Authentication Watermarkings for Binary Images

Hae Yong Kim
Escola Politécnica, Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158
CEP 05508-900, São Paulo, SP, Brazil
(55-11) 3091-5605, hae@lps.usp.br

Sergio Vicente Denser Pamboukian
Universidade Presbiteriana Mackenzie
R. Consolação, 930, CEP 01302-907, São Paulo, SP, Brazil
(55-11) 2114-8553, sergiop@mackenzie.com.br

Paulo Sérgio Licciardi Messeder Barreto
Escola Politécnica, Universidade de São Paulo
Av. Prof. Luciano Gualberto, trav. 3, 158
CEP 05508-900, São Paulo, SP, Brazil
(55-11) 3091-9749, pbarreto@larc.usp.br

Keywords: Authentication, digital watermark, public key encryption, JBIG2, binary image.

Abstract

Data hiding (DH) is a technique used to embed a sequence of bits in a cover image with small visual deterioration and the means to extract it afterwards. Authentication watermarking (AW) techniques use DHs to insert a particular data into an image, in order to detect later any accidental or malicious alterations in the image, as well as to certify that the image came from the right source. In recent years, some AWs for binary images have been proposed in the literature. The authentication of binary images is necessary in practice, because most scanned and computer-generated document images are binary. This publication describes techniques and theories involved in binary image AW: We describe DH techniques for binary images and analyze which of them are adequate to be used in AWs; analyze the most adequate secret and public-key cryptographic ciphers for the AWs; describe how to spatially localize the alteration in the image (besides detecting it) without compromising the security; present AWs for JBIG2-compressed binary images; present a reversible AW for binary images; and finally present our conclusions and future research.

1. Introduction

This publication describes techniques and theories involved in binary image authentication watermarking. The authentication of binary images is necessary in practice because most of scanned and computer-generated document images are binary. These documents must be protected against fraudulent alterations and impersonations.

Binary images can be classified as either halftone or non-halftone. Halftone images are binary representations of grayscale images. Halftoning techniques (Ulichney, 1987; Knuth, 1987; Roetling et al., 1994) simulate shades of gray by scattering proper amounts of black and white pixels. On the other hand, non-halftone binary images may be composed of characters, drawings, schematics, diagrams, cartoons, equations, etc. In many cases, a watermarking algorithm developed for halftone images cannot be applied to non-halftone images and vice-versa.

Data hiding (DH) or steganography is a technique used to embed a sequence of bits in a cover image with small visual deterioration and the means to extract it afterwards. Most DH techniques in the literature are designed for grayscale and color images and they cannot be directly applied to binary images. Many of continuous-tone DHs modify the least significant bits (Wong, 1998), modify the quantization index (Chen et al., 2001), or modify spectral components of data in a spread-spectrum-like fashion (Cox et al., 1997; Marvel et al., 1999). Many of the continuous-tone DHs makes use of transforms like DCT and wavelet. Unfortunately, none of the above concepts (least significant bits, quantization indices and spectral components) are applicable to binary images. Binary images can be viewed as special cases of grayscale images and consequently can be transformed using DCT or wavelet, resulting in continuous-tone images in transform-domain. However, modifying a transform-domain image to insert the hidden data and inverse transforming it, usually will not yield a binary image. Hence, transforms like DCT and wavelet cannot be used to hide data in binary images. As consequence of the reasoning above, special DH techniques must be designed specifically for binary images.

A watermark is a signal added to the original cover image that can be extracted later to make an assertion about the image. Digital watermarking techniques can be roughly classified as either “robust watermarks,” or “authentication watermarks.” 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.

Authentication watermarks (AWs) use DH techniques to insert the authentication data into an image, in order to detect later any accidental or malicious alterations in the image, as well as to certify that the image came from the right source. AWs can be further classified in two categories: fragile and semi-fragile watermarks.

Fragile watermarks are designed to detect any alteration in the image, even the slightest. They are easily corrupted by any image-processing procedure. However, watermarks for checking image integrity and authenticity can be fragile because if the watermark is removed, the watermark detection algorithm will correctly report the corruption of the image. We stress that fragile authentication watermarks are deliberately not robust in any sense. In the literature, there are many AW techniques for continuous-tone images (Zhao et al., 1995; Yeung et al., 1997; Wong, 1998; Barreto et al., 1999; Holliman et al., 2000; Barreto et al., 2002). It seems to be very difficult to design a really secure AW without making use of the solid cryptography theory and techniques. Indeed, those AWs that were not founded in cryptography theory (Zhao et al., 1995; Yeung et al., 1997) or those that applied cryptographic techniques without the due care (Wong, 1998; Li et al., 2000) were later shown to be unreliable (Barreto et al., 1999; Holliman et al., 2000; Barreto et al., 2002). In a cryptography-based authentication watermarking, the message authentication code (MAC) or the digital signature (DS) of the whole image is computed and inserted into the image itself. However, inserting the MAC/DS alters the image and consequently alters its MAC/DS, invalidating the watermark. This problem can be solved by dividing the cover image Z in two regions Z_1 and Z_2, computing the MAC/DS of Z_2, and inserting it into Z_1. For example, for uncompressed or lossless-compressed gray-scale and color images, usually the least significant bits (LSBs) are cleared, the MAC/DS of the LSB-cleared image is computed and then the code is inserted into the LSBs (Wong, 1998). For JPEG-compressed images, the 8×8 blocks are divided in two groups Z_1 and Z_2, MAC/DS of Z_2 is computed and each bit of the code is inserted in an 8×8 block of Z_1 by, for example, forcing the sum of the DCT coefficient to be odd or even (Marvel et al., 2000). In this publication, we describe similar fragile AW techniques for binary images and the associated security issues.

Semi-fragile watermarks, like fragile ones, are designed to check the image integrity and authenticity. However, semi-fragile watermarks try to distinguish harmless alterations (such as lossy compression, brightness/contrast adjusting, etc.) from malicious image forgeries (intended to remove, substitute or insert objects in the scene). The demarcation line between benign and malicious attacks is tenuous and application-dependent. Consequently, usually semi-fragile AWs are not as secure as cryptography-based fragile AWs.

We are not aware of any semi-fragile AW for binary images. In the literature, there are many semi-fragile watermarks for continuous-tone images (Kundur et al., 1998; Fridrich, 1999; Marvel et al., 2000; Lin et al., 2000a; Lin et al., 2000b; Lin et al., 2001; Eggers et al., 2001; Lan et al., 2001, Yu et al., 2001). Ekici et al. (2004) enumerate eight "permissible" alterations that a semi-fragile watermarking must withstand:

1. JPEG compression;
2. Histogram equalization;
3. Sharpening;
4. Low-pass filtering;
5. Median filtering;
6. Additive Gaussian noise;
7. Salt-and-pepper noise;
8. Random bit error.

However, to our knowledge, there are no similar techniques for binary images. This is explicable considering that most of the above benign attacks (1, 2, 3, 4 and 6) cannot be applied to binary images. The remaining attacks (5, 7 and 8) can be applied to binary images but they are not so important in practice to deserve designing special semi-fragile watermarkings. Instead, there are practical interests in designing semi-fragile AWs for binary images that resist:

a) Lossy JBIG2 compression;
b) Geometric attacks, that is, rotation, scaling, translation and cropping;
c) Print-scan and photocopy.

