Secure Authentication Watermarking for 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, Brasil.
hae@lps.usp.br, amir@usp.br

Abstract. Authentication watermarking is a hidden data inserted into an image, in order to detect any alterations. It seems to be almost impossible to design a really secure authentication watermarking without making use of the solid cryptography theory and techniques. In a cryptography-based authentication watermarking, a message authentication code (or digital signature) of the whole image is computed and the resulting code is inserted into the image itself. However, inserting the code alters the image and consequently its authentication code, invalidating the watermark. To avoid this problem, for gray-scale or color image, usually the least significant bits (LSBs) are cleared, the authentication code of the LSB-cleared image is computed and then the code is inserted into LSBs. Surely, one cannot perform the same procedure for binary images. This paper proposes a quite simple solution for inserting a secure authentication watermarking in dispersed-dot halftone images. This technique can also be applied to any kind of binary images (including clustered-dot halftones), though the visual quality is not as good as when applied to dispersed-dot halftones. The proposed technique can be used with both secret-key or public-key ciphers.

1. Introduction

A watermark is a signal added to digital data (audio, video and still images) that can be detected or extracted later to make an assertion about the data. In this paper, we will be concerned only with still images. Embedding a watermark in an image means inserting information in the image, such that the image quality does not deteriorate significantly.

A watermark can be visible or invisible. A visible watermark typically contains a visual message or a company logo that indicates the ownership of the image. An invisible watermarked image is visually very similar to the original image. The existence of such a watermark can be determined only through a watermark extraction or detection algorithm. Invisible 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 image integrity can be fragile because if the watermark is removed, the watermark detection algorithm will report the corruption of the image.

Halftoning is a process to convert a grayscale image G into 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. Halftone images can be dispersed-dot or clustered-dot. Usually, dispersed-dot has a better visual quality, but some devices cannot reproduce too-finely-dispersed dots (like laser printer) and they must use clustered-dot halftoning.

In the literature, there are many authentication watermarking techniques for continuous-tone images (grayscale and color) [Zhao, 1995; Yeung, 1997; Wong, 1997; Wong 1998; Li, 2000; Barreto, 1999; Holliman, 2000; Barreto, 2002]. However, it seems to be almost impossible to design a really secure authentication watermarking without making use of the solid cryptography theory and techniques. Indeed, those watermarking techniques that were not founded in cryptography theory [Zhao, 1995; Yeung, 1997] or those that applied incorrectly cryptographic techniques [Wong, 1997; Wong, 1998; Li, 2000] were later shown to be unreliable [Holliman, 2000; Barreto, 2002].

In cryptography-based authentication watermarking, a message authentication code (MAC) or 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 watermarking. To avoid this problem, for gray-scale and color images, usually the least significant bits (LSBs) are cleared, MAC/DS of the LSB-cleared image is computed and then the code is inserted into LSBs. In other words, those bits where the watermark is to be inserted are not taken into account when computing MAC/DS.

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, 1998; Chen, 2000; Wu, 2000; Fu, 2000; Fu, 2002a], we have no tidings of an cryptography-based authentication watermarking for binary and halftone images. Fu and Au [Fu, 2002b] present a watermarking 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 watermarking algorithm may be public without compromising the security of the scheme. In this paper, we propose an authentication watermarking that complies with this requirement. It is especially suitable for dispersed-dot halftone images. It can be applied also to any kind of binary images (including clustered-dot halftones), though the visual quality is not as good as to dispersed-dot halftones. The idea is quite simple. However, it seems to be a new idea. The method can be used in conjunction with symmetric cipher (secret-key cryptography) or asymmetric cipher (public-key cryptography). The technique is intended to authenticate only digital binary images. Printed images cannot be authenticated using the proposed technique.

A possible use of the proposed technique (in conjunction with a public-key cryptography) is in the trusted FAX. Using trusted FAX, the receiver of a document can certify the originator of the document, and that it has not been altered (accidentally or maliciously) during the transmission.

2. Data Hiding in Binary and Halftone Images

There are many papers in the literature on information embedding in binary and halftone images.

Two basic ways to embed data in binary image are by changing the values of individual pixels [Wu, 2000; Tseng, 2002] and by changing a group of pixels [Maxemchuk, 1997]. The first approach toggles a black pixel to white or vice versa. The second approach modifies such features as a thickness of strokes, curvature, relative positions, etc. This approach generally depends more on the types of images. Since the number of parameters that can be changed in this manner is limited, especially under the requirements of blind detection and invisibility, the amount of data that can be hidden is usually limited except for especial types of images.

