A FAST IMAGE-SCRAMBLE METHOD USING PUBLIC-KEY ENCRYPTION ALLOWING BACKWARD COMPATIBILITY WITH JPEG2000

Osamu WATANABE, Akiko NAKAZAKI, and Hitoshi KIYA

Department of Electrical Engineering, Tokyo Metropolitan University,
1-1 Minami-osawa, Hachioji-shi Tokyo, Japan

ABSTRACT

A new method for partial-scrambling of JPEG 2000 images based on public-key encryption is proposed. By using public-key encryption, the proposed method provides an easier way of managing the encryption key compared with the secret-key based method and also provides tamper resistance against attacks. Although public-key encryption is usually very time-consuming, the proposed method achieves fast encryption by controlling the number of bytes to be encrypted. An encrypted JPEG 2000 image generated by the proposed method has backward compatibility with a standard JPEG 2000 image, so that it can be decoded using a standard JPEG 2000 decoder. The proposed method also has scalability as to the degree of scrambling on the basis of JPEG 2000 coding units, i.e., layers, DWT-levels, subbands, or code-blocks.

1. INTRODUCTION

JPEG 2000 [1, 2] is a new international standard for image coding. Images encoded by JPEG 2000 have good image quality and various features for many kinds of applications. Digital images are much easier to duplicate than analogue images, and consequently, protecting copyrights and privacy has become an important issue. How to secure JPEG 2000 images is now under being discussed as JPEG 2000 part-8 (JPSEC) [3] in the standardization of JPEG 2000.

Several methods for making JPEG 2000 image secure have been proposed [4–8]. One of these is partial-scrambling [5–8]. Partial-scrambling allows authorized users to decode the scrambled JPEG 2000 images and view them at the original quality. Non-authorized users are allowed to view degraded-quality of partially scrambled JPEG 2000 images. In this paper, this property of scrambled JPEG 2000 images are referred to as backward compatibility with standard JPEG 2000 images.

The method described in [6, 7] uses a secret-key encryption algorithm as the scrambling method, and it encrypts all of the JPEG 2000 codestream except for the header information. A similar scrambling method which is based on modulo operation has been discussed in [8]. Both methods perform scrambling for compressed JPEG 2000 codestreams, so that the scrambling operation can be separated from the JPEG 2000 encoding operation. The secret-key encryption based scrambling in [6, 7] is not suitable for point-to-multipoint communications, owing to its key-management problem, and the modulo-based scrambling is not for the encryption algorithm; certain professional applications like digital cinema require scrambling based on an encryption algorithm.

The authors would like to thank Mr. Takahiro Fukuhara, Seiji Kimura, and Katsutoshi Ando of Sony Corp. for their useful advise.

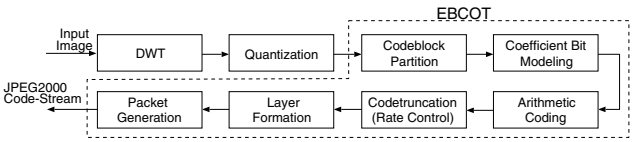


Fig. 1. JPEG 2000 encoder

In this paper, we propose a new partial-scrambling method based on the public-key encryption algorithm. Public-key encryption makes it possible to easily manage encryption keys and prevent tampering with the scrambled images. Public-key encryption requires hundreds or thousands of times as much computational complexity as secret-key encryption [9]; however, the proposed method can reduce the complexity by using a control parameter. Some experiments on JPEG 2000 images will be described, confirming that the scrambling time of the proposed encryption is approximately the same as that of secret-key-based encryption.

2. JPEG 2000 CODING

2.1. Procedure and codestream structure

Figure 1 shows a block diagram of the JPEG 2000 encoder. The coding procedure is briefly summarized as follows. An input image is decomposed into subbands by using a discrete wavelet transform (DWT). The number of DWT is called the DWT-level. Each subband is divided into code-blocks, which are the coding unit of the EBCOT algorithm [10]. EBCOT algorithm encodes each code-block individually and produces a compressed codestream. The compressed codestream may have more than one layer. A layer is defined as the set of compressed data of a certain image quality. The structure of the JPEG 2000 codestream is shown in Fig. 2. The codestream has a main header, packet headers, and so on. In this paper, these headers are refered to as header information, whereas the rest of the codestream is called body-data.

