A Transparent Encryption Scheme for Watermarked Biometric and Medical Images

Aniketh Talwai, Debabrata Sengupta, and Kannan Karthik

Abstract—**In today's digital world, maintaining electronic medical and biometric records is an effective way to store, disseminate, and analyze information for a person. However, protection of privacy of the person is also of paramount importance. In this paper, we propose an encryption scheme which is transparent to watermarking. Encryption is used to secure the contents of the images while watermarking can be used for tagging those images for effective indexing, retrieval and other operations. By making encryption transparent, we ensure that the information in the watermarked 'tags' can be accessed without having to decrypt and disclose the content of the images. The watermark is embedded in the quantized Discrete Cosine Transform (DCT) coefficients during image compression while encryption is carried out by using a combination of transposition and a modified Hill cipher. Experimental results confirm that the watermark can be read with complete success from sufficiently encrypted images.**

Index Terms—**Hill cipher, image watermarking, privacy enhanced image indexing, transparent encryption.**

I. Introduction

Protecting the privacy of medical and biometric data in electronic databases while enabling easy indexing, access and distribution is a pressing problem. For example, medical images of patients are often encrypted to preserve privacy but watermarks embedded in the image need to be accessed in the encrypted domain for tagging and related operations. In this paper we propose a novel method of transparent image encryption to achieve this objective. The medical image may be 'tagged' by embedding a watermark in the image, containing essential information. Due to transparent nature of the proposed encryption, we ensure that this 'tag' can be read in the encrypted domain without revealing the content of the image, thereby safeguarding privacy.

There are only a limited number of publications dealing with tagging and securing medical images by making the process of encryption transparent to watermarking. A detailed survey of watermarking techniques, including watermarking in the transform domain by LSB modulation is available in [1]. In [2], the authors tackle a similar problem as us, however there are two significant differences - in our case watermarking is done in the transform domain and not in the spatial domain so that the watermark gets distributed over several pixels in the image; we also propose making encryption transparent to watermarking so that the watermark can be read in the encrypted domain. In [3], the authors make encryption and watermarking commutative by encrypting and watermarking two independent parameters of the data stream. Although our method is not commutative in the strictest sense, its merit lies in the fact that a significantly large portion of the data stream can be encrypted and watermarked.

The rest of the paper is organized as follows. In section II, we define the problem we are tackling and our basic assumptions. The watermarking and encryption procedures are described in sections III and IV. We prove that our encryption algorithm is transparent to watermarking in section V. Simulation results are presented in section VI. In section VII, conclusions are drawn and future work is presented.

Manuscript received April 5, 2012; revised May 27, 2012.

The authors are with the Department of Electronics and Electrical Engineering, IIT Guwahati. He is now with IIM Ahmedabad. (e-mail: aniketh@iitg.ernet.in, dsgupta@stanford.edu, k.karthik@iitg.ernet.in)

II. Problem Formulation

In this paper, we tackle the problem of transparent encryption of images. The watermark is embedded in the quantized Discrete Cosine Transform (DCT) coefficients of the image during image compression, following which these watermarked coefficients are encrypted. Separate keys are used during watermarking and encryption. Our goals are: (i) to ensure that watermarking does not change the image perceptually (ii) to design the watermarking and encryption processes such that the watermark can be read from the encrypted image directly, without having to decrypt it. The entire process can be summarized in the form of a block diagram as shown in Fig. 1.

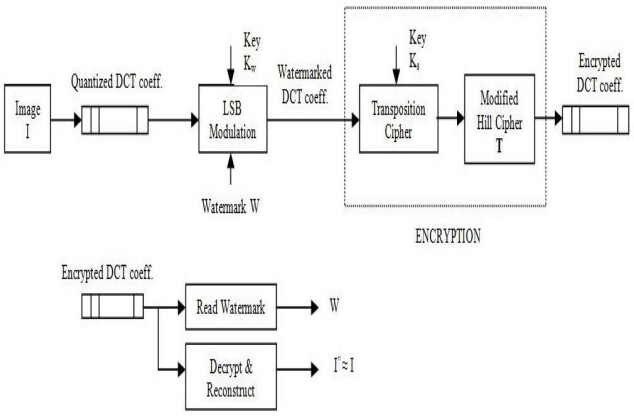

Fig 1. Block diagram showing the processes of watermarking, encryption, reading the watermark and decryption.