Let us consider the possibilities of developing these semi-fragile AWs:

a) The JBIG2 standard has been developed by the Joint Bi-level Experts Group (JBIG) for the efficient lossless and lossy compression of bilevel (black and white) images. It is capable of compressing black and white documents considerably more than the more commonly used CCIT Group 4 TIFF compression. It was incorporated to the well-known PDF format. To our knowledge, there is no semi-fragile watermarking for binary images that resists to different levels of lossy JBIG2 compression. In section 6, we discuss a DH technique named DHTCJ that embed bits in JBIG2-compressed images (both lossy or lossless). This technique is not semi-fragile, and consequently the hidden bits will be lost if the watermarked image is re-compressed to different compression levels. The hidden bits can be extracted from the bitmap image obtained by uncompressing JBIG2 image.

b) There are many watermarking techniques for continuous-tone images that can resist geometric distortions. For example, Kutter (1998) replicates the same watermark several times at horizontally and vertically shifted locations. The multiple embedding of the watermark results in additional autocorrelation peaks. By analyzing the configuration of the extracted peaks, the affine distortion applied to the image can be determined and inverted. Pereira et al. (1999, 2000) and Lin et al. (2001) present watermarking resistant to geometric distortions based on the logpolar or log-log maps. The technique presented by Kutter (1998) can be applied to halftone binary images. For example, Chun et al. (2004) insert spatially replicated registration dots to detect the affine distortion in watermarked halftone images. It seems that the logpolar transform cannot be directly applied to halftone images, because discrete halftone dots cannot withstand continuous logpolar transform. There are only a few DH techniques for non-halftone binary images that resist geometric distortions. They can be based on inserting and detecting some synchronization marks (Wu et al., 2004) or using document boundaries. Kim et al. (2007) present a geometric distortion-resistant DH technique for printed non-halftone binary images based on tiny, hardly visible synchronization dots. However, a watermarking or data-hiding technique that resists geometric attacks is not automatically a semi-fragile AW resistant to geometric distortions. In our opinion, a robust hashing must be somehow integrated to geometric distortion-resistant watermarking to yield geometric distortion-resistant semi-fragile AW. Robust hashing $h(A)$, also called perceptual image hashing or media hashing, is a value that identifies the image A (Schneider et al., 1996). Moreover, given two images A and B, the distance D between the hashing must be somehow proportional to the perceptual visual difference of the images A and B. Lu et al. (2005) present a robust hashing for continuous-tone image that withstand geometric-distortion. In short, to our knowledge, still there is no geometric distortion-resistant semi-fragile AW for binary images.

c) There are some DH techniques for binary images robust to print-photocopy-scan. Data may be embedded imperceptibly in printed text by altering some measurable property of a font such as position of a character or font size (Maxemchuk et al., 1997; Bassil et al., 1999).

Bhattacharjya et al. (1999) and Borges et al. (2007) insert the hidden data by modulating the luminance of the some elements of the binary image (for example, individual characters). These elements are printed in halftone, and the average brightness, standard deviation or other features are used to extract the hidden bits. Kim et al. (2007) print tiny barely visible dots that carry information. The information hidden in these dots survive the photocopy operation. However, a DH that resists print-photocopy-scan is not automatically a semi-fragile AW that resists print-photocopy-scan. We are not aware of any semi-fragile AW for binary images that resists print-photocopy-scan distortion.

This publication discusses only fragile AWs for binary images in digital form, because as we considered above, semi-fragile AWs seemingly are still in development.

A possible application of AW for binary images is in Internet fax transmission, i.e. for legal authentication of documents routed outside the phone network. Let us suppose that Alice wants to send an authenticated binary document to Bob. She watermarks the binary image using her private-key and sends it to Bob through an unreliable channel. Bob receives the watermarked document and, using Alice's public-key, can verify that Alice signed the document and that it was not modified after watermarking it. Bob sends a copy of the document to Carol, and she also can verify the authenticity and integrity of the document by the same means.

Friedman (1993) introduced the concept of "trustworthy digital camera." In the proposed camera, the image is authenticated as it emerges from the camera. To accomplish this, the camera produces two output files for each captured image: the captured image and an encrypted "digital signature" produced by applying the camera's unique private key embedded within the camera's secure microprocessor. Using watermarking, the digital signature can be embedded into the image. This scheme can be applied to scanners that scan binary documents using the authentication watermarking techniques presented in this publication.

The rest of this publication is organized as follows. In section 2, we describe some DH techniques for binary images. In section 3, we analyze which DH techniques are adequate to be used in binary image AWs. In section 4, we analyze the state-of-art in cryptography, describing how to get short message authentication code and digital signature without compromising the security. In section 5, we describe how to spatially localize the alterations in the watermarked stego image. In section 6, we present an AW for JBIG2-compressed binary images. The creation of secure AWs for compressed binary images is an important practical problem, because uncompressed binary images use to be very large and can be compressed with high compression rates. In section 7, we present a reversible DH for binary images and how to use it as an AW. Reversible DH allows recovering the original cover image exactly (besides allowing to insert a sequence of bits in the image with small visual deterioration and to recover it later). Finally, in section 8, we present our conclusions and future research.

2. Data Hiding Techniques for Binary 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. Component-wise: Change the characteristics of some pixel groups, e.g., the thickness of strokes, the position or the area of characters and words, etc. (Maxemchuk et al., 1997; Bassil et al., 1999). Unfortunately, the success of this approach depends highly on the type of the cover image.

2. Pixel-wise: Change the values of individual pixels. Those pixels can be chosen randomly (Fu et al., 2000) or according to some visual impact measure (Kim, 2005; Mei et al., 2001).
3. Block-wise: Divide the cover image into blocks and modify some characteristic of each block to hide the data. Some papers suggest changing the parity (or the quantization) of the number of black pixels in each block (Wu et al., 2004). Others suggest flipping one specific pixel in the block with m pixels to insert $\lfloor \log_2(m+1) \rfloor$ bits (Tseng et al., 2002; Chang et al., 2005).

In this section, we present briefly some of the above-mentioned DH techniques that will be used to obtain AWs:

2.1 Data Hiding by Self-Toggling (DHST, pixel-wise)

DHST is probably the simplest DH technique for binary images (Fu et al., 2000; Kim et al., 2004a). In DHST, a pseudo-random number generator with a known seed generates a sequence v of pseudo-random non-repeating data-bearing locations within the image. Then one bit is embedded in each data-bearing location by forcing it to be either black or white. To extract the data, the same sequence v is generated and the values of the data-bearing pixels of v are extracted. This technique is adequate primarily for dispersed-dot halftone images. Otherwise, images watermarked by this technique will present salt-and-pepper noise.

2.2 Data Hiding by Template Ranking (DHTR, block-wise)

In DHTR (Wu et al., 2004; Kim et al., 2004b), the cover image is divided into blocks (say, 8×8). One bit is inserted in each block by forcing the block to have even or odd number of black pixels. If the block already has the desired parity, it is left untouched. Otherwise, toggle the pixel in the block with the lowest visual impact. Figure 1 depicts one of many possible tables with 3×3 patterns in increasing visual impact order of their central pixels. As different blocks may have different quantities of low visibility pixels, it is suggested to "shuffle" the image before embedding data. This shuffling must use a data structure that allows accessing both the shuffled image (to distribute evenly low visible pixels among the blocks) and the original unshuffled image (to allow computing the visual impact of a pixel by examining its unshuffled neighborhood). Images watermarked by DHTR usually present high visual quality, because it flips preferentially the pixels with low visual impact.

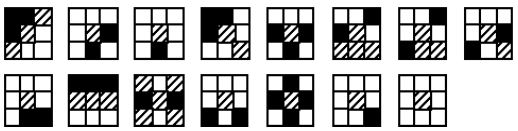
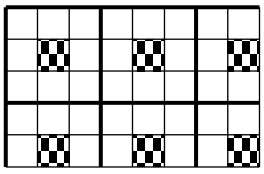
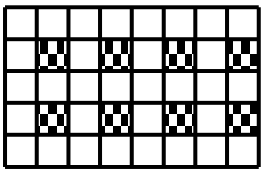


Fig. 1: A 3×3 template ranking in increasing visual impact order with "symmetrical central pixels." Hatched pixels match either black or white pixels (note that all patterns have hatched central pixels). The score of a given pattern is that of the matching template with the lowest impact. Mirrors, rotations and reverses of each pattern have the same score.



(a) Non-overlapping neighborhoods.



(b) Neighborhoods that do not contain another candidate to bear data.

Fig. 2: Distribution of candidate pixels to bear data, using 3×3 neighborhoods to evaluate visual impact scores.