For data hiding in halftone images, some works use two different dithering matrices for the halftone generation [Baharav, 1998; Hel-Or, 2001] such that the different statistical properties due to the two dithering matrices can be detected in the future. Other works use stochastic screen patterns [Fu, 2001] in which two screens are used to form two halftone images and the data are embedded through correlation between two screens. The embedded pattern cannot be recovered with only one halftone image. It can be viewed only when two images halftone are overlaid.

DHST (data hiding by self toggling) is another data hiding technique for halftone images, and it is especially interesting for its simplicity [Fu, 2000; Fu, 2002]. 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 pseudo-random locations within the image. Then one bit is embedded in each location by forcing it to be either black or white. With probability 0.5, the pixel of the original halftone is the desired value and thus no change is needed. And with probability 0.5 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 known seed to obtain the pseudo-random locations. Then the pixel values at those locations can be read easily. Evidently, DHST can also be used in any binary image. However, in this case, a visible salt-and-pepper noise will appear. In this paper, we will convert DHST into a cryptographically secure fragile authentication watermarking.

Fu and Au present also many improvements to this basic idea to enhance the visual quality of the image: DHPT (data hiding by pair toggling) and DHSPT (data hiding by smart pair toggling) [Fu, 2002]. The underlying idea of these improvements are 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.

3. Authentication Watermark for Contone Images

In the literature, most works on authentication watermarking are designed for continuous-tone (contone) images. Their goal is not only detect alterations in image but also to locate them. Papers [Holliman, 2000; Barreto, 2002] analyze thoroughly the requirements to obtain reliable alteration locating authentication watermarking. However, in the present paper, we will be satisfied by simply detecting the alteration, without the requirement of locating them. The reason for this simplification is that binary image presents smaller data hiding capacity than contone images.

A cryptography-based authentication watermarking [Wong, 1998; Holliman, 2000; Barreto, 2002] typically performs the following operations for the watermarking 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 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 digital signature *S*.
5. Insert *S* into the LSBs of *Z**, obtaining the marked image Z'.

The watermarking verifying algorithm is:

1. Let *X*' be a watermarked image. Let *X** be the image obtained from *X*' by clearing the LSB of all pixels. Using the same hashing function *H* chosen for insertion, compute the 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 noise-like 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 entirely white or black image, without compromising the security.

4. The Proposed Method

In secure authentication watermarking using some data hiding technique for binary image, one must compute a hashing function of the binary image *B*, obtaining the fingerprint $H = H(B)$. This fingerprint *H*, after the exclusive-or operation and encryption, becomes the digital signature *S*. This digital signature must be inserted into *B* itself, obtaining the marked image *B*'. The problem is that, with the insertion of watermark, the image *B* changes and consequently its fingerprint changes. That is, $H(B) \neq H(B')$. How can we overcome this difficulty?

We present a very simple solution using DHST. Differently from most binary image data hiding techniques, in DHST only a few bits are modified and the positions of those bits 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 image. Let us call the obtained technique AWST (authentication watermarking by self toggling). The AWST insertion algorithm is:

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 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 a digital signature *S*.
7. Insert *S* in the set of pixels *L*, generating the marked image *B*'.

The AWST extraction algorithm is:

1. Let *X*' be a watermarked image. Use the same pseudo-random number generator and seed used in watermark insertion to generate again the same set of 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.

Figure 1 illustrates the use of AWST watermarking scheme. Let's suppose that image *B* (figure 1a) is a sensitive image to be transmitted through an unreliable channel, where unintentional or intentional alterations may occur. To protect *B*, logo image *A* (figure 1b) was inserted into *A* using the AWST algorithm. The image *B*' (figure 1c) is the watermarked image where 1024 bits were embedded. This is enough to embed a RSA digital signature [Schneier, 1996]. If the extraction algorithm is performed, we obtain the check image *C* (figure 1d), exactly equal to the logo image *A*. If even a single pixel of *B*' is altered, the extracted image is completely noisy (figure 1f).