2.2. Marker codes

Marker codes in JPEG 2000 codestreams are special codes with values ranging from $FF90_h$ to $FFFF_h$, where "$_h$" indicates hexadecimal notation. All marker codes are represented with two bytes. The upper byte is FF_h and the lower byte is xx_h. Therefore, the marker codes are represented as $FFxx_h$. To achieve back-

0-7803-8554-3/04/$20.00 ©2004 IEEE.

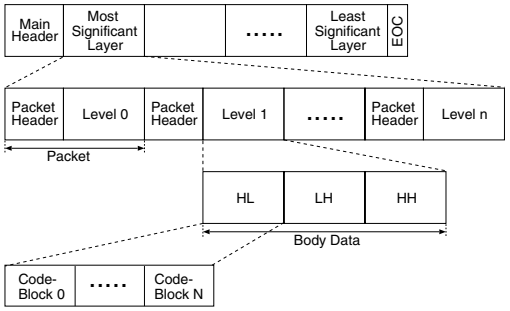


Fig. 2. Structure of JPEG 2000 codestream

Table 1. Marker codes

Value in HEX	Mnemonic	Name
$FF90_h$	SOT	Start of tile
$FF91_h$	SOP	Start of packet
$FF92_h$	EPH	End of packet header
$FF93_h$	SOD	Start of data
$FFD9_h$	EOC	End of codestream

ward compatibility with standard JPEG 2000 codestreams, partial-scrambling methods should avoid generating the marker codes shown in Table 1 because they are used to distinguish body-data from the rest of the codestream. If not, the decoding process of the scrambled codestream may fail.

3. PROPOSED PARTIAL-SCRAMBLING METHOD

The proposed method is intended to realize partial-scrambling without generation of marker codes in the body-data and to do so with low computational complexity even though it uses public-key encryption.

3.1. Conditions to avoid generation of marker codes

Let us assume that the original JPEG 2000 codestream has no marker codes in its body-data.

The conditions to avoid generation of marker codes are:

(a1) If a byte is below 90_h, it must still be below 90_h after scrambling.

(a2) If a byte is 90_h or above (except FF_h), it must still be below FF_h after scrambling.

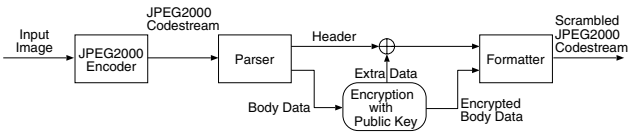
If the byte is FF_h, there is no scrambling restriction.

However, encryption algorithms that map the body-data to random values using one byte cannot meet the above conditions. To meet them, we use a half byte, not a byte, for scrambling. A byte is divided into an upper and a lower half. (a1) and (a2) can be thus rewritten for encryption algorithms as:

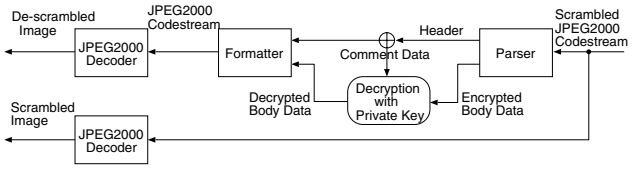
(b1) If a byte is below $F0_h$, there is no restriction on the value of the scrambled lower half byte.

(b2) If a byte is $F0_h$ or above, the lower half byte must be skipped.

Note that (b1) and (b2) can avoid the generation of new FF_h values. From the assumption that there are no marker codes in the original JPEG 2000 body-data, (b1) and (b2) are sufficient to avoid generation of marker codes on account of scrambling.