2.3 Data Hiding by Template ranking with symmetrical Central pixels (DHTC, pixel-wise)

DHTC is another pixel-based DH technique (Kim, 2005). Here, the sequence v of data-bearing locations is chosen according to some visual impact score, instead of randomly selected as in DHST. The pixels with low visual impact are selected preferentially to bear the data. However, flipping data-bearing pixels may modify the visual scores of the neighboring pixels, and consequently make it impossible to reconstruct v in the data extraction. This problem is solved by: (1) using visual impact scores that do not depend on the value of its central pixel (figure 1); (2) choosing data-bearing pixels such that their neighborhoods (used to compute the visual scores) do not contain another data-bearing pixel (figure 2b). In the original paper, the author stated that the data-bearing pixels' neighborhoods should not overlap (figure 2a); however, we noticed that it is enough that the neighborhoods of data-bearing pixels do not contain another data-bearing pixel (figure 2b), increasing the data embedding capacity. DHTC insertion algorithm is:

1. Let be given a cover image Z and n bits of data to be inserted into Z. Construct the sequence v of candidate pixels to bear data, as explained above.
2. Sort v in increasing order using the visual scores as the primary-key and non-repeating pseudo-random numbers as the secondary-key. The secondary-key prevents from embedding the data mostly in the upper part of the image.
3. Embed n bits of data flipping (if necessary) the n first pixels of v. Those n pixels are called data-bearing pixels.

Mei et al. (2001) present another technique based on similar ideas. The images watermarked by DHTC have high visual quality, because it flips preferentially the pixels with low visual impact.

2.4 Chang, Tseng and Lin's Data Hiding (DHCTL, block-wise)

Tseng et al. (2002) present a block-wise DH technique that modifies at most two pixels in a block with m pixels to insert $\lfloor \log_2(m+1) \rfloor$ bits. Chang et al. (2005) improved this technique to insert the same number of bits by modifying one bit at most. We will explain Chang et al.'s ideas through an example, instead of giving general formulas. Let us suppose that the cover binary image is divided into blocks with 2×4 pixels. In this case, each block can hide 3 bits. The pixels of a block receive "serial numbers" ranging from 001 to 111, as in figure 3a (some numbers, as 001 in the example, may be repeated). Figure 3b represents the cover block to be watermarked. This block is currently hiding the number 011⊗101⊗111 = 001 (exclusive-or of the serial numbers of the pixels with value 1). Let us suppose that the number 101 is to be hidden in this block. To modify the hidden number from 001 to 101, we have to flip the pixel

with the serial number 001⊗101 = 100. Figure 3c depicts the resulting block. A stego image marked by this technique will present salt-and-pepper noise, because no visual impact was taken into account to choose the flipping pixels.

001	010	011	100
101	110	111	001

(a) Binary "serial numbers."

0	0	1	0
1	0	1	0

(b) Cover block to watermark.

0	0	1	1
1	0	1	0

(c) Block with hidden 101.

Fig. 3: Illustration of DHCTL.

3. Authentication Watermarking for Binary Images

Cryptography-based AWs can be subdivided in three groups:

1. Keyless: Keyless AW is useful for detecting unintentional alterations in images. It is a sort of "check-sum." Cryptographic one-way hashing functions can be used to obtain the integrity index to be inserted in the cover image to certify its integrity.
2. Secret-key: In a secret-key AW, there must exist a secret-key known only by the image generator (say Alice) and the image receiver (say Bob). Alice computes the message authentication code (MAC) of the image to be protected using the secret-key and inserts it into the image itself. Then, the marked stego image is transmitted to Bob through an unreliable channel. Bob uses the secret-key to verify that the image was not modified after being watermarked by Alice.
3. Public-key: In a public-key AW, claims of image integrity and authenticity can be settled without disclosing any private information. Alice, the image generator, computes the digital signature (DS) of the image using her private-key and inserts it into the image. Only Alice can compute the correct DS, because only she knows her private key. Then, the stego image is transmitted through an unreliable channel. Anyone that receives the stego image can verify its authenticity (i.e., whether the image really came from Alice) and integrity (i.e., whether the image was not modified after being marked by Alice) using the Alice's public-key.

An AW scheme (of any of the three groups above) can either answer only a Boolean response (whether the image contains a valid watermark or not) or insert/extract a logo image (a valid logo will be extracted only if the stego image is authentic). Introductory books on cryptography, such as (Schneier, 1996), explain in more details concepts like one-way hashing, MAC and DS.

A DH technique can be transformed into an AW computing MAC/DS of the whole image and inserting it into the image itself. However, inserting the MAC/DS alters the image and consequently alters its MAC/DS, invalidating the watermark. This problem can be solved by dividing the cover image Z in two regions Z_1 and Z_2, computing the MAC/DS of Z_2, and inserting it into Z_1. Let us examine how this idea can be applied to the four DH techniques described in the previous section.

3.1 Authentication Watermarking by Self-Toggling (AWST)

AWST is obtained applying the idea above to DHST. In this case, region Z_1 where the MAC/DS will be inserted corresponds to the pixels that belong to the sequence v of data-bearing locations. We describe below the secret-key version of this algorithm that inserts and extracts a logo binary image. The other versions can be derived straightforwardly. Figure 4 illustrates this process.

1. Let Z be a cover binary image to be watermarked and let L be a binary logo. The number of pixels of L must be equal to the length of the chosen one-way hashing function H.
2. Use a pseudo-random number generator with a known seed to generate a sequence v of non-repeating pseudo-random data-bearing locations within the image Z.
3. Let Z_2 be the pixels of Z that do not belong to v, that is, $Z_2 \leftarrow Z \setminus v$. Compute the integrity index $H = H(Z_2)$, exclusive-or H with L, and encrypt the result with the secret-key, generating the MAC S.
4. Insert S flipping (if necessary) the pixels of the sequence v, generating the protected stego image Z'.

The AWST extraction algorithm is:

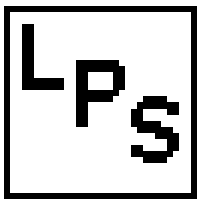
1. Let Z' be an AWST-marked image. Generate again the sequence of data-bearing pixels v.
2. Let $Z'_2 \leftarrow Z' \setminus v$. Compute the integrity index $H = H(Z'_2)$.
3. Extract the hidden data from Z' scanning the pixels in v and decrypt it using the secret-key, obtaining the decrypted data D.
4. Exclusive-or H with D, obtaining the check image C.
5. If C is equal to the inserted logo image L, the watermark is verified. Otherwise, the stego image Z' has been modified (or a wrong key has been used).

We suggest using AWCTL (subsection 3.4) instead of AWST, because the former has more data hiding capacity than the latter and an equivalent visual quality.

3.2 Authentication Watermarking by Template Ranking (AWTR) and Parity Attacks

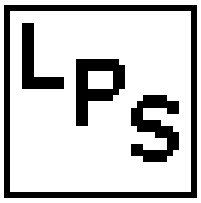
AWTR can be directly derived from the corresponding DH technique by dividing the cover image Z in two regions Z_1 and Z_2; computing the MAC/DS of Z_2; and inserting it into Z_1. However, some caution must be taken in transforming a DH scheme into an AW, because although the region Z_2 is well protected (with the security assured by the cryptography theory), the region Z_1 is not. For example, let us take the DH scheme that inserts one bit per connected component, forcing it to have even or odd number of black pixels. A connected component can be forced to have the desired parity by toggling one of its boundary pixels. This scheme can be transformed into an AW using the idea described above. Yet, a malicious hacker can arbitrarily alter the region Z_1, without being noticed by the AW scheme, as long as the parities of all connected components remain unaltered. For example, a character "a" in Z_1 region can be changed into an "e" (or any other character that contains only one connected component) as long as its parity remains unchanged. We refer to this as a "parity attack." In AWTR, the blocks of Z_1 can be modified, without being detected by the watermark, as long as their parities remain unchanged. To avoid parity attacks, we suggest using AWTC (subsection 3.3) instead of AWTR.

(a) Part of 512×512 cover halftone image *Z*.



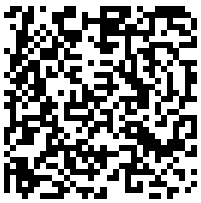
(b) Logo image *L* (32×32 pixels).

(c) Part of watermarked stego image *Z*'. 1024 bits were embedded.



(d) Check image *C* extracted from *Z*'.

(e) Part of the modified image *X*'.