Our basic assumptions in this paper are: (i) the elements of the data stream (DCT coefficients) are 8 bit integers, (ii) watermarking and encryption are controlled by separate keys, (iii) all the keys, including the algebraic cipher matrix have been exchanged securely between the transmitter and the

receiver using any key-exchange algorithm, (iv) the channel is noise free, or if noisy, there exist suitable error correcting codes to detect and correct all the errors, (v) that the images we are working with are biometric and medical images and hence possible forms of attack include trying to reveal the contents of the original image , or that of the watermark. Deliberate attacks to destroy the watermark are not tackled in this paper since the main aim of the attacker in this application would be to read, not remove, the watermark.

III. Watermarking Procedure

To watermark the given image, we perform least significant bit (LSB) modulation of certain key-determined DCT coefficients of the image obtained during compression. It was observed that toggling the LSBs of DCT coefficients does not produce any significant perceptual change in the image and hence can be used safely without corrupting the original image. As discussed previously, we do not envisage attacks aimed at totally destroying the watermark in this particular application, hence the question of the attacker forcibly setting all LSBs to zero (or one) does not arise.

A. Description

Let the data stream (comprising of quantized DCT coefficients) be denoted by *DS*, the key by K_w, and the watermark to be embedded by *W*. The locations at which the bits would be toggled is kept as a function of both the data stream and the key. First, a difference vector is created by subtracting certain elements (denoted by *E*) of the data stream from the key (parsed into 8 bit subsequences). The decision as to which elements would be used is determined both by the key and a key-selected portion of the data stream (denoted by *Q*). Hence the elements used in subtraction as well as the above mentioned key-selected portion are kept unavailable for watermarking and encryption. However the latter (*Q*), being a user defined system parameter, can be kept small and the former (*E*) is as long as the parsed key. Hence the overwhelming majority of the data stream can still be watermarked and later encrypted. The difference vector is used to generate a seed for a function which generates a pattern which determines the locations of the data stream where the LSBs have to be toggled. Finally, if the watermark bit is 0 (or 1), the corresponding location of the data stream is made even (or odd respectively). It is ensured that the location of the data stream which is watermarked is different from the portions of the stream used to calculate the difference vector.

In order to read the watermark at the receiver, the difference vector is re-created by subtracting the key (parsed into 8 bit subsequences) and the elements *E*, the locations of which are determined by the key and *Q*. Then, as before, the difference vector is used to generate a seed for the same function which returns the locations at which the watermark is embedded. The watermark is then read by checking whether the data stream at the corresponding location is even, indicating 0, or odd, indicating 1.

B. Analysis of Watermarking Scheme

A brute force attacker will have to generate all possible difference vectors corresponding to all possible key values. It must also be noted that this operation will need to be repeated with every new data stream even if the same key is being used, and the additional time taken to do this depends on both the run time of the function to generate the difference vector and the key length. This ensures that the system is more robust to brute force attacks and that the same key can be used to watermark multiple data streams.

Let T_F be the time taken to generate a difference vector for a key of length *k* bits, T_G be time taken to calculate watermark locations given all necessary parameters, T_P be the time for all other operations involved in watermarking. If we watermark *N* streams, the average time T_1 taken for a brute force attack to successfully read our watermark is

$$T_1 = (2^k T_G + 2^k T_F + T_P) N. \quad (1)$$

Whereas, if the difference vector were determined only by the key, the time T_2 for a brute force attack to succeed would be independent of N, given as

$$T_2 = 2^k T_G + T_P. \quad (2)$$

IV. Encryption Procedure

From the above discussion, it is evident that the watermark information is captured by the odd or even nature of the watermarked DCT coefficients. Hence, in order to preserve the watermark even after encryption, we design the encryption process so that the odd or even nature of coefficients does not change. As noted earlier (in section III A), a negligibly small part of the data stream is unavailable for encryption.

An algebraic substitution cipher, like the Hill matrix [4], is vulnerable to known plaintext attacks. For a *n* x *n* cipher matrix, the attacker needs to know *n* blocks (of length *n* each) of plaintext and the corresponding ciphertext to uniquely determine the cipher matrix. In order to tackle this issue, we use a two step approach for encryption.

A. Description

First, a transposition cipher is implemented. The data stream is parsed into blocks of length *p* and the elements of each block are permuted according to the key K_s. While swapping any two coefficients, all bits except for the LSBs of the corresponding coefficients are swapped. Thus, if we assume that the DCT coefficients are 8 bit integers, then only the first seven bits are swapped and the LSB is left untouched. This ensures that at a particular location, the even or odd nature of the coefficient is preserved before and after swapping. The order of permutation is decided by a random pattern generator to which the key is the initial seed and whose state changes as a function of the key. Thus, without knowing the key one cannot predict subsequent patterns from the current one.