Figure 2 depicts the quality of a AWST-watermarked document. A page of a magazine was scanned at 300 dpi, resulting a binary image with 3318 rows and 2536 columns (figure 2a). Figures 2b, 2c and 2d shows respectively the document with 64 bits, 320 bits and 1024 bits embedded. These quantity of bits are enough to insert, respectively, a secret-key message authentication code, public-key DSA signature and public-key RSA signature.

5. 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 5.1, 5.2 and 5.3.

Another possible variation is a watermarking scheme that does not use the logo image *A*. In this case, the detection algorithm will not extract the check image *C*, but it will answer a Boolean question: the image has or has not been altered (subsection 5.4).

The third possible variation is to use improved data hiding techniques DHPT and DHSPT to insert the watermarking, instead of using DHST (subsection 5.5).

5.1 Keyless AWST

Keyless authentication watermarks are useful for detecting unintentional alterations in image such as cropping and distortion due to dirt or human writing/marking. They are a sort of "check-sum". If watermarking insertion and detection algorithms are made public, anyone can insert and verify keyless authentication watermarks. In 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 hashing collision.

5.2 Secret-Key AWST

Secret-key fragile watermarks can be used to detect any alteration in image, even intentional or malicious ones. This kind of watermarking is similar to well-known message authentication codes, the only difference is that the authentication codes are inserted into an 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 public or kept secret, 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. Let us suppose that Mallory, a malicious active attacker, modifies one image in the database. Mallory cannot insert the correct watermark in the modified image because he does not know *k*. Moreover, Alice will be able to detect all images that was altered by Mallory using the AWST extraction algorithm and her secret-key *k*.

Some variations can be made in the AWST algorithm. For example, Wong [Wong, 1997] does not encrypt the fingerprint $\hat{H}$ using a symmetric cipher (step 6 of AWST insertion algorithm). Instead, he feeds the hashing function with the secret-key, that is, the step 4 of AWST insertion algorithm becomes $H = H(k, B^*)$. AWST extraction algorithm must be altered accordingly.

Typically, a 64-bits wide message authentication code (MAC) is considered cryptographically secure. The introductory book on cryptography [Schneier, 1996] expounds many MAC schemes.

5.3 Public-Key AWST

Public-key authentication watermarks [Wong, 1998; Barreto, 2002] use public-key cryptography [Schneier, 1996] to insert a digital signature into the 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 active attacker, 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 image 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 AWST watermark. The FAX receiver knows the public-key of the sending machine. Thus, the receiver can verify that the document was originated by a specific FAX machine 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 public-key AWST is in legal usage of binary documents. If documents are transmitted through internet, it is important that the receiver ensures that the document is signed by a specific person and that no alteration has occurred after the signature.

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

A public-key digital signature can be considered secure using 1024-bits RSA signature or 320-bits DSA signature [Schneier, 1996].

5.4 Boolean Answer

Although extracting a visible logo from a watermarked image may be appealing, actually 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 the step 5 from the AWST insertion algorithm and the step 4 from the AWST extraction algorithm.

5.5 Keeping Average Gray-Level

The visual quality of an AWST-marked dispersed-dot halftone image can be improved by using improved data hiding techniques DHPT and DHSPT [Au, 2002], 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.

However, to implement this scheme, no neighbor of 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 neighbors of a selected pseudo-random pixel, this alteration will not be detected by the AWST scheme.

6. Conclusions

This paper has proposed a cryptographically secure authentication watermarking for binary images and we named it AWST. The technique is suitable for watermarking dispersed-dot halftone images. However, as the necessary quantity of pixels to be altered is very small, the technique can be applied to any binary image without causing a noticeable loss of quality. The proposed technique can be used in three ways: keyless, secret-key and public-key. Public-key AWST is the most useful. Public-key AWST can be used in trusted FAX machines, to electronically sign binary documents, to protect a database of sensitive documents, etc.

7. Acknowledgements

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

8. References

[Baharav, 1998] Z. Baharav, D. Shaked, "Watermarking of Dither Halftone Images", Hewlett-Packard Labs. Tech. Rep. HPL-98-32 (1998).

[Barreto, 1999] P. S. L. M. Barreto, and H. Y. Kim, "Pitfalls in Public Key Watermarking," *Sibgrapi - Brazilian Symp. Computer Graphics and Image Processing*, 1999, pp. 241-242.

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

[Chen, 2000] Y.-Y. Chen, H.-K. Pan, and Y.-C. Tseng, "A Secure Data Hiding Scheme for Binary Images," *IEEE Symposium on Computers and Communications*, 2000, pp. 750-755.

