Journal of *Imaging*

Article

Integrated Model of Image Protection Techniques

Anu Aryal [1], Shoko Imaizumi [2,*], Takahiko Horiuchi [2] and Hitoshi Kiya [3]

[1] Graduate School of Advanced Integration Science, Chiba University, Chiba 263-8522, Japan; anu@chiba-u.jp
[2] Graduate School of Engineering, Chiba University, Chiba 263-8522, Japan; horiuchi@faculty.chiba-u.jp
[3] Faculty of System Design, Tokyo Metropolitan University, Tokyo 192-0364, Japan; kiya@tmu.ac.jp
[*] Correspondence: imaizumi@chiba-u.jp

Received: 6 September 2017; Accepted: 18 December 2017; Published: 21 December 2017

Abstract: We propose an integrated model of Block-Permutation-Based Encryption (BPBE) and Reversible Data Hiding (RDH). The BPBE scheme involves four processes for encryption, namely block scrambling, block-rotation/inversion, negative-positive transformation and the color component shuffling. A Histogram Shifting (HS) method is adopted for RDH in our model. The proposed scheme can be well suitable for the hierarchical access control system, where the data can be accessed with the different access rights. This scheme encrypts R, G and B components independently. Therefore, we can generate similar output images from different input images. Additionally, the key derivation scheme also provides the security according to the different access rights. Our scheme is also resilient against brute-force attacks and Jigsaw Puzzle Solvers (JPSs). Furthermore, the compression performance is also not severely degraded using a standard lossless compression method.

Keywords: block-permutation-based encryption; reversible data hiding; integrated model; hash chain; key derivation; key space

1. Introduction

Due to the development of digital communication technologies, there are more services such as E-learning, digital diagnosis and web conferences. Therefore, digital content needs to be secured properly, as it can be easily manipulated and have problems with copyright, data security, authentication, etc.

Generally, data hiding can be classified into two categories, namely Irreversible Data Hiding (IDH) [1–3] and Reversible Data Hiding (RDH) [4–9]. In IDH, the host signal cannot be completely recovered. On the other hand, RDH is also referred to as invertible or lossless data hiding, which has been extensively studied to embed secret message bits into a cover object such as an image/video or audio to generate the marked one. In RDH, not only the embedded message needs to be restored precisely, but also the cover image should be losslessly recovered. Therefore, RDH techniques are desirable in some special scenarios such as remote sensing, medical imagery, military communications and law forensics where no permanent change is permitted. Most of the RDH methods aim to provide a good performance on the data hiding rate, the quality of the marked image, the level of security and the computational complexity.

One of the popular methods for RDH is the Histogram Shifting (HS) method. The algorithm proposed by Ni et al. [9] is based on an HS method, in which the data are embedded to the peak of the histogram of an image.

Block-Permutation-Based Encryption (BPBE) schemes [10–13] comprise one of the perceptual encryption techniques. In BPBE schemes, first, an original image is divided into definite block size, and the four processes of encryption, namely block scrambling, block rotation/inversion, negative-positive transformation and color component shuffling, are processed. The main feature of BPBE schemes is that the compression efficiency of the encrypted images is compatible with JPEG compression [11]. Similarly, the BPBE schemes have been proposed for encryption-then-compression (ETC) technique [14–20], in which a user securely transmits images via a Social Network Service (SNS) provider.

Recently, the encryption-based RDH method using adaptive code embedding has been proposed [19]. It is a pixel-based scrambling method. This method has the advantage of a maximum embedding rate of 1.72 bpp and allows the severe distortion of the final encrypted image by embedding more data. However, the disadvantage of this method is that the decryption process is not possible without the extraction of embedded data. Similarly, the access rights cannot be controlled according to various permission levels. In addition, there is no consideration for the compatibility on compression efficiency using international standards. Therefore, we propose an integrated model of BPBE and RDH in this paper. The proposed scheme has the advantage of decryption of the image without extraction of data. Our scheme can also control the quality of the embedded image. Similarly, the access rights can be controlled according to various permission levels. This scheme also considers standard lossless compression methods such as JPEG-LS [21]. The proposed scheme can be attractive in scenarios such as the doctor-nurse relation in a hospital, large organizations and hierarchical file systems, where there is a hierarchical access control according to the various access rights. In some organizations, there is a complex hierarchy between the CEO and front line employees. Different employees have to access different types of information as per their requirements. For example, the CEO is the only user with full permission. A manager may have partial permission to know the salary of the employees, but not personal information such as telephone numbers. Furthermore, we also propose an efficient key derivation scheme to manage the multiple keys, which has been utilized in BPBE and RDH. The experimental results and analysis demonstrate the effectiveness of the proposed scheme.

