A Secure Authentication Watermarking for Halftone and Binary Images

Hae Yong Kim, Amir Afif
Universidade de São Paulo, Escola Politécnica, Av. Prof. Luciano Gualberto, trav. 3, 158, CEP 05508-900, São Paulo, SP, Brazil

ABSTRACT: An authentication watermark is a hidden data inserted into an image that can be used to detect any accidental or malicious alteration in the image. Many authentication-watermarking techniques for continuous-tone images are described in the literature, but only a quite small number of secure authentication watermarking techniques are available for binary/halftone images. This article proposes a simple solution for inserting a secure authentication watermark in binary/halftone images. It consists of choosing a set of pseudo-random pixels in the image, clearing them, computing the message authentication code (or the digital signature) of the now-cleared image, and inserting the resulting code into the selected random pixels. Dispersed-dot halftone images watermarked by the proposed technique present better visual quality than do watermarked generic binary images. However, in practice, the visual degradation is hardly noticeable in either case. The proposed technique seems to be the only binary/halftone watermarking scheme that can detect even a single pixel alteration in the host image. It can be used with secret-key or public-key ciphers. © 2004 Wiley Periodicals, Inc. Int J Imaging Syst Technol, 14, 147–152, 2004; Published online in Wiley InterScience (www.interscience.wiley.com). DOI 10.1002/ima.20018

Key words: authentication watermarking; fragile watermarking; halftoning; binary image

I. INTRODUCTION

A watermark is a signal added to digital data (audio, video, and still images) that can be extracted later to make an assertion about the data. In this article, we are concerned only with still images. Digital watermarking techniques can be classified as either “robust” or “fragile.” Robust watermarks are designed to be hard to remove and to resist common image-manipulation procedures. They are useful for copyright and ownership assertion purposes. Fragile watermarks (or authentication watermarks) are easily corrupted by any image-processing procedure. Watermarks for checking the image integrity can be fragile because if the watermark is removed, the watermark detection algorithm will correctly report the corruption of the image.

Halftoning is a process used to convert a grayscale image \( G \) into the corresponding binary image \( B \), such that \( B \) looks like \( G \) when viewed from a distance. Classic halftone methods include ordered dithering, error diffusion (Ulichney, 1987), and dot diffusion (Knuth, 1987). Halftone images appear routinely in books, magazines, newspapers, printer outputs, and fax documents. They can be dispersed-dot or clustered-dot. Usually, dispersed-dot halftone images present a better visual quality, but some devices cannot reproduce too-finely-dispersed dots (like laser printers) and they must use clustered-dot halftoning.

Many authentication-watermarking techniques for continuous-tone images are described in the literature (Barreto and Kim, 1999; Barreto et al., 2002; Hel-Or, 2001; Maxemchuk and Low, 1997; Pei and Guo, 1998; Chen et al., 2000; Chun and Ha, 2003; Fu and Au, 2000, 2002a; Hel-Or, 2001; Maxemchuk and Low, 1997; Pei and Guo, 2003; Wu et al., 2000), we have no knowledge of a cryptography-based authentication watermarking for binary and halftone images. Fu and Au (2002b) present a watermarking technique to detect unintentional changes in halftone images, but this cannot be considered to be an authentication watermarking because it does not withstand intentional or malicious attacks.

For binary and halftone images, this idea fails completely, because each pixel has only one bit. Modifying any pixel in order to embed watermarking modifies the fingerprint of the image and invalidates the watermarking. Consequently, although there are many papers on data hiding in binary images (Baharav and Shaked, 1998; Chen et al., 2000; Chun and Ha, 2003; Fu and Au, 2000, 2002a; Hel-Or, 2001; Maxemchuk and Low, 1997; Pei and Guo, 2003; Wu et al., 2000), we have no knowledge of a cryptography-based authentication watermarking for binary and halftone images. Fu and Au (2002b) present a watermarking technique to detect unintentional changes in halftone images, but this cannot be considered to be an authentication watermarking because it does not withstand intentional or malicious attacks.