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

[Fu, 2001] M. S. Fu, O. C. Au, "Data Hiding in Halftone Images by Stochastic Error Diffusion", *IEEE Int. Conf. Acoustics, Speech and Signal Processing*, May 2001.

[Fu, 2002a] M. S. Fu, and O. C. Au, "Data Hiding Watermarking for Halftone Images," *IEEE Trans. Image Processing*, vol. 11, no. 4, pp. 477- 484, 2002.

[Fu, 2002b] M. S. Fu, and O. C. Au, "A Robust Public Watermark for Halftone Images," *IEEE Int. Symp. Circuits* *and Systems*, vol. 3, pp. 639-642.

[Hel-Or, 2001] H. Z. Hel-Or, “Watermarking and Copyright Labeling of Printed Images,” Journal of Electronic Imaging, col. 10, no. 3, pp. 794-803, 2001.

[Holliman, 2000] M. Holliman, and N. Memon “Counterfeiting Attacks on Oblivious Block-wise Independent Invisible Watermarking Schemes,” *IEEE Trans. Image Processing*, 2000, vol. 9. no. 3, pp. 432-441.

[Knuth, 1987] D. E. Knuth, “Digital Halftones by Dot Diffusion,” ACM Trans. Graph., vol. 6, no. 4, Oct. 1987.

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

[Maxemchuk, 1997] N. F. Maxemchuk, and S. Low, “Marking Text Documents,” *Int. Conf. Image Processing*, vol. 3, pp. 13-17, 1997.

[Schneier, 1996] B. Schneier, *Applied Cryptography*, second edition, John Wiley & Sons, 1996.

[Tseng, 2002] Y.-C. Tseng, Y.-Y. Chen, and H.-K. Pan, “A Secure Data Hiding Scheme for Binary Images,” *IEEE Trans. on Communications*, Vol. 50, No. 8, Aug. 2002, pp. 1227-31.

[Ulichney, 1987] R. Ulichney, *Digital Halftoning*, The MIT Press, 1987.

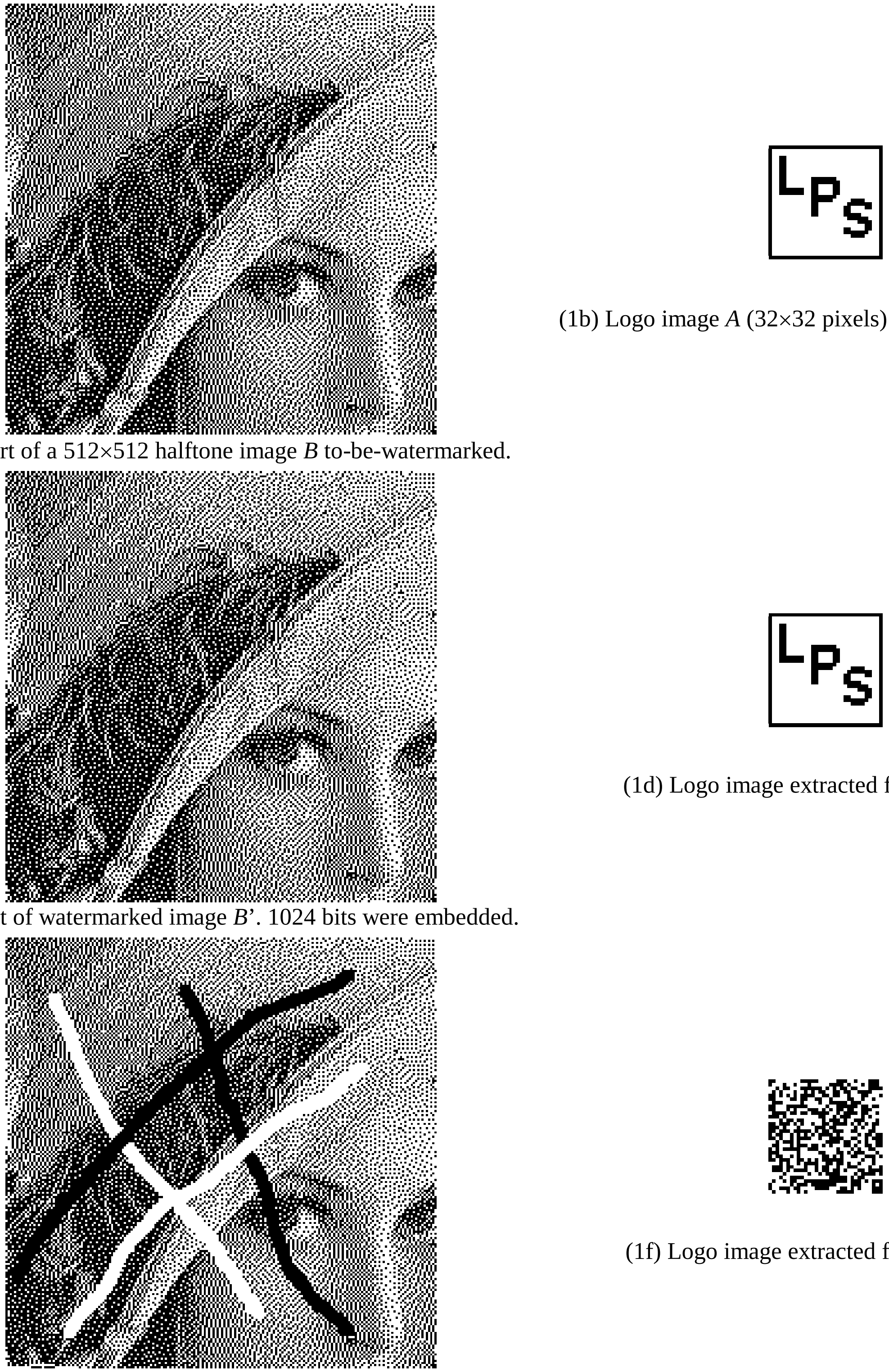
[Wong, 1997] P. W. Wong, “A Watermark for Image Integrity and Ownership Verification,” *IS&T PIC Conference*, (Portland, OR), May 1998 (also available as Hewlett-Packard Labs. Tech. Rep. HPL-97-72, May 1997).

