데이터의 삽입과 무결성이 보장되는 워터마킹

윤호빈[O] 박근수
서울대학교 컴퓨터공학부
{hbyoon, kpark}@theory.snu.ac.kr

A Watermark for Data Embedding and Image Verification

Hobin Yoon[O] Kunsoo Park
School of Computer Science and Engineering, Seoul National University

요 약

Fragile 워터마킹은 이미지의 무결성을 보장하기 위하여 원본 이미지에 사람이 지각할 수 없는 데이터를 삽입하는 것을 말한다. 본 논문은 이진 데이터의 삽입이 가능하며, 원본 이미지와 삽입된 데이터의 무결성이 보장되는 fragile 워터마킹의 한 방법을 제시한다. 제시된 방법은 hash 함수와 PRBG(pseudo random bit generator)를 이용한 one-time pad를 사용하며, 한 pixel당 약 2.8125bits의 정보를 저장할 수 있다.

1. 서론

디지털 멀티미디어 시대의 도래로 누구나 쉽게 이미지를 생성하고 배포할 수 있게 되었으나, 한편으로는 원본이미지의 불법적인 복사 또는 편집도 또한 용이하게 되었다. 이에 이미지의 저작권 보호와 인증이 새로운 문제로 떠올랐으며, 이의 해결책으로 워터마킹이 제시되었다. 워터마킹은 이미지 데이터에 사람이 지각할 수 없는 데이터를 삽입하는 것을 말한다. 워터마킹의 목적에는 저작권 보호, fingerprinting, 내용에 대한 인증, 내용에 대한 설명삽입, 복사의 제한, 내용의 보호 등이 있다. 이 중 fragile 워터마킹은 이미지의 인증을 위한 목적으로 쓰이며, robust 워터마킹은 저작권의 보호를 위하여 쓰인다.[3]

Fragile 워터마크는 원본이미지가 변경되면 높은 확률로 변경되거나 파괴되므로 이미지에 대한 인증의 용도로 사용된다. Fragile 워터마크가 사용되는 예는 믿을 수 있는 카메라(trustworthy camera)[1], 법정의 증거로 제출되는 이미지, 신문잡지의 사진, 첩보활동에 사용되는 이미지, 상업적인 용도에 관련된 이미지에 유용하게 사용된다. Fragile 워터마크는 공격자에 의해 쉽게 제거될 수 있으므로, 이미지의 저작권보호를 위한 목적으로 쓰이지는 않는다.

Fragile 워터마킹의 중요한 방법에는 다음과 같은 것들이 있다. Matsui와 Tanaka는 gray scale의 원본 이미지에서 인접한 pixel간의 차이를 바탕으로 메시지를 삽입하는 방법을 제안하였다. 원본 이미지의 i번째 pixel의 값을 x_i라고 하고, 인접한 pixel간의 차를 $\Delta_i = x_i - x_{i-1}$라고 한다. 삽입하고자 하는 이진 메시지를 c_i라고 한다. Δ_i에 따라 c_i의 값이 정해지며, 이는 워터마크를 삽입하는 사람과 추출하는 사람 사이에 약속된 cipher key table에 의해 결정된다. c_i가 원하는 값이 아니면, c_i가 원하는 값이 되게끔 Δ_i를 바꾸며, 이 때 원래의 Δ_i의 값과 바뀐 Δ_i의 값의 차가 최소가 되도록 한다. Matsui의 방법은 511bits의 cipher key table만 있으면 임의의 길이의 이미지를 encoding가능하다.[2]

Yeung과 Mintzer는 워터마크를 삽입하는 사람과 추출하는 사람 사이에 미리 약속된 random sequence를 이용한 encoding방법을 제안하였다. Extraction function을 이용해 각 pixel로부터 random sequence의 index를 얻고, 이 index로 random sequence에서 메시지를 선택하는 방법이다. Matsui의 방법과 마찬가지로 선택되어진 값이 원하는 값과 다를 때는, 원하는 값이 선택될 수 있도록 원본 이미지를 약간 변형시킨다. 이 방법은 특정 pattern을 가진 워터마크를 삽입하며, 이미지에 가해진 변화를 pixel단위로 찾아낼 수 있다.[8]

Wong은 hash function을 이용한 encoding방법을 제시하였다. 원본 이미지를 8×8등의 적당한 크기의 블록으로 나눈 다음 블록별로 hash를 한다. hash함수의 입력으로 이미지에서 LSB를 제외한 bit들, 이미지 크기의 정보, 이미지 저자의 비밀키를 연결한 것을 넣고, hash함수의 결과를 LSB에 저장한다. 특정 pattern을 가진 워터마크를 삽입하며, 이미지에 가해진 변화를 블록 단위로 탐지할 수 있다. 또한 이미지의 scaling이나 cropping과 같은 크기 변화도 탐지할 수 있다.[7]