After the well-accepted cryptography paradigm, the security of an authentication watermarking must lie only on the secrecy of the key. The fact that the image is watermarked as well as the water-
marking algorithm may be public without compromising the security of the scheme. In this article, we propose an authentication watermarking for half-tone and binary images that complies with this requirement. The idea is quite simple; however, it seems to be a new idea [we published it in an earlier conference paper: Kim and Afif, (2003)]. The main idea consists in clearing all those pixels of the host image that can be modified by the insertion of the watermark, before computing the MAC/DS. More precisely, it consists in choosing a set of pseudo-random pixels in the image, clearing them, computing the MAC/DS of the now-cleared image, and inserting the resulting code into the selected random pixels.

Dispersed-dot halftone images watermarked with this technique present better visual quality than do watermarked generic binary images (including clustered-dot halftone images). However, in practice, the visual degradation is hardly noticeable in either case. The size of MAC/DS is independent of the size of the host image. Consequently, the smaller the host image, the more visually noticeably will be the watermark insertion.

The proposed technique can be used in conjunction with a symmetric cipher (secret-key cryptography) or an asymmetric cipher (public-key cryptography) to detect any alteration of the watermarked image, even a single pixel toggling. The technique is intended to authenticate only digital binary images. Printed images cannot be authenticated using the proposed technique.

Friedman (1993) introduced the concept of the “trustworthy digital camera.” This concept can be straightforwardly adapted to fax machines, using the proposed technique. Using a “trustworthy fax,” the receiver of a document can certify the originator of the document, and that it has not been altered (accidentally or maliciously) during transmission.

II. DATA HIDING IN BINARY AND HALFTONE IMAGES

Many papers in the literature describe methods for inserting a sequence of bits in binary and halftone images. They can be divided into three basic classes:

1. Pixel-wise: Change the values of (usually pseudo-randomly chosen) individual pixels (Chun and Ha, 2003; Fu and Au, 2000, 2002a).

2. Component-wise: Change the characteristics of pixel groups—e.g., the thickness of strokes, the position or the area of connected components, etc. (Maxemchuk and Low, 1997). Unfortunately, the success of this approach depends highly on the type of the host image.

3. Block-wise: Divide the host image into blocks and modify some characteristic of each block. Some authors (Baharav and Shaked, 1998; Hel-Or, 2001; Pei and Guo, 2003) suggest alternating between two different dithering matrices to halftone an image such that the matrix used to halftone each block can be determined in the future by analyzing the statistical properties. Other works suggest modifying slightly the content of the block so that it hides the desired sequence of bits (Tseng et al., 2002; Wu et al., 2000).

DHST (data hiding by self-toggling) is a pixel-wise data hiding technique that is especially interesting for its simplicity (Fu and Au, 2000, 2002a). It was originally designed to embed bits in dispersed-dot halftone images. In DHST, a pseudo-random number generator with a known seed is used to generate a set of nonrepeating pseudo-random locations within the image. Then one bit is embedded in each location by forcing it to be either black or white. With 50% probability, the pixel of the original halftone is the desired value and thus no change is needed. And with 50% probability the pixel is opposite to that of the desired value, and the pixel needs to be altered. To read the hidden data, one simply uses the same random-number generator and the same seed to obtain again the nonrepeating pseudo-random locations. Then the pixel values at those locations can be read easily. Evidently, DHST can also be used to embed data in any binary image. However, in this case, a visible salt-and-pepper noise will appear. Here, we will transform DHST into a cryptographically secure fragile authentication watermarking. Fu and Au (2000, 2002a) present many improvements to this basic idea to enhance the visual quality of the host image: DHPT (data hiding by pair toggling) and DHSPT (data hiding by smart pair toggling). The underlying idea of these improvements is to keep constant the local average intensity. At the selected pseudo-random locations, the pixel alteration is accompanied by a complementary modification of a neighbor.