2. Preliminaries

2.1. BPBE Scheme

In the BPBE scheme [13], an original image with $M \times N$ pixels is divided into different non-overlapping blocks of $B_x \times B_y$ pixels. As illustrated in Figure 1, this scheme has four processes, i.e., block scrambling, block rotation/inversion, negative-positive transformation and color component shuffling, respectively. The procedures for the above-mentioned four processes are elaborated as follows.

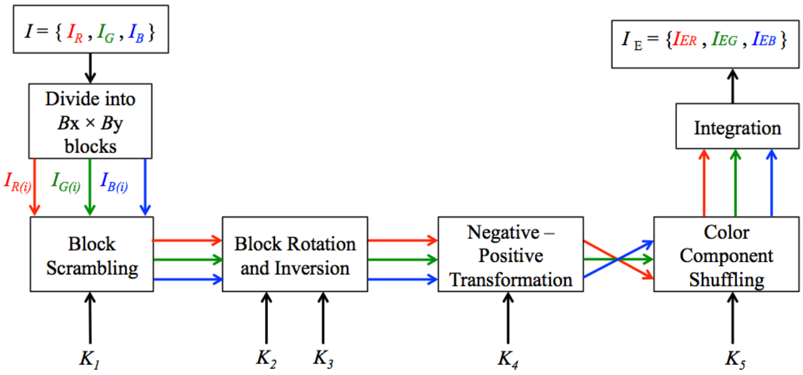
$I = \{I_R, I_G, I_B\}$
Divide into $Bx \times By$ blocks
$I_{R(i)}$
$I_{G(i)}$
$I_{B(i)}$
Block Scrambling
Block Rotation and Inversion
Negative – Positive Transformation
Color Component Shuffling
Integration
$I_E = \{I_{ER}, I_{EG}, I_{EB}\}$
K_1
K_2
K_3
K_4
K_5

Figure 1. Block-Permutation-Based Encryption (BPBE) scheme.

Step 1 Divide each color component of a color image $I = \{I_R, I_G, I_B\}$ into multiple blocks with $B_x \times B_y$ size.
Step 2 Permute the positions of the divided blocks randomly using a key K_1.
Step 3 Rotate and invert each of the divided blocks using keys K_2 and K_3.

Step 4 Apply a negative-positive transformation to the blocks using a key K_4.
Step 5 Shuffle each color components in each block using a key K_5.

In this scheme, the keys K_1, K_2, K_3 and K_4 are commonly used for the Red (R), Green (G) and Blue (B) color components.

2.2. RDH

For RDH, we have applied the conventional RDH algorithm, which is employed in the spatial domain and is based on HS [9], to our work as one of the examples. This algorithm is chosen to maintain the quality of an image. However, the other RDH algorithms can also be employed in our scheme.

3. Proposed Scheme

We propose an integrated model of BPBE and RDH that can be well applied to the hierarchical access control system. The major purposes of using a hierarchical system are that the embedded data can be extracted and also the encrypted image can be decrypted according to the various permission levels. Similarly, the level of security can be controlled by embedding more confidential data at a higher level and less confidential data at a lower level in the hierarchical system. Therefore, the users with a higher permission level are allowed to extract more confidential data than the users with a low permission level. On the other hand, for decryption-only permission, the users are able to decrypt the image, but are not allowed to extract the embedded data. In this manner, the access control can be made flexible for the various users who are accessing the confidential data from different levels in the hierarchy.

