の透かし埋め込みシステムは、乱数生成機能 RG の代わりに代用情報生成機能 RG' を呼び出す。図 6 に代用情報生成機能 RG' の略図を示す。

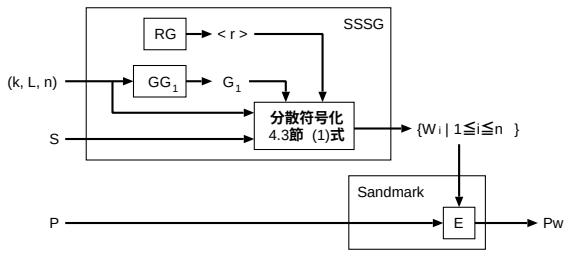

図 4 透かし埋め込みシステム

透かし埋め込み機能 E、透かし抽出機能 R、バースマーク抽出機能 BM は Sandmark [8] を用いる。ただし、バースマーク抽出機能 BM は bit 情報を出力するための変更

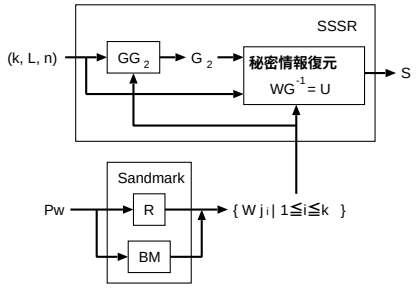

図 5 透かし抽出システム

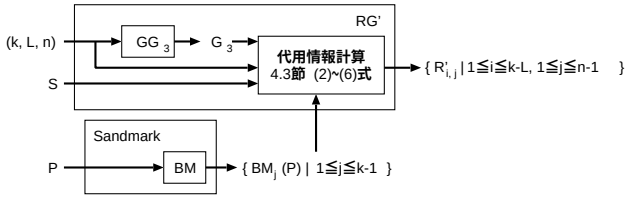

図 6 代用情報生成機能

を加えた。分散情報生成機能 SSSG と秘密情報復元機能 SSSR は Java 言語で作製した。また、生成行列生成機能 GG は状況に合わせて 3 種類の生成行列 G を生成する。

6.2 バースマークを変えない透かし埋め込み

1 つのクラスファイルに実際に透かしを埋め込み、埋め込み前と埋め込み後のバースマークの変化を調べた。透かし埋め込み手法として 8 種類、バースマーク抽出手法として 3 種類用いた。実験結果のうち、透かし埋め込みによってバースマークが変化した組合せを表 1 に示す。

表 1 透かし埋め込みによるバースマークの変化

Watermarks	Birthmarks		
	IS	SMC	UC
Register type	○	×	×
Monden	○	×	○
他 6 種類	○		

6.3 サイズ増加量

4 章で述べた手法を用いることで、透かしの埋め込み個数を減らすことができる。よって、透かし埋め込み個数を変化させた時のサイズ増加量を表 2 に示す。また、ランプ型秘密分散法を用いることで、透かし情報量を減らすことができる。よって、透かしの情報量を変化させた時のサイズ増加量を表 3 に示す。

6.4 実行時間増加量

100 万回実行されるブロック A と 1 回だけ実行されるブロック B があるソフトウェアを仮定し、ブロック A, B に分散情報を埋め込んだ。また、透かし埋め込み手法として DynamicPathBasedWatermark[3]、分散情報として 12bit の情報を用いた。結果として、ブロック A, B 両方に透かしを埋め込んだ場合と比べて、ブロック A をバースマークで補った場合は、34.1m 秒少なくなった。

表 2 埋め込み個数によるサイズの増加量 (byte)

Watermarks	埋め込み個数				
	1	2	3	4	5
Add Exp.	131	149	167	186	205
Add Ini.	131	172	200	228	256
Add M. and F.	243	312	413	489	565
Add Swi.	189	277	365	469	569

表 3 情報量によるサイズの増加量 (byte)

Watermarks	埋め込み情報量 (bit)				
	12	24	36	48	60
Add Exp.	129	130	135	139	139
Add Ini.	128	131	135	139	139
Add M. and F.	232	235	237	239	241
Add Swi.	144	174	204	234	264

7 おわりに

本研究では、透かし埋め込み cost を減らす手法として、秘密分散法とバースマークを用いる手法を提案し、実験によって性能を確かめた。

今後の課題として、4 章で述べたバースマークを変えない透かし埋め込み手法、5 章で述べた分散情報の保管方法のさらなる考察が必要である。また、提案手法に適したバースマークの抽出手法の考察も必要である。

参考文献

[1] C.Collberg, C.Thomborson, “Software Watermarking: Models and Dynamic Embedding,” POPL’99, pp.311-324, Jan. 1999.

[2] C.Collberg, C.Thomborson, “Watermarking, Tamper-Proofing, and Obfuscation - Tools for Software Protection,” IEEE Tr. on SE, Vol.28, No.8, pp.735-746, Aug. 2002.

[3] C.Collberg, E.Carter, S.Debray, A.Huntwork, C.Linn, M.Stepp, “Dynamic Path-Based Software Watermarking,” ACM SIGPLAN 2004.

[4] A.Shamir, “How to Share a Secret,” C.ACM, Vol.22, No.11, pp.612-613, 1979.

[5] G.Blakley, “Safeguarding Cryptographic Keys,” Proc.AFIPS, Vol.48, pp.313-317, 1979.

[6] 山本, “(k, L, n) しきい値秘密分散システム,” 電子通信学会論文誌, Vol.J68-A, No.9, pp.945-952, 1985.

[7] 藤井, 窪塚, 保坂, 多田, 加藤, “排他的論理和を用いた (k, n) しきい値法の構成法,” ISEC2007-5, pp.23-30.

[8] Sandmark, “A Tool for the Study of Software Protection,” http://sandmark.cs.arizona.edu/.