(a) Scrambling: Encoding and encrypting with public-key



(b) Descrambling: Decrypting with private-key and decoding

Fig. 3. The procedure of the proposed scrambling method

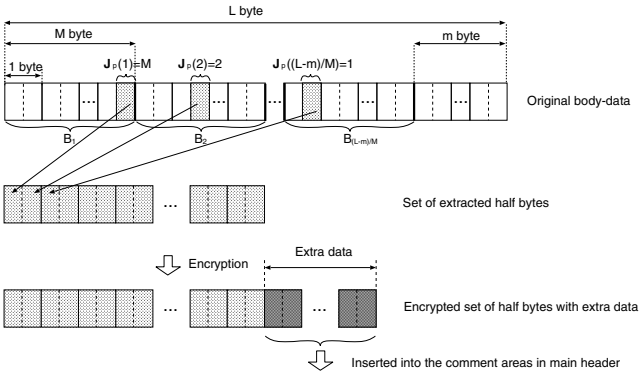


Fig. 4. body-data with L bytes ($M = 3, m = 2$); light-gray parts are selected half bytes and dark-gray parts are extra bytes generated by public-key encryption.

3.2. Procedure

Figure 3 shows the procedure of the proposed method. The proposed method takes a JPEG 2000 codestream as input and produces a scrambled JPEG 2000 codestream. A parser is used to extract the body-data from the codestream and analyze the format of the codestream.

(A) Scrambling

As shown in Fig. 3(a), a JPEG 2000 codestream is sent to the parser and decomposed into the header and body-data. The layers, DWT-levels, subbands, and code-blocks are then selected, and the body-data (presupposed L bytes) are distilled. A parameter M for controlling the computational complexity is introduced and the body-data is divided into blocks B_k ($k = 1, 2, ..., (L-m)/M, m = mod(L, M)$). Each block has M bytes. The following operation is

done to each block B_k, byte by byte (Fig. 4).

Step 1. Using an initial value p and an algorithm for generating random integers $J_p(k)$, generate random integers in the range 1 to M:

$$J_{p_1}(k) \in \{1,2,...,M\}, k = 1,2,.... \quad (1)$$

Step 2. Select the $J_p(k)$-th byte from B_k.

Step 3. Divide the selected $J_p(k)$-th byte into upper and lower half bytes on the basis of:

(3.1) If the selected byte is below $F0_h$, the lower half byte is selected.

(3.2) If the selected byte is $F0_h$ or above, the lower half byte is skipped (block B_k is excluded from scrambling).

Finally, the half-bytes obtained in Step 3 are gathered in the encryption buffer. Then, the public-key encryption is performed with a certain algorithm, such as RSA [11]. The public key of a user is used at this time.

In general, the encrypted data produced by the public-key encryption is longer than that of plain data; i.e., extra data are added by the encryption(Fig. 4). Except for this extra data, all encrypted data is sent back into the original half byte's position. Because the JPEG 2000 codestream has the length information for the body-data, the extra data cannot be returned to the body-data. Therefore, the extra data is inserted into the comment area of the main header. Comment areas are allowed in the main header of the JPEG 2000 codestream and their start positions are indicated by the *COM* marker [1].

(B) Descrambling

The descrambling operation uses an algorithm for generating random integers ($J_p(k)$), the initial value p, an area of L bytes for the body-data (layer, DWT-level, subband, or code-block), and a parameter M. As shown in Fig. 3(b), the encoded data are sent to the parser and decomposed into the header and the body. The descrambling is done as follows.

Step 1. Given p, random integers $J_p(k)$ are generated in the range 1 to M.

Step 2. In each B_k, the $J_p(k)$-th byte selected on the scrambled side is located.

Step 3. A half byte of the $J_p(k)$-th byte is selected as the descrambling target.

(3.1) If the selected byte is below $F0_h$, the lower half byte is selected.

(3.2) If the selected byte is $F0_h$ or above, the byte is skipped (block B_k is excluded from descrambling).