Wong은 위의 방법을 공개키 암호시스템을 사용한 방법으로 확장하였다. 블록별로 나뉘어진 이미지의 LSB를 제외한 bit들을 모두 모아서 이미지 크기에 대한 정보와 함께 hash한다. hash된 결과는 블록내의 LSB의 수와 같다. 이 hash값을 저자의 비밀키를 이용하여 암호화하고, 이를 삽입하고자 하는 binary 워터마크와 xor하여 LSB에 저장한다. 워터마크를 추출할 때는 삽입할 때와 같은 방법으로 얻어진 hash값을 저자의 public key로 복호화한 다음, LSB와 xor하면 된다.[6]

Matsui와 Tanaka의 방법은 이진 데이터를 삽입할 수는

있으나, 이미지의 무결성을 보장할 수는 없으며, Yeung과 Mintzer의 방법과 Wong의 두 가지 방법은 원본 이미지에 가해진 변화를 탐지할 수는 있으나, 이진 데이터를 저장할 수 없는 단점이 있다. 즉 기존의 방법들은 이진 데이터의 삽입 또는 이미지의 무결성 보장은 가능했으나, 이 둘을 동시에 만족시키는 것은 없었다.

본 논문에서는 이진 데이터의 삽입과, 원본 이미지와 삽입된 데이터에 대한 무결성 보장이 동시에 가능한 방법을 제시한다.

2. 알고리즘 기술

2.1. 워터마킹 삽입

원본 이미지 I는 총 n개의 pixel로 이루어진 1차원 배열이다. 한 pixel x_i는 24bits로 이루어져 있다.

$$I=\{x_i \mid x_i \in \{0,1,\cdots,2^{24}-1\},\ 0\le i<n\}$$

원본 이미지에 삽입하고자 하는 정보를 A라고 한다. A는 총 길이 l의 이진 메시지이다. 이것은 이미지의 저자, 저작권정보, 또는 정보를 삽입하는 사람이 추출하는 사람에게 전달하고자 하는 다른 정보일 수도 있다.

$$A=\{a_i \mid a_i \in \{0,1\},\ 0\le i<l\}$$

2.1.1. 정보 삽입

n개의 pixel로 이루어진 원본 이미지를 그림1 과 같이 64개의 pixel로 이루어진 블록으로 나눈다.

$$n=64k+t\ \ (0\le t\le 63)$$

마지막 블록을 제외한 모든 블록은 64개의 pixel로 이루어져 있으며, 마지막 블록은 $64+t$개의 pixel로 이루어져 있다.

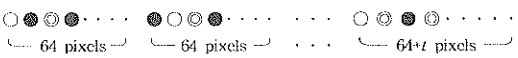

그림 1

하나의 pixel은 8bits씩 Green, Blue, Red의 색소를 나타내므로 총 24bits이다. 한 이미지는 $24n$개의 bit으로 이루어지며, 마지막 블록을 제외한 하나의 블록은 1536개의 bit으로 이루어지며(64pixels×24bits=1536bits), 마지막 블록은 $1536+24t$개의 bit으로 이루어져 있다.

각 bit을 $b_i(0\le i<24n)$라고 하고, 각 색소의 LSB(Least Significant Bit)를 l_i라고 하자.

$$l_i=b_{8i+7}\ \ (0\le i<3n)$$

각 블록의 마지막 4개의 pixel을 제외한 pixel의 LSB에 삽입하고자 하는 정보 a_i를 다음과 같이 저장한다. (그림 2)

$$\begin{cases} l_{192(j-1)+i}=a_{180(j-1)+i} & (0\le i<180) \quad \text{if } 1\le j<k \\ l_{192(j-1)+i}=a_{180(j-1)+i} & (0\le i<180+3t) \quad \text{if } j=k \end{cases}$$

2.1.2. 인증 정보 삽입

- **$j(1\le j<k)$번째 블록의 경우**

위의 정보 삽입 단계를 거치면, $b_{1536(j-1)+i}(0\le i<1536)$ 중 $b_{1536(j-1)+1447+8i}(0\le i<12)$를 제외한 모든 bit은 정의된다. 이들을 연결시킨 것을 M_j라고 하자.