[Wong, 1998] P. W. Wong, “A Public Key Watermark for Image Verification and Authentication,” *IEEE Int. Conf. Image Processing*, 1998, vol. 1, pp. 455-459, (MA11.07).

[Wu, 2000] M. Wu, E. Tang, and B. Liu, “Data Hiding in Digital Binary Image,” *IEEE Int. Conf. Multimedia and Expo*, ICME’00, New York, USA, 2000.

[Yeung, 1997] M. M. Yeung, and F. Mintzer, “An Invisible Watermarking Technique for Image Verification,” *IEEE Int. Conf. Image Processing*, 1997, vol. 1, pp. 680-683.

[Zhao, 1995] J. Zhao and E. Koch, “Embedding Robust Labels into Images for Copyright Protection,” *Proc. Int. Cong. Intellectual Property Rights, Knowledge and New Technologies*, 1995, pp. 242-251.



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

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

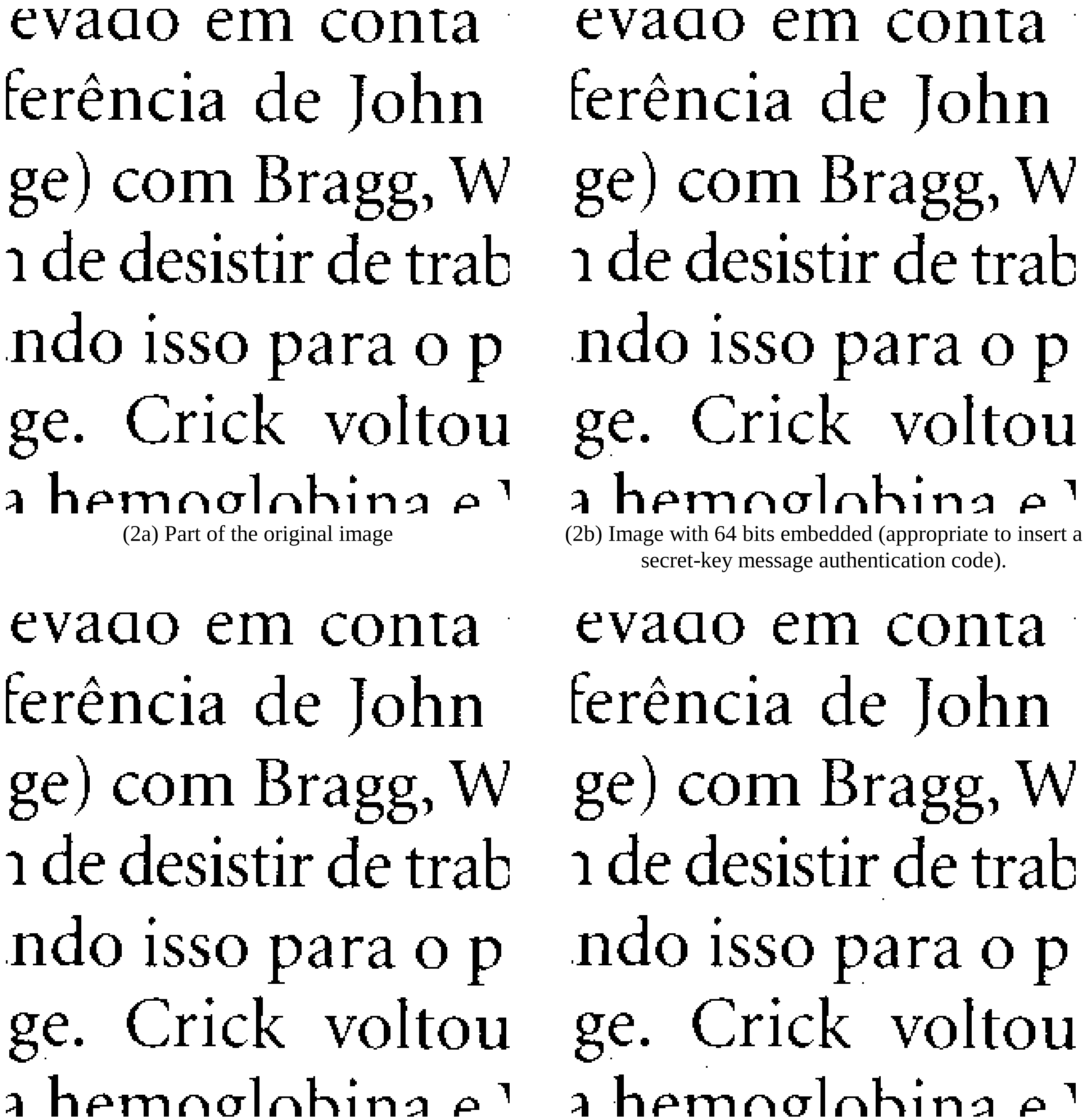
(1c) Part of watermarked image *B*’. 1024 bits were embedded.

(1d) Logo image extracted from image *B*’.

(1e) Part of the modified image *X*’.

(1f) Logo image extracted from image *X*’.

Fig. 1: Illustration of public-key AWST. The logo image *A* (1b) was inserted into image *B* (1a) using public-key cipher. Figure (1c) depicts the watermarked image. Executing the watermark extraction algorithm, figure (1d) is obtained. If the watermarked image is modified even slightly (1e), a completely random image is extracted (1f).