The bytes selected in Step 3 are sent to the decryption buffer. The extra data written in the comment areas of the main header are concatenated with the data in the decryption buffer to obtain the encrypted data. Finally, decryption using the private key of the user is performed on the encrypted data.

3.3. Features

The features of the proposed method are summarized below.

(1) Backward compatibility with JPEG 2000: Since the generation of marker codes is avoided and the extra data generated by using public-key encryption is inserted as comments in the main header, the scrambled JPEG 2000 codestreams can be decoded by a standard JPEG 2000 decoder.

(2) Public-key based encryption: Compared with secret-key encryption, public-key encryption significantly reduces the cost of managing encryption keys in a non-secure environment and the number of necessary keys.

Moreover, since the scrambling key is different from the descrambling key, public-key encryption realizes a kind of tamper resistance. For example, by installing the proposed scrambling function in a digital camera, tamper-resistant photo-communication system can be realized. No one can descramble photographs inside the camera except the owner of the decryption key, e.g., a news medium who buys the photographs from the photographer.

(3) Fast processing using M: Generally, the processing time of the public-key encryption is hundreds or thousands of times longer than in secret-key encryption. M, which is used to make the processing blocks B_k, also has the effect of reducing the scrambling time to approximately $1/M$.

(4) Scalability of scrambling: M also reduces the computational complexity, as this parameter controls the degree of scrambling; A smaller M makes the degree of scrambling greater. For example, if $M = 1$, the whole body-data will be encrypted. Of course, since the JPEG 2000 codestreams are scalable, it is possible to specific layers, DWT-levels, subbands, or code-blocks.

4. EXPERIMENTAL RESULTS

The experiment evaluated the effectiveness of the proposed method. "Lena" (512×512, gray-scale) was used as the test image. This experiment used JPEG 2000 VM8.6 Software [12] as the JPEG 2000 codec. The coding-rate was 0.5 [bits/pixel], and the number of layers was five. Each layer had equal amounts of data (=0.1[bits/pixel]). The RSA [11] based public-key encryption with an 88-bit key was used for scrambling/ descrambling. The platform for this experiment was a Linux PC which had 1.2-GHz Celeron processor and 512-MB RAM.

4.1. Control of scrambling time by using M

Table 2 shows the measured scrambling time for different values of M. B_{length} and E_{length} stand for the length of the target body-data and that of the extra data in bytes. The value of E_{length} depends on the length of the encryption keys. In this experiment, E_{length} is about 10% of B_{length}. The results confirm that the using M reduces the scrambling time to about $1/M$, that is, the scrambling time is almost inversely proportional to the value of M. As compared with the result (A), the result (B) indicates that the scrambling of single layer requires less time than that of all layers.

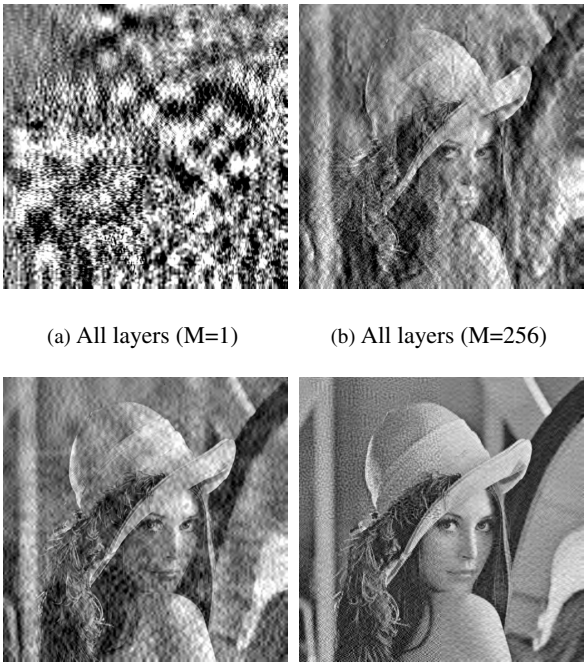
Table 2. Scrambling time vs value of parameter M (Target of scrambling is the whole body-data. B_{length} and E_{length} stand for the length of the target body-data and that of the extra data in bytes, respectively.)