Here, we use three independent keys for the R, G and B components [10] for the encryption and embedding process. In the case that we use JPEG-LS [21], which processes the images in the RGB color space without conversion to any other color spaces for the final encrypted images, we can maintain the compression efficiency.

3.1. Encryption and Embedding Process

In this section, we elaborate the encryption and embedding process as shown in Figure 2. For the simulation, we have employed a hundred different test images with 768 × 512 (70) and 512 × 768 pixels (30) from the image database "Content-based image retrieval database" [22]. The divided block size is selected as 16 × 16 pixels for encryption to maintain JPEG compression efficiency [11].

Step 1 Apply RDH to an original image $I = \{I_R, I_G, I_B\}$ of $M \times N$ pixels using keys K_1^R, K_1^G and K_1^B.
Step 2 Divide each color component of an original image into multiple blocks with $B_x \times B_y$ pixels.
Step 3 Permute the positions of the divided blocks randomly using keys K_2^R, K_2^G and K_2^B.
Step 4 Apply RDH using keys K_3^R, K_3^G and K_3^B.
Step 5 Rotate and invert each block randomly using keys K_4^R, K_4^G, K_4^B, K_5^R, K_5^G and K_5^B.
Step 6 Apply RDH using keys K_6^R, K_6^G and K_6^B.
Step 7 Apply the negative-positive transformation for each block using keys K_7^R, K_7^G and K_7^B.
Step 8 Apply RDH using keys K_8^R, K_8^G and K_8^B.
Step 9 Shuffle the three color components, i.e., R, G,and B in each block by using a key K_9.
Step 10 Apply RDH using keys K_{10}^R, K_{10}^G and K_{10}^B.
Step 11 Generate the encrypted image $I_E = \{I_{ER}, I_{EG}, I_{EB}\}$ by integrating all the transformed blocks.

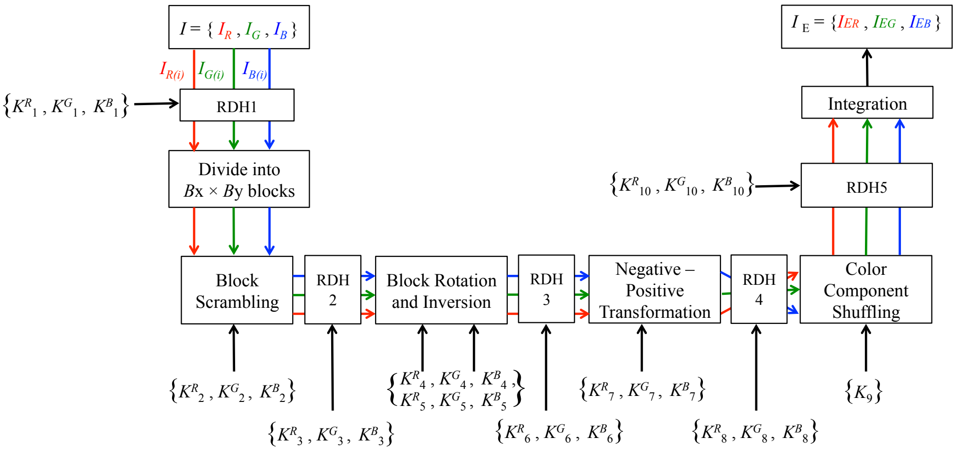
$I = \{I_R, I_G, I_B\}$
$I_{R(i)}$
$I_{G(i)}$
$I_{B(i)}$
$\{K^R_1, K^G_1, K^B_1\}$
RDH1
Divide into Bx × By blocks
Block Scrambling
RDH 2
Block Rotation and Inversion
RDH 3
Negative – Positive Transformation
RDH 4
Color Component Shuffling
RDH5
Integration
$I_E = \{I_{ER}, I_{EG}, I_{EB}\}$
$\{K^R_{10}, K^G_{10}, K^B_{10}\}$
$\{K^R_2, K^G_2, K^B_2\}$
$\{K^R_3, K^G_3, K^B_3\}$
$\{K^R_4, K^G_4, K^B_4, K^R_5, K^G_5, K^B_5\}$
$\{K^R_6, K^G_6, K^B_6\}$
$\{K^R_7, K^G_7, K^B_7\}$
$\{K^R_8, K^G_8, K^B_8\}$
$\{K_9\}$

Figure 2. Encryption and embedding process. RDH, Reversible Data Hiding.