$$\begin{aligned} M_j=b_{1536(j-1)+0}\,\|\cdots\|\,b_{1536(j-1)+1446}\,\|\,b_{1536(j-1)+1448}\,\|\cdots \\ \|\,b_{1536(j-1)+1454}\,\|\,b_{1536(j-1)+1456}\,\|\cdots \\ \|\,b_{1536(j-1)+1462}\,\|\,b_{1536(j-1)+1464}\,\|\cdots \\ \|\,b_{1536(j-1)+1470}\,\|\,b_{1536(j-1)+1472}\,\|\cdots \\ \|\,b_{1536(j-1)+1478}\,\|\,b_{1536(j-1)+1480}\,\|\cdots \\ \|\,b_{1536(j-1)+1486}\,\|\,b_{1536(j-1)+1488}\,\|\cdots \\ \|\,b_{1536(j-1)+1494}\,\|\,b_{1536(j-1)+1496}\,\|\cdots \\ \|\,b_{1536(j-1)+1502}\,\|\,b_{1536(j-1)+1503}\,\|\cdots \\ \|\,b_{1536(j-1)+1510}\,\|\,b_{1536(j-1)+1512}\,\|\cdots \\ \|\,b_{1536(j-1)+1518}\,\|\,b_{1536(j-1)+1520}\,\|\cdots \\ \|\,b_{1536(j-1)+1526}\,\|\,b_{1536(j-1)+1528}\,\|\cdots \\ \|\,b_{1536(j-1)+1534} \end{aligned}$$

M_j의 bit 수는 $|M_j|=64\times 24-12=1524$이다. 이를 MD5를 이용한 hash알고리즘으로 hash하여 128bits의 output H_j를 얻는다.[4] H_j의 각 bit을 $h_{j,i}(0\le i<127)$라 한다.

$$H_j=\text{hash}(M_j)$$

H_j중에서 11의 배수인 bit을 $l_{192(j-1)+180+i}(0\le i<12)$에 대입한다. (그림 3)

$$l_{192(j-1)+180+i}=h_{j,11i}\ \ (0\le i<12)$$

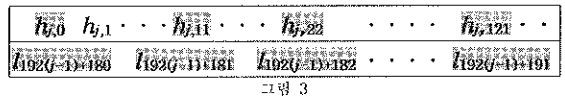

그림 3

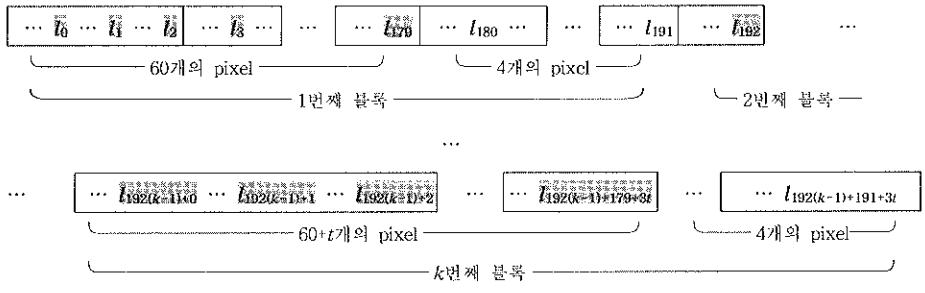

그림 2

• **k번째 블록의 경우**

M_k는 $b_{1536(k-1)+i}(0 \le i < 1536+24t)$중 $b_{1536(k-1)+1447+24t+8i}$ $(0 \le i < 12)$를 제외한 모든 bit을 연결시킨 것이다. $|M_k| = 1524+24t$이다. 이를 위와 같은 방법으로 hash하여 H_k를 만들고, 마지막 4개의 pixel의 LSB들에 대입한다.

$$l_{192(k-1)+180+3t+i} = h_{k,11i} \quad (0 \le i < 12)$$

2.1.3. 암호화

$l_i (0 \le i < 3n)$을 암호화하기 위해서 비밀키 암호시스템의 일종인 one-time pad를 사용한다. One-time pad를 위해 사용되는 비밀키는 s이며 이는 워터마크를 삽입하는 사람과 추출하는 사람이 공유한다. 이 s로부터 RSA generator나 Blum-Blum-Shub generator등의 PRBG(Pseudo Random Bit Generator)를 사용하여 이진 stream $s_i (s_i \in \{0,1\}, 0 \le i < 3n)$를 계산한 후[5], 각 l_i에 s_i를 xor(exclusive-or)한다.

$$l_i' = l_i \oplus s_i \quad (0 \le i < 3n)$$

2.2. 워터마킹 추출

2.2.1. 복호화

비밀키 s로부터 이진 stream $s_i (s_i \in \{0,1\}, 0 \le i < 3n)$를 계산한다. 이를 LSB와 xor하여 복호화된 l_i를 얻어낸다.

2.2.2. 인증 정보 추출

각 블록의 마지막 4개의 pixel의 LSB들의 제외한 bit들을 연결시킨 다음 이를 hash한다. 이 hash로부터 얻어진 결과와 마지막 4개의 pixel의 LSB들과 비교한다. 만약 두 정보가 일치하지 않으면, 이미지에 변경이 가해진 것이므로 그 블록은 인증되지 못한다.

2.2.3. 정보 추출

2.2.2에서 인증된 블록들의 정보는 유효한 것이다. 이 블록들의 마지막 4개의 pixel을 제외한 LSB들을 연결시켜 원래의 정보 A를 재구성한다.