The second step is the encryption of the swapped coefficients parsed into blocks of *n* by using a *n* x *n* key matrix to perform a linear transformation. The process can be represented in the matrix form as Y = (T X) mod 256, where X is the matrix formed from swapped and parsed coefficients, Y is the corresponding ciphertext matrix and mod 256 refers to the modulo 256 operation (since the elements are 8 bit integers). The *n* x *n* transformation matrix T is chosen such that only the elements in the leading diagonal are odd, while all other elements of the matrix are even. All elements of T,

T_{ij} belong to the set of integers {0,1,…,255}. Thus, T can be expressed as (2A + I)mod 256, where A is some *n* x *n* matrix of integers and I is the *n* x *n* identity matrix. The corresponding ciphertext elements in Y will also lie within the same range as that of X.

In order to decrypt the ciphertext matrix Y, the receiver will perform the operation $(T^{-1}Y)$ mod 256 to get back the plaintext matrix X. Then, the elements in X can be swapped back using the key K_s to the same positions that they previously occupied, thereby producing the decrypted (but watermarked) DCT coefficients. However, for decryption to be possible, the matrix T should be invertible. This issue is dealt with below.

B. Analysis of Encryption Scheme

Invertibility of T : The key matrix in our case is invertible if it satisfies the properties: (i) det T (mod 256) is non zero (ii) gcd(det T (mod 256), 256) = 1, i.e. det T (mod 256) and 256 are relatively prime [5].
Let T_n be a *n* x *n* square matrix such that

$$T_n = 2\, A_{n\,x\,n} + I_n. \tag{3}$$

Now, det T_n is given as

$$det\ T_n = T_{11}\ det\ T_{n-1} + T_{12}\ det\ C_{12} + \ldots + T_{1n}\ det\ C_{1n}, \tag{4}$$

where C_{ij} are the corresponding cofactor matrices. Since T_{1k} is odd only for k=1 and even otherwise, we have

$$det\ T_n = (2u+1)\ det\ T_{n-1} + 2r\,, \tag{5}$$

where *u* and *r* are integers. Thus, det T_n is odd if and only if det T_{n-1} is odd. Also, from (3)

$$det\ T_1 = 2a + 1 = \text{odd}. \tag{6}$$

Hence by induction, det T_n is odd, and so is det T_n (mod 256). Since 256 is a power of 2 while det T_n (mod 256) is odd, they must be relatively prime and hence both the properties mentioned before are satisfied. Thus, any T chosen in this fashion will always be invertible.

Bound on the block size for swapping: While permuting the blocks of *p* watermarked DCT coefficients using the key K_s, we want the key-space to be larger than the number of possible permutations. This will ensure that even if the permutation is known to the attacker, the key used to get this permutation cannot be uniquely determined. If we assume that the length of the key K_s is *k* bits and the number of inverse mappings (from permutation to key) for every permutation is α, then we must have

$$2^k/(p!) = \alpha. \tag{7}$$

This puts an upper bound on the allowable block size (*p*) used for permutation.

Security analysis of the encryption process: Through the permutation operation, we wish to ensure that *n* elements in any column of X do not appear in the same order as they do before swapping. An attacker will still be able to crack the cipher matrix T if he manages to get hold of *n* such linearly independent columns of X which do not get changed after permutation. Assuming that the key K_s (of length *k* bits) determines the choice of a permutation, the total number of possible permutations for a block of p elements is given as

$$\beta = \min\,(\,p!\,,\,2^k\,). \tag{8}$$

Without loss of generality, for our analysis we assume that both the blocks of size *n* and *p* start from the same element. If $n \le p$, then probability P_1 that *n* elements occur in the same order after permutation is given by

$$P_1 = (p-n)!\,/\,\beta \quad \text{for } n \le p. \tag{9}$$

For a general case, if $p < n \le tp$, the expression for *P* is

$$P_1 = [\ 1\ /\ (\beta)^{t-1}\ ]\ \text{x}\ [\ (p - \{n - (t-1)p\})\ !\ /\ \beta\ ]$$
$$= (tp-n)\ !\ /\ (\beta)^t \quad \text{for } p < n \le tp. \tag{10}$$

P_1 gives the probability that the attacker successfully obtains one column of X unchanged. He needs *n* such columns to decipher T. From this result, we observe that increasing *n* will decrease the value of P_1, thereby making it tougher to crack the cipher matrix. However, it must be noted that all the elements of T must be securely exchanged between the transmitter and receiver beforehand and constitutes the side-information available to the receiver. Thus, choosing a large value of *n* strengthens the cipher at the expense of having to provide more side information.