3.2. Key Derivation

A large amount of keys would be generated in the proposed scheme due to the use of independent keys K_1^i, K_2^i, …, K_8^i, K_{10}^i $(i = R, G, B)$ for three components. Therefore, the management of those multiple keys is an important issue. Hence, we also consider deriving an efficient key management scheme with the use of hash chains [23] and decrease the number of managed keys. With the use of hash chains, we assign the derived keys to each step of encryption and embedding. The number of managed keys is diminished to one, that is key K_M. Keys K_x can be given by:

$$K_x = H^x(K_M), \tag{1}$$

where $x = 1, 2, \ldots, 10$ and $H(\cdot)$ is a one-way hash function.

An efficient key derivation scheme is as shown in Figure 3. The keys $K^i_{\mathrm{e}(u)}$ are the representation for the embedding process, whereas the keys $K^i_{\mathrm{c}(u)}$ are the representation for the encryption process. Here, $u(u = 1, 2, \ldots, 5)$ indicates the number of the encryption or the embedding process. In the embedding process, a key $K^R_{\mathrm{e}(1)}$ can be derived by performing a one-way hash chain to the result obtained by XOR (exclusive or) operation between key K_M and its associated random numbers a^R_{e}. Similarly, $K^G_{\mathrm{e}(1)}$ can be achieved by a one-way hash function to the result of XOR operation between $K^R_{\mathrm{e}(1)}$ and a^G_{e}. A key $K^B_{\mathrm{e}(1)}$ can be derived by performing a one-way hash function to the result obtained by XOR operation between $K^G_{\mathrm{e}(1)}$ and a^B_{e}. The key derivation can be given as follows.

$$K^R_{\mathrm{e}(1)} = H(K_M \oplus a^R_{\mathrm{e}}), \tag{2}$$

$$K^G_{\mathrm{e}(1)} = H(K^R_{\mathrm{e}(1)} \oplus a^G_{\mathrm{e}}), \tag{3}$$

$$K^B_{\mathrm{e}(1)} = H(K^G_{\mathrm{e}(1)} \oplus a^B_{\mathrm{e}}), \tag{4}$$

where $\oplus$ represents a bitwise XOR operation.

In addition, by using hash chains, all other keys $K^i_{\mathrm{e}(1)}$, $K^i_{\mathrm{e}(2)}$, …, $K^i_{\mathrm{e}(5)}$, $(i = R, G, B)$ can be obtained as follows.

$$K^R_{\mathrm{e}(u)} = H^{u-1}(K^R_{\mathrm{e}(1)}), \tag{5}$$

$$K^G_{\mathrm{e}(u)} = H^{u-1}(K^G_{\mathrm{e}(1)}), \tag{6}$$

$$K^B_{\mathrm{e}(u)} = H^{u-1}(K^B_{\mathrm{e}(1)}), \tag{7}$$

where $u = 2, 3, 4, 5$.

