電子透かしにおける鍵管理問題に関する一考察*

3D − 9

富岡 淳樹　小川 宏　中村 高雄　高嶋 洋一†
NTT ヒューマンインタフェース研究所‡

1 はじめに

ディジタルコンテンツの流通拡大にともない、著作権保護技術として電子透かしが近年注目を集めている。本稿では電子透かし技術において stego object（透かし情報）[1] の埋め込みに用いる「鍵」の性質に着目し、ディジタルデータの冗長性と「鍵」の性質を利用した攻撃方法及び対処方法を論ずる。

2 電子透かしの invertible 性

一般に電子透かしにおいては stego object の埋め込み及び読み取りの際に「鍵」を必要とする。ここでいう「鍵」とは一定の権限を持った人間のみが埋め込んだ stego object の閲覧できること、または他人が stego object を改竄できないことを保証するためのものである。実装上「鍵」の意味するところは電子透かしのアルゴリズムによって異なってくるが、乱数生成の seed であるという点では一致しているとみてよい。例えば直接スペクトル拡散方式においては PN 系列を発生させるための seed であり, [2] などの frequency hopping 方式では stego object を埋め込む周波数帯域を選定するための乱数の seed である。

ここで注意しなければならないのは同じ「鍵」というメタファを用いているが、暗号系における「鍵」と電子透かしにおける「鍵」では性質が異なる点である。例えば、暗号系において平文 A から鍵 key を用いて暗号文 B を作成する $\mathrm{Crypt}(A, key) \rightarrow B$ という暗号化に対し、$\mathrm{Crypt}(C, key') \rightarrow B$ という暗号化を行なう別の鍵 key' を作ることは非常に困難である。何故ならたとえ C を生成したとしても、平文 A の冗長性の低さから C が意味のある文になる可能性は確率的にほとんど無いといえるからである。ところが、電子透かしにおいては事情が異なる。電子透かしの対象となる画像、音声などのディジタルデータの冗長性の高さから、C をつくり出すということは十分に可能である。この問題を論じているのが [3] や [4] であり、これらの論文ではこういった問題を抱える電子透かしを "invertible" な電子透かしと呼び、secure な hash 関数や DES を用いた "non-invertible" な電子透かしを提案している。

[3] や [4] では marked object を元に別の cover object を生成し、marked object に対する著作権を主張されてしまうという問題を論じているが、本稿ではこれとは別の視点から電子透かしの invertible 性について考察する。つまり、ディジタルデータの冗長性を利用し、正当に埋め込まれたものとは別の stego object を読み出せるような鍵を生成する攻撃方法について述べる。

3 鍵生成攻撃

アルゴリズムが公開されていれば別の stego object を取り出す鍵を求めることは求めることは理論上可能である。いくつかのケースについて以下に述べる。

3.1 CASE 1: frequency hopping 方式の場合

frequency hopping 方式は cover object を周波数変換して得られる周波数成分行列の特定の周波数成分を変更するというものである。その際にどの周波数成分を変更するかを疑似乱数系列を用いて決定する。鍵はその乱数系列の生成に用いられるというのは前述の通りである。情報の埋め込みは周波数成分値をある決まった幅で量子化することで 1 ビットの情報を埋め込むのが一般的である。逆に読み出す際は "0" と "1" のどちらの領域に周波数成分値が属しているかを判別して読み出すことになる。

電子透かしのアルゴリズムが公開されていれば—つまり "0" と "1" の割当てのルールがわかっていれば、そのルールを元に、正当に埋め込まれた stego object とは別の stego object を読み取るための鍵を生成することができる。つまり、$m \times m$ の周波数成分行列から n ビットの stego object を取り出したい場合、

```
b[n] : 取り出したい stego object のビット列
C[m][m] : 周波数成分行列
detect() : stego object 読取り関数 ("0"か"1"を返す)
rx[n] : 生成する乱数列
ry[n] : 生成する乱数列

for (i=0; i++; i<n) {
  for (j=0; j++; j<m) {
    for (K=0; k++; k<m) {
      if ((detect(C[j][k]) == b[i])
            && C[j][k] != FALSE) {
        rx[i] = j; ry[i] = k;
        C[j][k] = FALSE;
        goto Next;
      }
    }
  }
  Next:
}
```