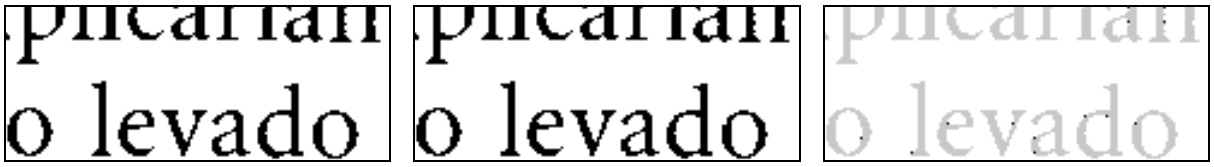
(f) Check image *C* extracted from *X*'.

Fig. 4: Logo image *L* (b) was inserted into cover image *Z* (a) using AWST. Figure (c) depicts the watermarked stego image. The correct check image *C* (d) was extracted from the stego image. When the stego image was modified (e), a completely random check image was extracted (f).

3.3 AW by Template ranking with symmetrical Central pixels (AWTC)

Surprisingly, the simple AWST cannot be assaulted by parity attacks. This happens because, in AWST, the number of pixels in Z_1 region is exactly equal to the length of the adopted MAC/DS. All image pixels (except the *n* pixels that will bear the *n* bits of the MAC/DS) are taken into account to compute the AS. Consequently, *any* alteration of Z_2 region can be detected because it changes the integrity index of the stego image, and *any* alteration of Z_1 region can also be detected because it changes the stored MAC/DS. The probability of not detecting an alteration is only 2^{-n} (where *n* is the length of MAC/DS), which can be neglected.

An image watermarked by AWTR presents high visual quality, but it can be assaulted by parity attacks. On the other hand, an image watermarked by AWST is noisy, but it cannot be assaulted by parity attacks. Is it possible to design an AW with AWTR's visual quality and AWST's security? Fortunately, AWTC (derived directly from DHTC) has high visual quality and is immune to parity attacks. Figure 5 illustrates this technique. A page of a magazine was scanned at 300 dpi (figure 5a) and watermarked by AWTC with 10240-bits long MAC (much longer than the usual), resulting in figure 5b. Note in figure 5c that only low-visibility pixels located at borders of characters were flipped.



(a) Cover document. (b) Stego document. (c) Flipped pixels.

Fig. 5: A page of a magazine scanned at 300 dpi and watermarked by AWTC using an unusually long MAC.

3.4 Authentication watermarking derived from Chang, Tseng and Lin's data hiding (AWCTL)

Sometimes, we may not be interested in flipping only low-visibility pixels, for example, to watermarking a dispersed-dot halftone image. In this case, it is possible to use AWST. However, a better technique can be obtained converting DHCTL into an AW. The advantage of AWCTL over AWST is that the MAC/DS can be embedded into the cover image flipping a smaller number of pixels. For example, using blocks with 255 pixels, 1024-bits long MAC/DS can be embedded flipping at most 128 pixels in AWCTL (instead of 1024 bits as in AWST).

AWCTL can be assaulted by parity attack only if $\log_2(m+1)$ is not an integer (m is the number of pixels of a block). Consider figure 3a, where two pixels have received the same serial number 001. If these two pixels are flipped together, the data hidden in the block will not change. If a hacker flips together any two pixels in a block with the same serial number, this alteration will not be detected by the watermarking scheme. However, if $\log_2(m+1)$ is an integer, there is no set of pixels with the same serial number, and thus this attack becomes impossible.

4. Public and Secret Key Cryptography for Authentication Watermarking

The very nature of any watermark requires minimizing the amount of data embedded in the cover image (to avoid deteriorating the quality of the resulting image) and maximizing the processing speed (due to the naturally high number of signatures one must generate and verify in realistic images, especially when spatial localization of alterations is involved). In cryptographic terms, the signatures inserted in a cover image must be as compact as possible, and its processing must be as efficient as feasible.

To address these requirements, watermarking schemes usually adopt either public-key digital signatures like Schnorr (1991), BLS (Boneh et al., 2002), and ZSNS (Zhang et al., 2004) (the latter two based on the concept of bilinear pairings), or secret-key message authentication

codes like CMAC (NIST, 2005) and Galois-Carter-Wegman (McGrew et al., 2005) (adopted in existing standards and/or worth of note due to their high performance). Schemes based on the former cryptographic primitives have unique properties like public verifiability (whereby detecting and/or verifying a watermark does not imply revealing private information), while schemes based on the latter are usually much faster (as much as two orders of magnitude).

We now briefly review the BLS and CMAC algorithms, which seem most suitable for watermark-based image authentication.

4.1 BLS Signatures

The signature algorithm we now describe makes use of notions from algebraic geometry; the interested reader can refer to (Cohen et al., 2005) for a thorough exposition of the concepts involved. BLS signatures adopt as public parameters: an elliptic curve E defined over a finite field GF(q) and the same curve over an extension field GF(q^k); two points $P \in E(\mathrm{GF}(q))$ and $Q \in E(\mathrm{GF}(q^k))$ of prime order r; a cryptographically secure hash function H: $I \to \langle P \rangle$ where I is the set of all valid images (or image blocks); and a bilinear pairing e: $\langle P \rangle \times \langle Q \rangle \to \mathrm{GF}(q^k)$ satisfying the relations $e(P, Q) \neq 1$ and $e(xP, Q) = e(P, xQ)$ for any integer x. For security and efficiency reasons, typically $\log_2 r \approx 160$ bits for $k = 6$; suitable parameters matching these constraints can be constructed using the MNT technique (Miyaji et al., 2001).

The entity (called the *signer*) wishing to create a BLS-based watermark establishes a key pair (s, V), where the secret key s in an integer randomly selected in range 1 to r-1, and the public key is the curve point $V = sQ$. To sign an image block $b \in I$, the signer computes $B = H(b)$ and $S = sB$; the signature is the point S. To verify the received signature of a received image block b, any interested party (called the *verifier*) computes $B = H(b)$ and checks that $e(S, Q) = e(B, V)$. This protocol works because, if the received block and signature are authentic, then $e(S, Q) = e(sB, Q) = e(B, sQ) = e(B, V)$. The signature size is only $\log_2 r$, hence typically 160 bits; by comparison, a typical RSA signature for the same security level is 1024-bits long.

4.2 CMAC message authentication code

The CMAC message authentication code is defined for an underlying block cipher E accepting a k-bit secret key to encrypt n-bit data blocks. Typically the cipher is either 3DES (k = 168 bits and n = 64 bits) or AES (k = 128, 192 or 256 bits and n = 128 bits). We write $E_K(b)$ for the encryption of a data block b under the key K with cipher E. CMAC authentication tags have an arbitrary size t between n/2 and n bits, with recommended values t = 64 for 3DES and t = 96 for AES. Both the signer and the verifier must know the key K. The image to be signed is assumed to be partitioned into blocks b_1, …, b_m according to some conventional ordering, with all blocks except possibly the last one being n bits in length; for convenience, we define a dummy block b_0 consisting of binary zeroes. Let $L = E_K(0^n)$ be the encryption of a string of n binary zeroes under the key K, viewed as a polynomial $L(x)$ in the finite field GF(2^n). Let $2L$ and $4L$ respectively stand for the polynomials $x \cdot L(x)$ and $x^2 \cdot L(x)$ in the same finite field. Notice that, on the assumption that K is not a weak key for the cipher E, neither of L, $2L$, or $4L$ consists exclusively of binary zeroes.

To generate a CMAC authentication tag, the signer computes $c_i = E_K(b_i \text{ XOR } c_{i-1})$ for i = 1, …, m–1, and finally $c_m = E_K(\mathrm{pad}(b_m) \text{ XOR } c_{m-1})$ where $\mathrm{pad}(b_m) = b_m \text{ XOR } 2L$ if the length $|b_m|$ of b_m is exactly n, or else $\mathrm{pad}(b_m) = (b_m \,||\, 1 \,||\, 0^*) \text{ XOR } 4L$ if $|b_m|$ is strictly smaller than n, where XOR stands for bitwise exclusive-or, || stands for bit string concatenation, and 0* denotes a (possible empty) string of binary zeroes long enough to complete an n-bit string (thus

in this context the length of 0* is $n - |b_m| - 1$). The CMAC tag is the value of c_m truncated to the t leftmost bits. To verify a CMAC tag, the verifier repeats this computation for the received image and compares the result with the received tag.