III. AUTHENTICATION WATERMARK FOR CONTINUOUS-TONE IMAGES

In the literature, most works on authentication watermarking are designed for continuous-tone images. Their goal is not only to detect alteration in the host image but also to spatially locate them. Holliman and Memon (2000) and Barreto et al. (2002) have analyzed thoroughly the requirements to obtain a reliable alteration-locating authentication-watermarking technique. However, in this article, we will simply detect the alteration, without locating it. The reason for this simplification is that binary images present a data hiding capacity much smaller than continuous-tone images.

A cryptography-based authentication watermarking (Barreto et al., 2002; Holliman and Memon, 2000; Wong, 1998) typically performs the following operations for the watermark insertion:

1. Let $Z$ be a grayscale image to be watermarked and let $A$ be a logo binary image to be inserted into $Z$.
2. Let $Z^*$ be the image obtained from $Z$ by clearing the LSBs of all pixels. Using a cryptographically secure hashing function $H$, compute the image fingerprint $H = H(Z^*)$.
3. Exclusive-or $H$ with $A$, getting the marked fingerprint $\hat{H}$.
4. Encrypt $\hat{H}$ with the secret key (symmetric cipher) or private key (asymmetric cipher), thus generating a MAC/DS $S$.
5. Insert $S$ into the LSBs of $Z^*$, obtaining the marked image $Z^*$.

The watermark-verifying algorithm is as follows:

1. Let $X'$ be a watermarked image. Let $X^*$ be the image obtained from $X'$ by clearing the LSBs of all pixels. Using the same hashing function $H$ chosen for the insertion, compute the image fingerprint $H = H(X^*)$.
2. Extract the LSBs from $X'$ and decrypt the result using the secret key (symmetric cipher) or public key (asymmetric cipher), obtaining the decrypted data $D$.
3. Exclusive-or $H$ with $D$, obtaining the check image $C$.
4. If $C$ and $A$ are equal, the watermark is verified. Otherwise, the marked image $X'$ has been modified.

Notice that, theoretically, the image $A$ must be publicly available for the verification to take place. In practice, however, $A$ is a meaningful logo image and any change in $X'$ will most likely
generate a noiselike image C, which cannot be mixed up with A, even if A is not publicly available. Moreover, A can be a very simple image, like an entirely white or black image, without compromising the security of the scheme.

IV. THE PROPOSED METHOD

As we said in Section I, for binary/halftone images, the insertion of the watermark changes the host binary image B and consequently changes its fingerprint. That is, \( H(B) \neq H(B') \). How can we overcome this problem?

We propose a very simple solution using the DHST scheme. Differently from most binary image data hiding techniques, in DHST only a few pixels are modified and the precise positions of those pixels are known both in the insertion and extraction phases. Consequently, these pixels can be cleared before computing the hashing function, just like clearing LSBs for grayscale images. Let us call the obtained technique AWST (authentication watermarking by self toggling). The AWST insertion algorithm is as follows:

1. Let \( B \) be a binary image to be watermarked and let \( A \) be a logo binary image to be inserted into \( B \).
2. Use a pseudo-random number generator with a known seed to generate a set of nonrepeating pseudo-random locations \( L \) within the image \( B \).
3. Clear all pixels of \( B \) that belong to \( L \), obtaining \( B^* \).
4. Compute the fingerprint \( H = H(B^*) \).
5. Exclusive-or \( H \) with \( A \), getting the marked fingerprint \( \hat{H} \).
6. Encrypt \( \hat{H} \) with the secret key (symmetric cipher) or private key (asymmetric cipher), thus generating the MAC/DS \( S \).
7. Insert \( S \) into the set of pixels \( L \), generating the watermarked image \( B' \).