というアルゴリズムで埋め込み対象となる周波数帯域の順列を生成することができ、その順列を生成する seed が求める鍵となる。一般に乱数生成アルゴリズムは一方向性が無いため、得られた乱数値から seed を求めることは可能である。例えば n 段の M 系列であれば n+1 個の乱数値があれば seed を求めることができる。

3.2 CASE 2: 直接スペクトル拡散方式の場合

[5] や [6] では seed を 1 から 1000 まで廻して相関を取り、ピークを見つけている（図 1 左）。しかし、[6] と同様のアルゴリズムを実装し、256×256 の 256 階調のグレイスケール画像 "Lena" に対して stego object を埋め込み、marked object に対して 1 から 10000000 まで[2]seed を廻して相関を取る実験をした結果、他の値でも threshold を越えるピークが立ち得ることがわかった（図 1 右）。この場合、得られる stego object は当然無意味なものになるが、stego object に適当な暗号化や変調を施したあとで埋め込んだのだと主張されれば、あらかじめ特別な手段を講じていない限り、それを否定するのは困難であろう。

4 問題の分類

4.1 stego object の読み取りにおける cover object の利用／非利用

問題となるのは stego object の読み出しに cover object を用いない場合である。何故なら [3] で証明されているように、cover

*A consideration of the key-management probrem in digital watermarking

†Atsuki Tomioka, Hiroshi Ogawa, Takao Nakamura, Youichi Takashima
‡NTT Human Interface Laboratories

[1]本稿では [1] にならい、透かし情報を埋め込む前の原データを cover object, 埋め込まれる透かし情報を stego object, 透かし情報を埋め込まれた後のデータを marked object と呼ぶ。

[2]鍵長が 32 ビットだとすれば、これでも全鍵空間の 0.2%を調べたに過ぎない。

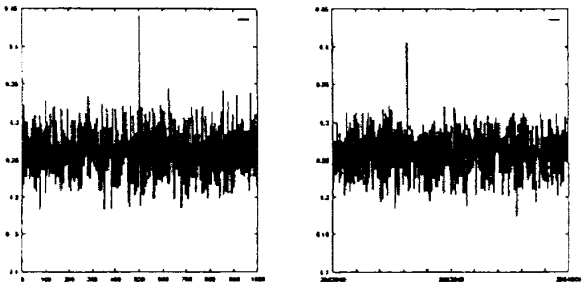

図 1: seed=500 と seed=2653315 のピーク

object を用いない場合は本質的に invertible だからである。つまり cover object のどの領域を操作したかを判別できないために marked object と cover object との対応が取れず、著作権の詐称を許してしまうことになるのである。これは鍵生成攻撃の場合でも同じことである。

4.2 鍵と stego object の結合度

鍵の偽造可能性を評価するために、電子透かしの利用形態による分類をしてみる。ここでは分類の基準として stego object の数と利用可能な鍵の数を用いることにする（表 1）[3]。つまり、表 1 は stego object に対して鍵をどれだけ割り当てるかについての分類である。

表 1: 鍵と stego object による分類

	cover obj.	stego obj.	鍵
type1	任意	1	1
type2	任意	任意	1
type3	任意	1	任意
type4	任意	任意	任意

type1 を考えてみる。この場合は 1 つの stego object に対してただ 1 つの鍵が割り当てられている。これは例えば著作者が自らのコンテンツに著作者 ID のような著作者毎の情報を著作者固有の鍵で埋め込む場合が考えられよう。stego object と鍵は 1 対 1 で厳密に対応づけられる。type2 は著作物の名前のような著作物毎の情報を著作者が著作者固有の鍵で埋め込むような場合であり、type3 は著作者が著作者毎の情報を著作物毎に異なる鍵で埋め込む場合である。type4 は秘匿通信が代表例となろう。ユーザは任意のコンテンツに任意の情報を任意の鍵で埋め込むことになる。