5. Spatially Localizing the Alterations

Watermarks that are capable of not only detecting, but also spatially localizing alterations in stego images with a previously established resolution are called *topological*. Wong (1998) has proposed a topological scheme that consists of dividing the cover continuous-tone image in blocks, computing MAC/DS of each block, and inserting the result MAC/DS into the least significant bits of the block. Similar ideas can be applied to binary images as well.

However, Wong's and other topological schemes succumb to a number of attacks, ranging from simple copy-and-paste attacks (whereby individually signed image blocks taken from legitimate images are undetectably copied onto equally sized blocks of other images or different positions in the same images) to the subtler *transplantation attacks* and *advanced birthday attacks*. Only a few proposed watermarking techniques can effectively counter these shortcomings, by introducing either non-deterministic or unbounded context to the generated signatures.

We first examine the principles of the aforementioned attacks, and then we show how to construct topological watermarking schemes that are able to resist them.

5.1 Transplantation attacks

Assume that an authentication scheme adds *locality context* to the signed data, in the sense that the signature of an image region Y depends also on the contents and coordinates of another region X; denote this relation between the regions by $X \to Y$. Also, write $X \sim Y$ if regions X and Y share the same contents (pixel values). Suppose that an opponent obtains two sets S and T of signed regions satisfying the relations:

$$A_S \to U_S \to B_S \to C_S,$$

$$A_T \to V_T \to B_T \to C_T,$$

where $A_S \sim A_T$, $B_S \sim B_T$, and $C_S \sim C_T$, but $U_S \neq V_T$. If each region carries along its own authentication tag, the pair of regions (U_S, B_S) is undetectably interchangeable with the pair (V_T, B_T), regardless of the signature algorithm being used.

5.2 Advanced birthday attacks

Let f be a surjective function whose range consists of 2^n distinct values, and suppose that a sequence $(f_1, f_2, \ldots, f_k)$ of values output by f is uniformly distributed at random in the range of f. The probability that two equal values occur in this sequence gets larger than ½ as soon as the sequence length becomes $O(2^{n/2})$. This purely stochastic phenomenon does not depend on the details of f, and is known as the *birthday paradox* (Stinson, 2002).

Assume that the signatures are n bits long, and suppose that a valid message contains three regions satisfying the relations:

$$L \to M \to R,$$

A naïve approach to forge a block M' in place of M would be to collect $O(2^{n/2})$ valid signatures from unrelated blocks (possibly from other images authenticated by the same signer) and to create $O(2^{n/2})$ semantically equivalent variants of M'; by the birthday paradox, with high probability at least one collected signature $s_{M'}$ that fits one of the variants of M'. But since the signature of R depends on the contents of M, its verification will almost certainly fail, not only thwarting this attack but still being capable of locating the forgery with a positional error of just one block. However, a clever attacker can still subvert any such scheme with a somewhat larger effort. Given $O(2^{3n/5})$ collected signatures, the strategy is to create $O(2^{3n/5})$ forged variants of M' and also $O(2^{3n/5})$ forged variants of L', and then to search the database for three signatures $s_{L'}$, $s_{M'}$, s_R that fit the resulting relations:

$$L' \rightarrow M' \rightarrow R.$$

By the birthday paradox, with high probability at least one such triple exists. Alternatively, the attack could be mounted from a database of $O(2^{n/2})$ collected signatures as in the naïve approach, as long as the forger creates $O(2^{3n/4})$ variants of each of L' and M'.

This attack works whenever the context used in the signatures are limited and deterministic, regardless of the details of the signature algorithm.

5.3. Hash block chaining (HBC) and other schemes

Perhaps the simplest way to thwart the aforementioned attacks against topological watermarks is by adopting *non-deterministic* signatures and extending dependencies of the form $X \rightarrow Y$ to $(X, \text{signature}(X)) \rightarrow Y$. A signature scheme is said to be non-deterministic if the computation of hash values and generation of signatures depend on one-time private random nonces (Schnorr signatures, for instance, are essentially non-deterministic). Since the modified dependency relation above effectively creates a chain of hash values (i.e. the non-deterministic hash of X is included in the computation of the hash of Y and so on), the resulting scheme is called *hash block chaining* (Barreto et al., 2002), by analogy to the cipher block chaining mode used for data encryption. The image dimensions (M lines and N columns) are also relevant for locating alterations and hence are usually present in the dependency relation, which becomes $(M, N, X, \text{signature}(X)) \rightarrow Y$.

For binary document images, there may be many entirely white blocks. If these blocks were watermarked, they would be contaminated by salt and pepper noise. To avoid this, instead of watermark them directly one can adopt a dependency of form $(M, N, X, \text{signature}(X), k) \rightarrow Y$, where k is the number of white blocks between blocks X and Y.

Few other topological schemes are able to thwart the advanced attacks described above. It is possible to avoid non-deterministic context by adopting a t-ary tree organization of signatures, whereby the individually signed image blocks are grouped into sets of t blocks, each set being individually signed as well; in turn, these sets are grouped into sets of t sets and so on, up to one single set covering the whole image, each intermediate set receiving its own signature (Celik et al., 2002b). The advantage of using this method is that, when the image is legitimate, only the upper level signature needs to be verified. The disadvantages are the lack of resolution in alteration localization within a set of blocks (where even the copy-and-paste attack can be mounted) and the larger amount of data that needs to be inserted in the cover image (twice as much for a typical arity of $t = 2$, which minimizes the resolution loss).

6. Data Hiding in JBIG2-Compressed Images (DHTCJ)

The creation and implementation of secure AWs for compressed binary images is an important practical problem, because scanned documents are usually large binary images, which may be stored in compressed formats in order to save the storage space. JBIG2 is an international standard for compressing bi-level images (both lossy and lossless) (ISO, 1999; Howard, 1998). In this standard, the image is decomposed in several regions (text, halftone and generic) and each region is compressed using the most appropriate method. In this section, we will describe a DH technique named DHTCJ, derived from DHTC (subsection 2.3), that hides data in the text region of JBIG2-compressed binary images (Pamboukian et al., 2005). DHST and DHCTL are not adequate to watermark JBIG2 text region because they introduce salt-and-pepper noise, and DHTR is not adequate because of the parity attacks.

A JBIG2 text region is coded using two kinds of segments: (1) *Symbol Dictionary Segment* that contains bitmaps of the characters present in the text region and (2) *Text Region Segment* that describes locations of characters within the text region, with references to the symbol dictionary. Many instances of a character can refer to the same symbol in the dictionary, increasing the compression rate. The compression is lossy if similar instances can refer to a unique symbol in the dictionary, and lossless if only identical instances can refer to a single symbol. The following algorithm embeds information in a JBIG2 text region:

1. Let be given a JBIG2-encoded image Y and n bits of data to be inserted into Y. Decode the text region of Y, obtaining the uncompressed binary image Z.
2. Construct the sequence v of candidate pixels to bear the data and sort it as described in DHTC (subsection 2.3).
3. Identify in the *text region segment* the symbols that contain the n first pixels of the sorted sequence v and their references to the symbols to bear the data in the *symbol dictionary segment*. Note that the number of data bearing symbols (DBSs) can be smaller than n, because each symbol can bear more than one bit.
4. Verify how many times each DBS is referenced in the *text region segment*. If there's only one reference, the data will be stored in the original symbol. If there's more than one reference, the symbol must be duplicated and inserted at the end of *symbol dictionary segment*. The data will be inserted in the duplicated symbol, instead of the original. The reference to the symbol in the *text region segment* should also be modified.
5. Insert n bits of data in the DBSs by flipping, if necessary, the n first pixels of the sorted sequence v.

Special care must be taken to the avoid connection or disconnection of the DBSs. To simplify the implementation, we suggest using the template ranking depicted in figure 6, where only the templates that cannot cause symbol connection or disconnection are listed. DHTCJ data extraction algorithm is straightforward, as well as the derivation of an AW technique. Images watermarked with DHTCJ have pleasant visual quality, as can be seen in figure 7.