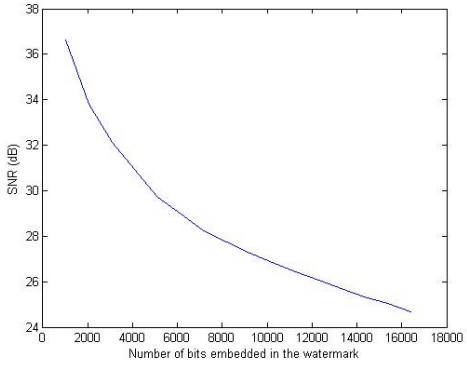

Fig. 2. Plot of SNR of watermarked image versus the number of bits embedded in the watermark

There are no constraints in arranging DCT coefficients and any arrangement could be valid, unlike the alphabet where letters are arranged as dictionary words. Hence, conventional methods to break a transposition cipher become difficult to apply in this scenario.

Secrecy of **T** : If the elements of the cipher matrix **T** are chosen as in (3), then each position of the matrix can be filled in 128 ways only. This reduces the randomness in each position by half. However, the probability P_2 of guessing all the n^2 elements of the key successfully for a brute force attack is given by

$$P_2 = [(1\ /\ 128)^n]^n. \tag{11}$$

For all reasonable values of *n* and *p*, the probability of a brute force attack being successful (P_2) is significantly lower than the probability P_1 in (10). Thus, restricting the choice of numbers for each position of the matrix **T** does not have any impact on the security of the cipher.

V. Proof of Transparency of Encryption

In this section, we prove that the embedded watermark is preserved during encryption and it can be read without having to perform decryption. Suppose that some particular element x_i at position *i* of the stream of DCT coefficients has been watermarked and has been made either even or odd by

LSB toggling. On performing permutation as the first step of encryption, let us say that some other element x_j takes the place of x_i. We perform swapping keeping the LSB untouched. Hence the odd or even nature of x_i has been preserved at the location *i*. If we now consider a block of *n*

TABLE I: VALUES OF DIFFERENT PARAMETERS FOR WATERMARKING AND ENCRYPTION OF IMAGES IN FIG.3

	Biometric Image	Medical Image
Size of Image	288 x 400	370 x 370
Size of watermarking key (K_w)	128 bits	128 bits
Number of watermark bits (as in Table 3)	62	62
Size of encryption key (K_s)	128 bits	128 bits
Block size for swapping (p)	6 elements	6 elements
Size of key matrix **T**	6 x 6	6 x 6

TABLE II: PSNR(IN DB) AT DIFFERENT STAGES FOR MEDICAL AND BIOMETRIC IMAGES IN FIG.3

	Watermarked Image	Encrypted Image	Decrypted Image
Medical Image	82.2770	18.1313	82.2770
Biometric Image	80.1121	11.2064	80.1121

TABLE III: WATERMARK GENERATION – NAME AND ID, CORRESPONDING DECIMAL AND BINARY REPRESENTATIONS

Data	B	o	b		L	e	E		I	D	-	123
Decimal Form	2	15	2	0	12	5	5	0	9	4	27	123
Binary Form	00010	01111	00010	00000	10010	00101	00101	00000	01001	00100	11011	1111011

TABLE IV: WATERMARK READ FROM THE ENCRYPTED IMAGE

Bits read	000100111100010000001001000101001010000001001001001101111111011											
Parsed bit seq.	00010	01111	00010	00000	10010	00101	00101	00000	01001	00100	11011	1111011
Decimal Form	2	15	2	0	12	5	5	0	9	4	27	123
Recover-ed Data	B	o	b		L	e	E		I	D	-	123

Coefficients $[x_1\ x_2\ \ldots\ x_n]^T$ to be encrypted by using the matrix T, we get the ciphertext element y_k corresponding to x_k as:

$$y_k = (T_{k1}x_1 + T_{k2}x_2 + \ldots + T_{kk}x_k + \ldots + T_{kn}x_n) \bmod 256. \quad (12)$$

We have constructed matrix T such that T_{ij}=odd only if $i=j$. Hence (12) can be expressed as

$$y_k = \{(2u+1)x_k + 2r(x_1 + x_2 + .. + x_{k-1} + x_{k+1} + .. + x_n)\} \bmod 256, \quad (13)$$

where *u* and *r* are integers. Hence, y_k is odd (or even) if and only if x_k is odd (or even). Since the odd or even nature of the original coefficients is preserved even after encryption, the watermark can be read from the encrypted coefficients in the same way as described in Section III.