2.3. 저장되는 정보 A의 길이(l)

총 블록의 개수는 $\lfloor \frac{n}{64} \rfloor$이며, 마지막 블록을 제외한 각 블록은 180bits의 정보를 저장한다. 마지막 블록에는 $180+(n \bmod 64)\times 3$bits의 정보가 저장된다. 따라서 저장되는 정보 A의 길이 l은

$$l = \left\{\left(\lfloor \frac{n}{64} \rfloor - 1\right)\times 180\right\} + \{180+(n \bmod 64)\times 3\}$$
$$= \lfloor \frac{n}{64} \rfloor \times 180 + (n \bmod 64)\times 3$$

이다. 따라서 하나의 pixel당 약 2.8125bits의 정보를 저장할 수 있다.

2.4. 알고리즘 분석

Hash함수는 원본 데이터 I와 삽입된 데이터 A의 무결성 보장을 위해서 사용된다. 한 블록의 인증을 위하여 12bits의 정보를 사용한다. Hash함수의 output이 고르게 나온다는 가정 하에서, 공격자가 한 블록의 이미지를 바꾸었을 때, 탐지될 확률은 $99.98\%(=1-2^{-12})$이다. Wong은 한 블록의 데이터를 인증하기 위해서 192bits의 LSB를 모두 사용하였다. 이 경우 공격자가 한 블록의 이미지를 바꾸었을 때, 탐지될 확률은 거의 $100\%(=1-2^{-192})$이다. Wong의 방법과 비교했을 때, 본 논문에서 제시하는 방법은 원본 이미지의 무결성을 약간 희생하는 대신, 블록당 180bits의 데이터를 삽입할 수 있었다. 인증정보의 양과 삽입하고자 하는 데이터의 양은 각각의 응용에 따라 적절히 조절될 수 있을 것이다.

암호화는 삽입된 데이터와 인증정보를 보호하기 위한 목적으로 사용된다. One-time pad는 안전성이 보장되나 실용적으로 쓰이기에는 키의 길이가 너무 길다. 따라서 작은 길이의 키 s를 이용하여 긴 길이의 bit stream을 생성하는 PRBG를 사용하였다. 암호화에 사용된 알고리즘의 안전도는 사용된 PRBG의 안전도에 달려 있으며, 사용된 PRBG는 안전함이 알려져 있다.[5]

3. 결론

본 논문은 이진 데이터의 삽입이 가능하며, 원본 데이터와 삽입된 데이터의 인증이 가능한 fragile 워터마킹의 한 방법을 제시하였다. 한 블록 내에서 공격자가 원본 데이터를 바꾸었을 때, 탐지될 확률은 99.98%이며, 하나의 pixel당 저장할 수 있는 데이터는 약 2.8125bits이다.

본 논문에서는 암호화를 위해 비밀키 암호시스템인 one-time pad를 사용하였다. 이를 공개키 암호시스템을 사용한 version으로 바꿀 수도 있다. 즉, 데이터의 삽입과 인증 정보의 삽입이 끝난 다음 모든 LSB들을 RSA나 ECC와 같은 공개키 암호시스템을 사용하여 암호화할 수 있다. 이렇게 할 경우 인증시 이미지 저자의 비밀키가 노출되지 않으며, 저자는 같은 비밀키를 여러 번 사용할 수 있는 장점이 있다. 실제의 구현은 진행중이다.

참고문헌

[1] G. L. Friedman, "The trustworthy digital camera: Restoring credibility to the photographic image", IEEE Transactions on Consumer Electronics, vol.39, pp.93-103, Nov. 1993

[2] K. Matsui and K. Tanaka, Video-steganography: How to embed a signature in a picture, Proc. IMA Intellectual Property, Jan. 1994, vol. 1, no. 1, pp. 187-206

[3] N. Memon, P. W. Wong. Protecting digital media content, CACM, July 1998, vol.41, no.7, pp.34-43.

[4] R. L. Rivest, The MD5 message digest algorithm, Internet RFC 1321, April 1992

[5] D. R. Stinson, Cryptography: Theory and practice, CRC Press, 1995

[6] P. W. Wong, A public key watermark for image verification and authentication, Proceedings 1998 International Conference on Image Processing. ICIP98, IEEE Computing Society, 1998, pp.455-9, vol.1, Los Alamitos, CA, USA.

[7] P. W. Wong, A watermark for image integrity and ownership verification, Proceedings of IS&T's PICS Conference, *51st Annual Conference. Society of Imaging Science and Technology* 1998, pp.374-9, Springfield, VA, USA.

[8] M. Yeung, F. Mintzer, An invisible watermarking technique for image verification, Proceedings of International Conference on Image Processing, IEEE Computing Society, 1997, pp.680-3, vol. 2, Los Alamitos, CA, USA.