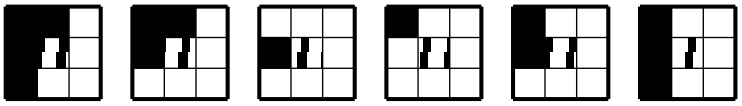


Fig. 6: Set of 3×3 templates designed to be used with DHTCJ, in increasing visual impact order. Only the templates that do not cause symbol connection or disconnection are listed. Hatched pixels match either black or white pixels. The score of a given pattern is that of the matching template with the lowest impact. Mirrors, rotations and reverses of each pattern have the same score.

(a) Part of the cover image.

(b) Stego image.

(c) Flipped pixels.

Fig. 7: Part of an image scanned at 300 dpi, with 626×240 pixels, 93 symbols instances and watermarked using AWTCJ with 128-bits long MAC.

7. Reversible Data-Hiding for Binary Images

Most DH techniques modify and distort the cover image in order to insert the additional information. This distortion is usually small but irreversible. Reversible DH techniques insert the information by modifying the cover image, but enable the exact (lossless) restoration of the original image after extracting the embedded information from the stego image. Some authors (Awrangjeb et al., 2004; Celik et al., 2005; Shi, 2004) classify reversible DHs in two types:

(1) Techniques that make use of additive spread spectrum techniques (Honsinger, 2001; Fridrich et al., 2001, section 3).
(2) Techniques where some portions of the host signal are compressed to provide space to store the net payload data (Fridrich et al., 2001, sections 4 and 5; Fridrich et al., 2002; Celik et al., 2002a; Tian, 2002; Tian, 2003; Awrangjeb et al., 2004; Ni et al., 2004; Celik et al., 2005).

Usually the techniques of the first type can hide only a few bits, while the techniques of the second type have more data hiding capacity. This section presents a reversible DH of the second type for binary images, named RDTC (Reversible Data hiding by Template ranking with symmetrical Central pixels; Pamboukian et al., 2006). Although there are many reversible DH techniques for continuous-tone images, to our knowledge RDTC is the only published reversible DH specifically designed for binary images. It is based on DHTC and uses the Golomb code to compress prediction errors of low-visibility pixels to obtain the space to store the hidden data.

In RDTC (as in most reversible DH of the second type) two kinds of information must be embedded in the host image: the compressed data to allow recovering the original image and the net payload data to be hidden. That is, the n data bearing pixels' (DBPs') original values are compressed in order to create space to store the payload data. There are some difficulties to compress the DBPs' original values. Most compression algorithms based on redundancy and dictionaries do not work, because the next bit cannot be predicted based on the previous bits since they correspond to the pixels dispersed throughout the whole image. The solution we found is to compress the prediction errors of DBPs' values, using its neighborhood as the side-information.

We tested two prediction schemes:

1. A pixel can be either of the same color or of the different color than the majority of its spatial neighboring pixels. Let us assume that the first hypothesis is more probable than the second. Let b be the number of black neighbor pixels of a DBP (using 3×3 templates, a DBP has 8 neighbor pixels). The prediction is correct (represented by 0) if the original DBP is black and $b>4$, or if it is white and $b\leq 4$. Otherwise, the prediction is wrong (represented by 1). If this prediction is reasonable, the predicted value and the true value should be the same with probability higher than 50%. As we store zero when the prediction is correct and one when it is wrong, subsequences of zeroes will be longer (in most cases) than subsequences of ones.

2. We also tested a more elaborate prediction scheme. We constructed a table with 256 elements (all possible configurations of 8 neighbor pixels) and, using typical binary images, determined the most probable central pixels' colors, based on the 8 neighbors' configurations.

Surprisingly, the two prediction schemes yielded almost the same results. The sequence of prediction errors consists of usually long segments of zeroes separated by usually short segments of ones, because a zero occurs with high probability p and a one occurs with low probability 1-p. An efficient method to compress this type of information is the Golomb code (Golomb, 1966; Salomon, 2004). The original sequence is first converted in run-lengths of zeroes (sequence of non-negative integers). Then, each integer is compressed by the Golomb code. The Golomb code depends on the choice of an integer parameter $m\geq 2$ and it becomes the best prefix code when

$$m = \left\lceil -\frac{\log_2(1+p)}{\log_2 p} \right\rceil .$$

The reader is referred to (Golomb, 1966; Salomon, 2004) for more details on the Golomb code.

The stored compressed vector of prediction errors, together with the neighborhoods of DBPs, allows recovering the original DBPs' values. RDTC insertion algorithm is:

1. Let be given the binary cover image Z. Construct the sequence v of candidate pixels for bearing data as described in DHTC (subsection 2.3).
2. Sort v in increasing order using the visual scores as the primary-key, the number of black pixels around the central pixels as the secondary-key and non-repeating pseudo-random numbers as the tertiary-key.
3. Estimate the smallest length n of DBPs capable of storing the header (size h), the compressed vector of prediction errors (size w) and the given net payload data (size p). I.e., find n that satisfies $n \geq h+w+p$. Try iteratively different values of n, until obtaining the smallest n that satisfies the inequality.
4. Insert the header (the values of n, w, p and the Golomb code parameter m), the compressed vector of prediction errors and the payload by flipping the central pixels of the first n pieces of the sorted v.

To extract the payload and recover the original image, the sequence v is reconstructed and sorted. Then, the data is extracted from the n first central pixels of v. The compressed vector of prediction errors is uncompressed and used to restore the original image. To obtain AW based on a reversible DH, it is not necessary to compute the MAC/DS of a region of the cover image and store it in another region. Instead, the MAC/DS of the whole image is computed and inserted. The authentication can be verified because it is possible to recover the original image Z and the stored MAC/DS.

We have embedded the data at the beginning of v, because this part has the least visible pixels. However, they cannot be predicted accurately, because usually they have similar number of black and white pixels in their neighborhoods (since they are boundary pixels). As we move forward in the vector, we find pixels that can be predicted more accurately, but with more visual impact. To obtain a higher embedding capacity (sacrificing the visual quality), we can scan the vector v searching for a segment that allows a better compression. In this case, the initial index of the embedded data in v must also be stored in the header. Table 1 shows the number n of needed DBPs to hide 165 bits of payload at the beginning of v (the best quality) and in a segment that allows the best compression. In the latter case, the values of 176 pixels could be compressed to only 11 bits. Table 1 also shows the maximum amount σ of bits that can be reversibly inserted in each image.

Table 1: Insertion of 128 bits of payload and 37 bits of header in different images, where n is the number of DBPs and w is the size of the compressed DBPs. σ is the maximum amount of bits that can be reversibly inserted.

Image description	Size	Best quality			Best compression			σ
		n	w	n-w	n	w	n-w	
Computer-generated text	1275×1650	336	146	190	176	11	165	401,600
150 dpi scanned text	1275×1650	432	265	167	176	11	165	214,032
300 dpi scanned text	2384×3194	496	325	171	176	11	165	1,543,680

A reversible fragile authentication watermarking can be easily derived from RDTC, using secret-key or public/private key ciphers. Similar reversible authentication techniques for continuous-tone images can be found in many papers on reversible DHs, for example, (Fridrich et al., 2001; Dittmann et al., 2004). The public/private-key version of authentication watermarking insertion algorithm is:

1. Given a binary image Z to be authenticated, compute the integrity index of Z using a one-way hashing function $H = H(Z)$. Cryptograph the integrity index H using the private-key, obtaining the digital signature S.
2. Insert S into Z using RDTC, obtaining the watermarked stego image Z'.

The authenticity verification algorithm is:

1. Given a stego image Z', extract the authentication signature S and decrypted it using the public-key, obtaining the extracted integrity index E.
2. Extract the vector of prediction errors, uncompress it and restore the original cover image Z. Recalculate the hashing function, obtaining the check integrity index $C = H(Z)$.
3. If the extracted integrity index E and the check integrity-index C are the same, the image is authentic. Otherwise, the image was modified.

8. Conclusions

As we have seen, it is possible to construct secure topological AW schemes for binary images in digital form, particularly for JBIG2-encoded (lossy of lossless) binary images. Such schemes, based on digital signatures or message authentication codes, can pinpoint alterations in stego images, and are closely related to DH techniques. In this regard, we have described a reversible DH scheme for authenticating binary images in such a way that the cover image can be entirely recovered.