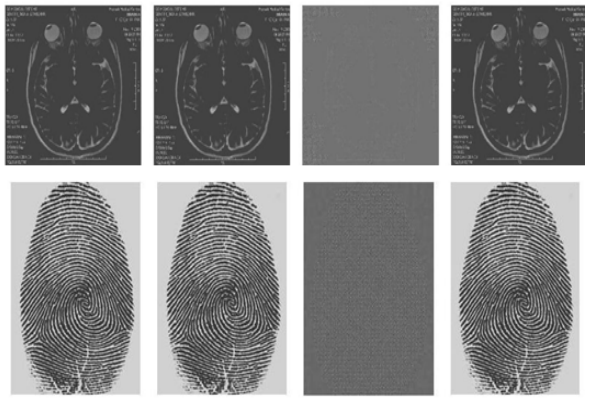

Fig. 3. (Top and Bottom) (a) Original image, (b) Watermarked image, (c) Encrypted image, (d) Decrypted image
[Online Images; source: www.londonink.com/wordpress, www.theaviationnation.com]

VI. SIMULATION RESULTS

The algorithm presented in this paper was implemented on medical and biometric images. For the watermarking scheme, data of varying lengths was embedded in a 288x400 image of a fingerprint and the resulting distortion (in terms of SNR) was analyzed. A plot between the SNR of the watermarked image and the number of watermark bits embedded is shown in Fig. 2. The values of different parameters of our model used for watermarking and encryption of images in Fig. 3 are given in Table 1. The PSNR of images at each stage of Fig. 3 is given in Table 2.

The watermark containing essential information like the person's name and ID was generated as shown in Table 3. The watermark bits read from the encrypted image are shown in Table 4, and are identical to the embedded watermark bits, thereby proving that encryption is transparent to watermarking. As can be seen from Fig. 3(c), the encrypted image does not leak out any significant perceptual information of the original image and hence is secure.

VII. CONCLUSIONS AND FUTURE WORK

In this paper, we have proposed algorithms for watermarking and encryption which ensure that the watermark can be read directly from the encrypted stream. We have also suggested that these schemes can be used to securely store and transmit biometric and medical images identified and processed solely on the basis of the watermark which acts as the 'tag'.

In future, we intend to modify our encryption and watermarking algorithms so that even semi-fragile watermarks can be read from the encrypted stream. Strengthening the watermarking process will open up a host of other potential applications where the possible forms of attack also include geometric attacks, recompression and the like. We also hope to tackle the problem of making encryption followed by watermarking of a data stream commutative in future.

REFERENCES

[1] F. Hartung and M. Kutter, "Multimedia watermarking techniques," *Proceedings of the IEEE*, vol. 87, pp. 1079-1107, 1999.

[2] D. Anand and U. Niranjan, "Watermarking Medical Images with Patient Information," in *Proc. of IEEE/EMBS Conference*, pp. 703-7-6, 1998.

[3] S. Lian, Z. Liu, R. Zhen, and H. Wang, "Commutative Encryption and Watermarking in Video Compression," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, no. 6, June 2007.

[4] L.S. Hill, "Cryptography in an Algebraic Alphabet," American Mathematical Monthly, Vol.36, No.6, pp.306-312, 1929.
[5] M. Toorani and A.Falahati, "A secure variant of the Hill Cipher," in *Proc. of the 14th IEEE Symposium on Computers and Communications (ISCC '09)*, pp. 313-316, July 2009.

Aniketh Talwai graduated with a Bachelor of Technology degree in Electronics and Communication Engineering from the Indian Institute of Technology, Guwahati in 2011.
He is currently studying at the Indian Institute of Management, Ahmedabad, and has previously interned at the University of Toronto and the Indian Institute of Science, Bangalore.
He was awarded the President of India Gold Medal for graduating at the top of his class at IIT Guwahati.

Debabrata Sengupta received his Bachelor of Technology degree in Electronics and Communication Engineering from the Indian Institute of Technology Guwahati, in May 2011.He is currently pursuing a Master of Science degree in the department of Electrical Engineering, Stanford University, California, USA.
He received the DAAD fellowship for a summer research internship in Germany in 2010 and worked with the Multimedia Computing Group at University of Augsburg, Germany. His present research interests include Computer Vision and Machine Learning.

Kannan Karthik Received Ph. D. degree from University of Toronto in 2006. Currently he is an Assistant Professor in the Department of Electronics and Electrical Engineering, Indian Institute of Technology Guwahati. His research interests are in exploring new problems in the field of Multimedia Security and using signal processing models to characterize its solutions.