The AWST extraction algorithm is as follows:

1. Let \( X' \) be a watermarked image. Use the same pseudo-random number generator and the same seed to generate again the same set of nonrepeating pseudo-random locations \( L \) where the watermark has been inserted.
2. Let \( X^* \) be the image obtained from \( X' \) by clearing all pixels in \( L \). Using the same hashing function \( H \) chosen for the insertion, compute the fingerprint \( H = H(X^*) \).
3. Extract the watermark from \( X' \) by scanning pixels in \( L \) and decrypt the result using the secret key (symmetric cipher) or public key (asymmetric cipher), obtaining the decrypted data \( D \).
4. Exclusive-or \( H \) with \( D \), obtaining the check image \( C \).
5. If \( C \) and \( A \) are equal, the watermark is verified. Otherwise, the marked image \( X' \) has been modified (or a wrong key has been used).

Figure 1 illustrates the AWST watermarking scheme. Let us suppose that image \( B \) [Fig. 1(a)] is a sensitive image to be transmitted through an unreliable channel, where unintentional or intentional alterations may occur. To protect \( B \), a logo image \( A \) [Fig. 1(b)] was inserted into \( B \) using the AWST algorithm. Image \( B' \) [Fig. 1(c)] is the watermarked image where 1024 bits were embedded. This is enough to embed a RSA digital signature. If the extraction algorithm is performed, we obtain the check image \( C \) [Fig. 1(d)], exactly equal to the logo image \( A \). If even a single pixel of \( B' \) is altered, the extracted image is completely noisy [Fig. 1(f)].

Figure 2 depicts the quality of an AWST-watermarked document. A page of a magazine was scanned at 300 dpi, resulting in a “typical” binary document with 3318 rows and 2536 columns [Fig. 2(a)]. Figures 2(b)–(d) show respectively the document with 128 bits, 320 bits, and 1024 bits embedded. These quantities of bits are enough to insert, respectively, a secret-key message authentication code, a public-key DSA digital signature, and a public-key RSA digital signature (Schneier, 1996). The visual qualities of the watermarked documents are excellent, because the quantities of inserted bits are insignificant compared to the number of pixels of the host image.

V. VARIATIONS ON THE PROPOSED METHOD

Fragile authentication watermarks can be subdivided into three subcategories: keyless, secret-key, and public-key watermarks. All three subcategories can be obtained using the AWST and they are described in subsections V.A, V.B, and V.C.

Another possible variation is a watermarking scheme that does not use the logo image. In this case, the detection algorithm will not extract a check image \( C \), but it will answer a Boolean question: the image has or has not been altered (subsection V.D).

Another possible variation is to use the improved data hiding techniques DHPT and DHSP (instead of the DHST) to insert the watermark in dispersed-dot halftone images, keeping constant the average grayscale around the pseudo-random pixels (subsection V.E).

A. Keyless AWST. Keyless authentication watermarking is useful for detecting unintentional alterations in images such as cropping and distortions due to dirt or human writing/marking. It is a sort of “check-sum.” If the watermarking insertion and detection algorithms are made public, anyone can insert and verify keyless authentication watermarks. In the keyless AWST, the seed of the pseudo-random number generator must be made public. The encryption (step 6 of the AWST insertion algorithm) must be eliminated as well as the decryption (step 3 of the AWST extraction algorithm). The hashing function can be very short, say, 24 bits, because it is very unlikely that an unintentional alteration will cause a hashing collision.

B. Secret-Key AWST. Secret-key authentication watermarking can be used to detect any alteration in image, even intentional or malicious ones. This kind of watermarking is similar to the well-known message authentication code. The only difference is that the authentication code is inserted into the host image instead of being independently stored. Algorithms for watermark insertion and detection can be public and a secret key is used in both phases. The seed of the pseudo-random number generator may be kept secret or made public, because the security does not lie on the secrecy of the seed.

Let us suppose that Alice administers a large image database where each image is signed with a secret-key \( k \) that only Alice knows. Suppose that Mallory, a malicious active attacker, modifies one image in the database. Mallory cannot insert the correct watermark into the modified image because he does not know \( k \). Moreover, Alice will be able to detect all images that were altered by Mallory using the AWST extraction algorithm and her secret-key \( k \).