9. Future Research Directions

Semi-fragile authentication watermarking for binary images, especially printed binary documents, is an open issue, object of future research. Hardcopy document authentication depends on three components: a data hiding technique that resists print-photocopy-scanning; a perceptual hash that assigns a unique index to visually identical documents (even if these documents contain noise, are rotated, scaled, etc.) and cryptographic algorithms. Among these components, only cryptography is a mature technology. The other two are still undergoing development.

References

Awrangjeb, M., & Kankanhalli, M. S. (2004). Lossless Watermarking Considering the Human Visual System. *Int. Workshop on Digital Watermarking 2003,* Lecture Notes in Computer Science 2939, pp. 581–592.

Barreto, P. S. L. M. & Kim, H. Y. (1999). Pitfalls in Public Key Watermarking, *Proc. Brazilian Symp on Computer Graphics and Image Processing*, pp. 241-242.

Barreto, P. S. L. M., Kim, H. Y., & Rijmen, V. (2002) Toward a Secure Public-Key Blockwise Fragile Authentication Watermarking. *IEE Proc. Vision, Image and Signal Processing*, vol. 149, no. 2, pp. 57–62.

Bhattacharjya, A.K., & Ancin, H. (1999). Data Embedding in Text for a Copier System. *Int. Conf. Image Processing*. vol. 2, pp. 245-249.

Boneh, D., Lynn, B., & Shacham, H. (2002). Short Signatures from the Weil Pairing. *Advances in Cryptology – Asiacrypt'2001*, Lecture Notes in Computer Science 2248, pp. 514–532.

J.T. Brassil, S. Low, N.F. Maxemchuk, "Copyright Protection for the Electronic Distribution of Text Docu-ments," Proc. of IEEE, vol. 87, no. 7, pp. 1181-1196, July 1999.

Borges, P. V. K., & Mayer, J. (2007). Text Luminance Modulation for Hardcopy Watermarking. Signal Processing, vol. 87, no. 7, pp. 1754-1771.

Celik, M. U., Sharma, G., Tekalp, A. M., & Saber, E. (2002a). Reversible Data Hiding. *IEEE Int. Conf. on Image Processing*, vol. 2, pp. 157–160.

Celik, M. U., Sharma, G., Saber, E., & Tekalp, A. M. (2002b). Hierarchical Watermarking for Secure Image Authentication with Localization. *IEEE Trans. Image Processing*, vol. 11, no. 6, pp. 585-595.

Celik, M. U., Sharma, G., Tekalp, A. M., & Saber, E. (2005). Lossless Generalized-LSB Data Embedding. *IEEE Trans. Image Processing*, vol. 14, no. 2, pp. 253-266.

Chang, C.-C., Tseng, C.-S., & Lin, C.-C. (2005). Hiding Data in Binary Images. *Lecture Notes in Computer Science* 3439, pp. 338–349.

Chen B., & Wornell G. W. (2001). Quantization Index Modulation: A Class of Provably Good Methods for Digital Watermarking and Information Embedding. *IEEE Trans. Information Theory*, vol. 47, no. 4, pp. 1423-1443.

Chun, I. G., & Ha, S. H. (2004). A Robust Printed Image Watermarking Based on Iterative Halftoning Method, *Int. Workshop on Digital Watermarking 2003*, Lecture Notes in Computer Science 2939, pp 200-211.

Cohen, H., Frey, G., Avanzi, R. M., Doche, C., Lange, T., Nguyen, K., & Vercauteren, F. (2005). *Handbook of Elliptic and Hyperelliptic Curve Cryptography*. Chapman & Hall/CRC.

Cox, I., Kilian, J., Leighton, F. T., & Shamoon T. (1997), Secure Spread Spectrum Watermarking for Multimedia. *IEEE T. Image Processing*. vol. 6, no. 12, pp. 1673-1687.

Dittmann J., Katzenbeisser S., Schallhart C., & Veith, H. (2004). Provably Secure Authentication of Digital Media Through Invertible Watermarks. *Cryptology ePrint Archive: Report 2004/293*, available at http://eprint.iacr.org/2004/293.

Eggers, J. J., & Girod, B. (2001). Blind Watermarking Applied to Image Authentication. *ICASSP'2001: Int. Conf. Acoustics, Speech and Signal Processing*, Salt Lake City, USA.

Ekici, O., Sankur, B. & Akcay, M. (2004). A Comparative Evaluation of Semi-Fragile Watermarking Algorithms. *Journal of Electronic Imaging*. vol. 13, no. 1, pp. 209-219.

Fridrich, J. (1999). Methods for Tamper Detection in Digital Images. *Proc. ACM Workshop on Multimedia and Security*, pp. 19-23.

Fridrich, J., Goljan, M., & Du, R. (2001). Invertible Authentication. *in Proc. SPIE Security and Watermarking of Multimedia Contents III*, (San Jose, California, USA), vol. 3971, pp. 197-208.

Fridrich, J., Goljan, M., & Du, R. (2002) Lossless Data Embedding — New Paradigm in DigitalWatermarking. *EURASIP Journal on Applied Signal Processing*. 2002:2, pp. 185–196.

Friedman, G. L. (1993). The Trustworthy Digital Camera: Restoring Credibility to the Photographic Image, *IEEE Trans. Consumer Electronics*, vol. 39, no. 4, pp. 905-910.

Fu, M. S., & Au, O. C. (2000). Data Hiding by Smart Pair Toggling for Halftone Images, *IEEE Int. Conf. Acoustics Speech and Signal Processing*, vol. 4, pp. 2318–2321.

Golomb, S. W. (1966). Run-Length Encodings. *IEEE T. Inform. Theory*, vol. IT-12, pp. 399–401.

Holliman, M., & Memon, N. (2000). Counterfeiting Attacks on Oblivious Block-wise Independent Invisible Watermarking Schemes. *IEEE Trans. Image Processing*, vol. 9. no. 3, pp. 432-441.

Honsinger, C. W., Jones, P. W., Rabbani, M., & Stoffel, J. C. (2001). Lossless Recovery of an Original Image Containing Embedded Data. *US Patent* #6,278,791.

Howard, P. G., Kossentini, F., Martins, B., Forchhammer, S., & Rucklidge, W. J. (1998). The Emerging JBIG2 Standard. *IEEE Trans. Circ. Syst. Video Tech.*, vol. 8, no. 7, pp. 838–848.

ISO (1999) *Information Technology — Coded Representation of Picture and Audio Information — Lossy/Lossless Coding of Bi-Level Images*. Final Committee Draft for ISO/IEC International Standard 14492. Available at http://www.jpeg.org/public/fcd14492.pdf.

Lan, T. H., Mansour, M. F. & Tewfik, A. H. (2001). Robust High Capacity Data Embedding. *IEEE Int. Conf. Acoustics Speech and Signal Processing*, vol. 1, pp. 581-584.

Kim, H. Y., & Afif, A. (2004a). Secure Authentication Watermarking for Halftone and Binary Images. *Int. J. Imaging Systems and Technology*, vol. 14, no. 4, pp. 147–152.

Kim, H. Y., & de Queiroz, R. L. (2004b). Alteration-Locating Authentication Watermarking for Binary Images. *Int. Workshop on Digital Watermarking 2004, (Seoul),* Lecture Notes in Computer Science 3304, pp. 125–136.

Kim, H. Y. (2005). A New Public-Key Authentication Watermarking for Binary Document Images Resistant to Parity Attacks. *IEEE Int. Conf. on Image Processing*, vol. 2, pp. 1074–1077.

Kim, H. Y., & Mayer, J. (2007). Data Hiding for Binary Documents Robust to Print-Scan, Photocopy and Geometric Distortions. *To appear in Proc. Simpósio Bras. Comp. Gráfica e Proc. Imagens*.

Knuth, D. E. (1987) "Digital Halftones by Dot Diffusion," *ACM Trans. Graph.*, vol. 6, no. 4.

Kundur, D. & Hatzinakos, D. (1998). Towards a Telltale Watermarking Technique for Tamper-Proofing. *Proc. IEEE Int. Conf. On Image Processing*, vol. 2, pp. 409-413.