ここで鍵と stego object の結合度という概念を導入する。ここで 結合度 = (1 つの stegoobject に対する利用可能な鍵の数) である。例えば type1 の場合はユーザは任意の stego object に対してただ 1 つの鍵を用いることになるため、結合度は非常に高い。対して、type4 の場合は任意の stego object に対して任意の鍵を用いることになるので結合度は非常に弱い。つまり、結合度はある stego object に対する鍵の選択範囲の広さを表している。

結合度は鍵空間における鍵選択の自由度を表しているため、結合度が弱ければ鍵生成攻撃に対して弱いということがいえる。しかし、結合度が高いということは結託攻撃に対して弱いということに注意しなければならない。これを防ぐためには後述のように、鍵の自由度を保ちながら stego object と鍵を hash 関数などで結合する、あるいは cover object ごとに透かしの埋め込み位置を変えるなどの方法を取ればよい。

[3]cover object が 1 の場合は実際の運用を考えた場合実用的ではないため、表 1 では除外している。

5 鍵生成攻撃に対する対策

5.1 stego object の多重化

もっとも単純で効果的なのは stego object の多重化である。1 つの cover object に stego object を繰り返し埋め込み、stego object の読み取り時に多数決などの判定法を用いることで、乱数列の生成や高い相関を持つ seed を見つけることが飛躍的に困難になる。

5.2 object、ユーザ、鍵の binding

鍵が正当なものであるか、つまり、ある marked object に stego object を埋め込む際に確かにその鍵が用いられたことを証明する仕組みがあればよい。鍵や cover object との binding を考慮したものに [7] がある。

また、以下のように鍵空間 K の部分空間 L への写像を得、その部分空間 L に属する鍵のみを用いる、という方法もあるだろう：

鍵空間：K, 部分空間：$L \subset K$, cover object：o

$o \overset{f}{\hookrightarrow} l$ ただし $l \in L$

f は例えば secure な hash 関数など。

こうすることで、たとえ cover object 毎に鍵を変えるような場合でも鍵と cover object との binding を得ることができ、鍵生成攻撃に対する耐性を高めることができる。

鍵空間を限定することは鍵の推測を容易にし、電子透かしのセキュリティ的な強度の低下につながるので注意が必要である。電子透かしの用途や使用状況に応じて鍵空間の大きさと結合度のバランスを取ることが必要となる。

6 おわりに

電子透かしにおける鍵の特性について考察し、その性質を利用した攻撃方法である鍵生成攻撃、およびそれに対する対策を提案した。電子透かしの実利用を考えた場合、埋め込み／読み取りアルゴリズムだけでなく、こうした運用上の問題にまで踏み込んだ議論は今後ますます必要になってくるであろう。

参考文献

[1] B. Pfitzmann. Information hiding terminology. In Ross J. Anderson, editor, *Information hiding: first international workshop*, Vol. 1174 of *Lecture notes in computer science*, pp. 347–350, Berlin, Germany, May 1996. University of Cambridge, Isaac Newton Institute, Springer Verlag. Results of an informal plenary meeting and additional proposals.

[2] 中村高雄, 小川宏, 高嶋洋一. 画質を考慮した電子透かし. 電子情報通信学会 1997 年ソサイエティ大会 SA-7-3.

[3] L. Qiao and K. Nahrstedt. Watermarking schemes and protocols for protecting rightful ownership and customer's rights, 1998. submitted to Academic Press Journal of Visual Communication and Image Representation.

[4] S. Craver, N. Memon, B. Yeo, and M. Yeung. Can invisible watermarks resolve rightful ownerships? Technical Report RC 20509, IBM Research Division, July 1996.

[5] I. J. Cox, J. Kilian, T. Leighton, and T. Shamoon. Secure spread spectrum watermarking for multimedia. Technical Report 95-10, NEC Research Institute, Princeton, NJ, 1995.

[6] A. Piva, M. Barni, F. Bartolini, and V. Cappellini. DCT-based watermark recovering without resorting to the uncorrupted original image. In *Proceedings of 4th IEEE International Conference on Image Processing ICIP'97*, Vol. I, pp. 520–523, Santa Barbara, CA, USA, October 26-29 1997.

[7] J. Fridrich, A. C. Baldoza, and R. J. Simard. Robust digital watermarking based on key-dependent basis function. a paper for the 2nd Information Hiding Workshop in Portland, OR, April 1998.