Typically a 128-bit-long message authentication code (MAC) is considered cryptographically secure. Introductory books on cryptography, such as Schneier’s (1996), explain many different MAC schemes.
Figure 1. Illustration of the public-key AWST. Logo image $A$ (32×32 pixels) was inserted into image $B$. Panel (c) depicts the watermarked image. Executing the watermark extraction algorithm, the correct check image $C$ (d) was obtained. When the watermarked image was modified (e), a completely random check image was extracted (f).

(1a) Part of a 512×512 dispersed-dot halftone image $B$ to-be-watermarked.

(1b) Logo image $A$ (32×32 pixels) to-be-inserted into $B$.

(1c) Part of watermarked image $B'$. 1024 bits were embedded.

(1d) Check image $C$ extracted from image $B'$.

(1e) Part of the modified image $X'$.

(1f) Check image $C$ extracted from image $X'$. 
C. Public-Key AWST. Public-key authentication watermarking techniques use a public-key cryptography to insert the digital signature into the host image. Using a public-key cipher, claims of image authenticity can be judged without the necessity of disclosing any private information.

Let us suppose that Alice wants to send to Bob a sensitive image without disclosing her secret key. Alice uses her private key to insert watermark into the image and sends it to Bob. Bob uses Alice’s public key to verify that Alice signed the image and nobody introduced any alteration after Alice signing it. If Mallory, a malicious hacker, alters the image, he cannot insert the correct watermark into the falsified image because he does not know Alice’s private key.

Public-key authentication watermarking for binary images can be used, for example, in secure fax transmission. Let us suppose that each fax machine has its own internal secret key. Each time a transmission is performed, the sending fax machine inserts the AWST watermark. The fax receiver knows the public key of the sending machine. Thus, the receiver can verify that a specific fax machine originated the document and that the document has not been manipulated. The verification takes place using the digital image received by the fax machine. The printed version of the image cannot be authenticated using AWST.

Another possible application of the public-key AWST is in legal usage of binary documents. If documents are transmitted through the Internet, it is important that the receiver ensures that a specific person has signed the document and that no alteration has occurred after the signature.

A database of sensitive documents can be protected against fraudulent manipulations using the public-key AWST and anyone can verify the authenticity of a document using the public key.

The most well-known digital signature, RSA, is considered secure with 320 bits. A newer scheme, DSA, is considered secure with 1024 bits (Schneier, 1996).

D. Boolean Answer. Although extracting a visible logo from a watermarked image may be appealing, we only need to receive a binary answer from the AWST extraction algorithm: whether the image contains a valid watermark or not. To obtain a Boolean answer, we can eliminate step 5 from the AWST insertion algorithm and step 4 from the AWST extraction algorithm.

E. Keeping Constant Local Average Grayscales. The visual quality of an AWST-marked dispersed-dot halftone image can be improved by using improved data-hiding techniques DHPT and DHSPT (Fu and Au, 2000, 2002a), instead of DHST. These improvements try to keep constant the local average intensity. At the selected pseudo-random locations, the pixel alteration is accompanied by a complementary modification of a neighbor pixel.

However, to implement this scheme, no neighbor of the selected pseudo-random pixels must be fed into the hashing function. Consequently, these locations will remain unprotected, that is, if an alteration occurs in one of the neighbors of a selected pseudo-random pixel, this alteration will not be detected by the AWST scheme.

VI. CONCLUSIONS

We have proposed a cryptographically secure authentication-watermarking technique for halftone and binary images, named AWST. It is especially suitable for authenticating dispersed-dot halftone images. However, as the quantity of pixels to be altered is very small (compared to the size of a “typical” binary document), it can be applied to any binary image without causing a considerable loss of quality. The proposed technique can be used in three ways: keyless, secret key, and publickey. Public-key AWST is the most useful; it can be used in trustworthy fax machines, to electronically sign binary documents, to protect a database of sensitive documents, etc.

ACKNOWLEDGMENTS

The authors would like to express their gratitude to FAPESP and CNPq for partial financial support of this work under grants 2001/02400-9 and 300689/98-5, respectively.

REFERENCES