Kutter, M. (1998). Watermarking Resisting to Translation, Rotation, and Scaling. *SPIE Conf. Multimedia Syst. and App.*, vol. 3528, pp. 423-431.

Li, C. T., Lou, D. C. & Chen, T. H. (2000). Image Authentication and Integrity Verification via Content-Based Watermarks and a Public Key Cryptosystem. *IEEE Int. Conf. Image Processing*, vol. 3, pp. 694-697.

Lin, C. Y., Wu, M., Bloom, J. A., Cox, I. J., Miller, M. L., & Lui, Y. M. (2001). Rotation, Scale, and Translation Resilient Watermarking for Images. *IEEE T. Image Processing*, vol. 10, no. 5, pp. 767-782.

Lin, C.-Y. & Chang, S.-F. (2000a). Semi-Fragile Watermarking for Authenticating JPEG Visual Content. *Proc. SPIE Int. Soc. Opt. Eng*, vol. SPIE-3971, pp. 140-151.

Lin, E., Podilchuk C. & Delp E. (2000b). Detection of Image Alterations Using Semi-Fragile Watermarks, *Proc. SPIE Int. Conf. on Security and Watermarking of Multimedia Contents II*, vol. 3971.

Lin, C.-Y. & Chang, S.-F. (2001). A Robust Image Authentication Method Distinguishing JPEG Compression from Malicious Manipulation. *IEEE Trans. Circuits and Systems of Video Technology*, vol. 11, no. 2, pp. 153-168.

Lu, C.-S., & Hsu, C.-Y. (2005). Geometric Distortion-Resilient Image Hashing Scheme and Its Applications on Copy Detection and Authentication. *Multimedia Systems*, vol. 11, no. 2, pp. 159–173.

McGrew, D. A. & Viega, J. (2005). *The Galois/Counter Mode of Operation (GCM)*, NIST Draft Special Publication 800-38D. Available at http://csrc.nist.gov/CryptoToolkit/modes/proposedmodes/gcm/gcm-revised-spec.pdf.

Marvel, L. M., Boncelet Jr., C. G., & Retter, C. T. (1999). Spread Spectrum Image Steganography, *IEEE Trans. Image Processing*, vol. 8, no. 8, pp. 1075-1083.

Marvel, L. M., Hartwig Jr, G. W., Boncelet Jr, C. (2000). Compression Compatible Fragile and Semifragile Tamper Detection. *Proc. SPIE International Conference on Security and Watermarking of Multimedia Contents II*, vol. 3971.

Maxemchuk, N. F., & Low, S. (1997). Marking Text Documents. *IEEE Int. Conf. Image Processing*, vol. 3, pp. 13-17.

Mei, Q., Wong, E. K., & Memon, N. (2001). Data Hiding in Binary Text Documents. *Proceedings of SPIE*, vol. 4314, pp. 369–375.

Miyaji, A., Nakabayashi, M., & Takano, S. (2001). New explicit conditions of elliptic curve traces for FR-reduction. *IEICE Trans. on Fundamentals*, E84-A(5):1234–1243.

Ni, Z. C., Shi, Y. Q., Ansari, N., Su, W., Sun, Q. B., Lin, X. (2004). *Robust Lossless Image Data Hiding*, *IEEE Int. Conf. Multimedia and Expo 2004*, pp. 2199-2202.

NIST (2005). Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication. Special Publication 800-38B. Available at http://csrc.nist.gov/CryptoToolkit/modes/800-38_Series_Publications/SP800-38B.pdf.

Pamboukian S. V. D., & Kim, H. Y. (2006). Reversible Data Hiding and Reversible Authentication Watermarking for Binary Images. *Proc. Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*. Available at http://www.lps.usp.br/~hae/sbseg2006-rdtc.pdf

Pamboukian, S. V. D., & Kim, H. Y. (2005). New Public-Key Authentication Watermarking for JBIG2 Resistant to Parity Attacks. *Int. Workshop on Digital Watermarking 2005, (Siena),* Lecture Notes in Computer Science 3710, pp. 286–298.

Pereira, S., Ruanaidh, J. J. K. O., Deguillaume, F., Csurka, G., & Pun, T. (1999). Template Based Recovery of Fourier-Based Watermarks Using Log-Polar and Log-Log Maps. *IEEE Int. Conf. Multimedia Comp. Systems*, vol. 1, pp. 870-874.

Pereira, S., & Pun, T. (2000), Robust Template Matching for Affine Resistant Image Watermarks, *IEEE Trans. Image Processing*. vol. 9, no. 6, pp. 1123-1129.

Roetling P. & Loce, R. (1994). Digital Halftoning. *Chapter 10 in Digital Image Processing Methods*, E. Dougherty Ed., Marcel Dekker, New York, NY, 1994.

Salomon, D. (2004). *Data Compression: The Complete Reference*, 3rd Edition, Springer.

Schneider, M., & Chang, S.-F. (1996). A Robust Content Based Digital Signature for Image Authentication. *IEEE Int. Conf. Image Processing*, vol. 3, pp. 227-230.

Schneier, B. (1996). *Applied Cryptography*, 2nd Edition, John Wiley & Sons.

Schnorr, C. P. (1991). Efficient Signature Generation for Smart Cards. *Journal of Cryptology* 4(3), pp. 161–174.

Shi, Y. Q. (2004). Reversible Data Hiding. *Int. Workshop on Digital Watermarking 2004, (Seoul), Lecture Notes in Computer Science*, 3304, pp. 1-13.

Stinson, D. (2002). *Cryptography: Theory and Practice*, 2nd Edition. Chapman & Hall/CRC.

Tian, J. (2002). Wavelet-Based Reversible Watermarking for Authentication. *in Proc. SPIE Security and Watermarking of Multimedia Contents IV*. vol. 4675, pp. 679-690.

Tian, J. (2003). Reversible Data Embedding Using Difference Expansion. *IEEE Trans. Circuits Systems and Video Technology*, vol. 13, no. 8, pp. 890-896.

Tseng, Y. C., Chen, Y. Y., & Pan, H. K. (2002). A Secure Data Hiding Scheme for Binary Images. *IEEE Trans. on Communications*, vol. 50, no. 8, pp.1227–1231.

Ulichney, R. (1987). *Digital Halftoning*. The MIT Press.

Wong, P. W. (1998). A Public Key Watermark for Image Verification and Authentication. *IEEE Int. Conference on Image Processing*, vol. 1, pp. 455-459.

Wu, M., & Liu, B. (2004). Data Hiding in Binary Image for Authentication and Annotation. *IEEE Trans. on Multimedia*, vol. 6, no. 4, pp. 528–538.

Yeung, M. M. & Mintzer, F. (1997). An Invisible Watermarking Technique for Image Verification. *IEEE Int. Conf. Image Processing*, vol. 1, pp. 680-683.

Yu, G. J., Lu, C. S. & Liao, H. Y. M. (2001). Mean-Quantization-Based Fragile Watermarking for Image Authentication. *Optical Engineering*. vol. 40, no. 7, pp. 1396-1408.

Zhang, F., Safavi-Naini, R., & Susilo, W. (2004). An Efficient Signature Scheme from Bilinear Pairings and Its Applications. *Practice and Theory in Public Key Cryptography – PKC'2004*, Lecture Notes in Computer Science 2947, pp. 277–290.

Zhao, J. & Koch, E. (1995). Embedding Robust Labels into Images for Copyright Protection. *Int. Cong. Intellectual Property Rights, Knowledge and New Technologies*, pp. 242-251.

Additional Reading

Most of the interesting reading materials were referenced in the main text. Section 3 and 4 use many criptographic techniques. For readers not familiar with criptography, we suggest reading (Stinson, 2002; Schneier, 1996). We suggest reading (Barreto, 2002; Celik, 2002b; Holliman, 2000) for more details on topological AW. Section 7 makes use of data compression techniques and Salomon (2004) presents a good introductory text on this subject. In Section 8, we said that DH for hardcopy binary documents is an open issue. Probably, future hardcopy DH techniques will make use of the ideas developed for watermarking continuous-tone images resistant to rotation, translation and scaling. Papers (Pereira, 1999; Lin, 2001; Kutter, 1998; Chun, 2003) are some of the interesting references on this subject.