Scrambling Target	M	B_{length}	E_{length}	time[sec]
(A) All layers	1	7,523	760	1.866
	4	3,762	385	0.465
	16	1,888	191	0.119
	64	114	18	0.030
	256	31	13	0.011
(B) Layer 0	1	1,431	153	0.356
	4	370	37	0.090
	16	87	12	0.025
	64	21	12	0.008
	256	5	6	0.002

4.2. Control of scrambling degree

The resulting images of the proposed scrambling are shown in Fig. 5. Figure 5(a) and 5(b) show the results of scrambling all layers at $M = 1$ and $M = 256$, respectively. From these figures, it is verified that the degree of scrambling can be controlled by using M.

In Figs. 5(c) and 5(d), the degree of scrambling is clearly controlled by the selection of the target layer. Similarly, the proposed method could select target DWT-levels, sub-bands, or code-blocks.



(a) All layers (M=1) (b) All layers (M=256)

(c) Layer 0 (M=256) (d) Layer 1 (M=256)

Fig. 5. Results of the proposed scrambling (each image differs as to the scrambling target.)

5. CONCLUSIONS

We described a new method for partial-scrambling of images based on JPEG 2000. The method has backward compatibility with standard JPEG 2000 codestreams. The computational complexity of public-key-encryption based scrambling is reduced to about $1/M$ by using a parameter M. This parameter can also be used to control the degree of scrambling. Moreover, the advantages of public-key encryption, i.e., easy key-management and tamper resistance, can be exploited in the proposed method.

6. REFERENCES

[1] "Information technology — JPEG 2000 image coding system – Part 1: Core coding system," Int. Std. ISO/IEC IS-15444-1, 2000 Dec.

[2] David Taubman and Michael Marcellin, *JPEG2000 Image Compression Fundamentals, Standards and Practice*, Kluwer Academic Publishers, 2002.

[3] "JPSEC working draft - version 3.6," ISO/IEC JTC 1/SC 29/WG 1 N3171, Dec. 2003.

[4] Raphaël Grosbois and Touradj Ebrahimi, "Watermarking in the JPEG2000 domain," in *Proc. of IEEE Workshop on Multimedia Signal Processing(MMSP)*, Oct. 2001.

[5] Raphaël Grosbois and Pierre Gerbelot and Touradj Ebrahimi, "Authentication and access control in the JPEG2000 compressed domain," in *Proc. of SPIE 46th Annual Meeting, Applications of Digital Image Processing XXIV*, July 2001, vol. 4472, pp. 95–104.

[6] Hitoshi Kiya, Shoko Imaizumi, and Osamu Watanabe, "Partial-scrambling of JPEG2000 Images without Generating Marker Codes(in Japanese)," *IEICE Transaction on Fundamentals*, vol. J-86-D-II, no. 11, pp. 1628–1636, Nov. 2003.

[7] Hitoshi Kiya, Shoko Imaizumi, and Osamu Watanabe, "Partial-scrambling of JPEG2000 Images without Generating Marker Codes," in *Proc. of IEEE International Conference on Image Processing(ICIP)*, Sept. 2003.

[8] Frédéric Dufaux, Diego Santa Cruz, and Touradj Ebrahimi, "EPFL's Proposal for JPSEC Core Experiment," ISO/IEC JTC 1/SC29/WG1 N3082, Oct. 2003.

[9] Bruce Schneier, *Applied Cryptography Second Edition : protocols, algorithms, and source code in C*, John Wiley & Sons, Inc., 1996.

[10] David Taubman, "High performance scalable image compression with EBCOT," *IEEE Trans. Image Processing*, vol. 9, no. 7, pp. 1158–1170, July 2000.

[11] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman, "A method for obtaining digital signatures and public key cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978.

[12] "JPEG 2000 verification model 8.6 software," ISO/IEC JTC 1/SC 29/WG 1 N1894, 2000.