evado em conta
ferência de John
ge) com Bragg, W
n de desistir de trab
ndo isso para o p
ge. Crick voltou
a hemoglobina e

(2a) Part of the original image

evado em conta
ferência de John
ge) com Bragg, W
n de desistir de trab
ndo isso para o p
ge. Crick voltou
a hemoglobina e

(2b) Image with 64 bits embedded (appropriate to insert a secret-key message authentication code).

evado em conta
ferência de John
ge) com Bragg, W
n de desistir de trab
ndo isso para o p
ge. Crick voltou
a hemoglobina e

(2c) Image with 320 bits embedded (appropriate to insert public-key DSA signature).

evado em conta
ferência de John
ge) com Bragg, W
n de desistir de trab
ndo isso para o p
ge. Crick voltou
a hemoglobina e

(2d) Image with 1024 bits embedded (appropriate to insert public-key RSA signature).

Fig. 2: Illustration of the AWST watermarked document image quality. A page of a magazine was scanned at 300 dpi, resulting a binary image with 3318 rows and 2536 columns (2a). AWST watermark using secret-key needs to embed 64 bits in image (2b). Using DSA public-key signature, 320-bits embedding is necessary (2c). Using RSA public-key signature, 1024 bits is required to be embedded (2d). Note that the image degradation is low even embedding 1024 bits.