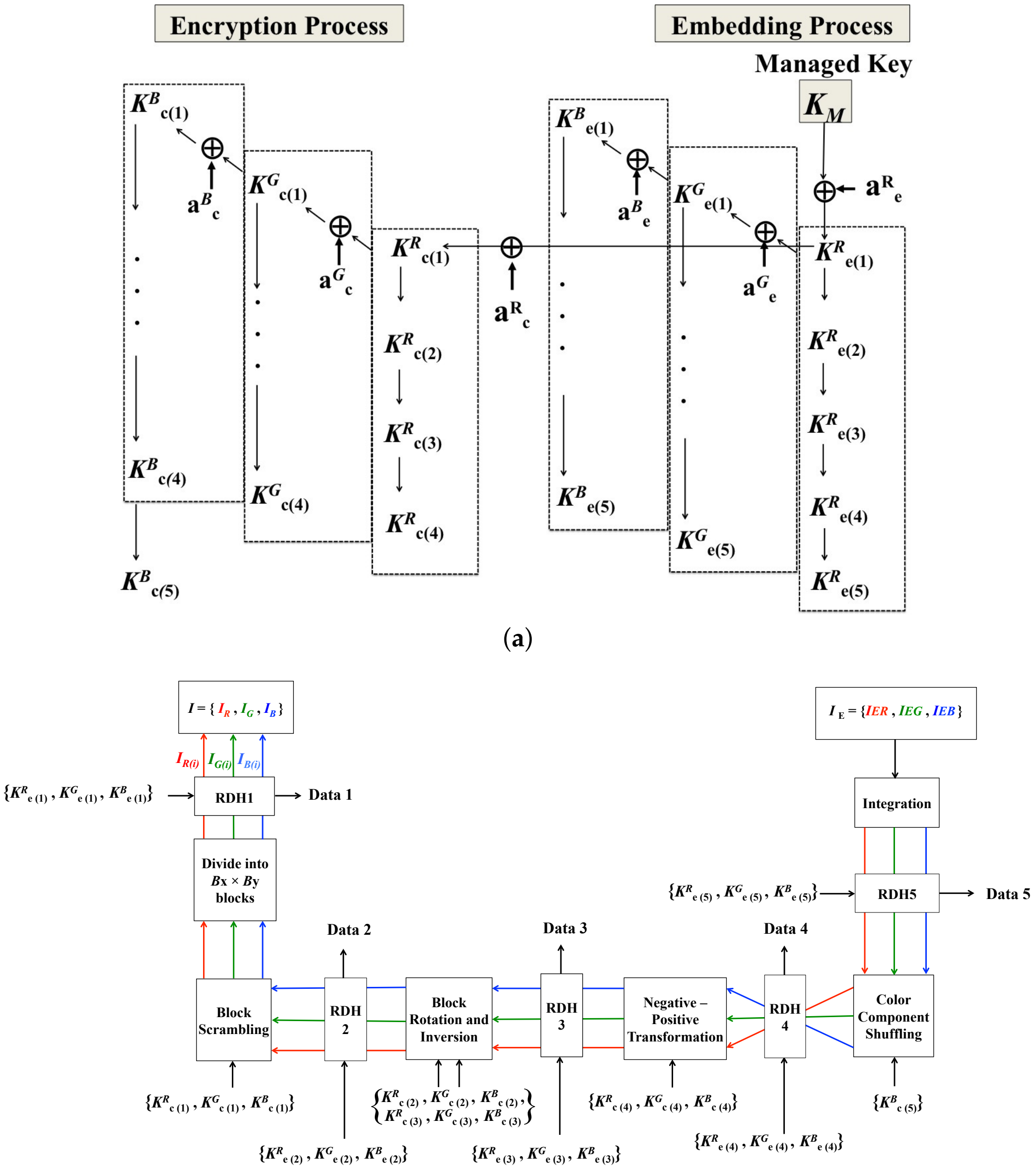
Encryption Process
Embedding Process
Managed Key
K_M
$K^B{}_{c(1)}$
$a^B{}_c$
$K^B{}_{c(4)}$
$K^B{}_{c(5)}$
$K^G{}_{c(1)}$
$a^G{}_c$
$K^G{}_{c(4)}$
$K^R{}_{c(1)}$
$K^R{}_{c(2)}$
$K^R{}_{c(3)}$
$K^R{}_{c(4)}$
$a^R{}_c$
$K^B{}_{e(1)}$
$a^B{}_e$
$K^B{}_{e(5)}$
$K^G{}_{e(1)}$
$a^G{}_e$
$K^G{}_{e(5)}$
$a^R{}_e$
$K^R{}_{e(1)}$
$K^R{}_{e(2)}$
$K^R{}_{e(3)}$
$K^R{}_{e(4)}$
$K^R{}_{e(5)}$
(a)
$I = \{I_R, I_G, I_B\}$
$I_{R(i)}$ $I_{G(i)}$ $I_{B(i)}$
$\{K^R{}_{e(1)}, K^G{}_{e(1)}, K^B{}_{e(1)}\}$
RDH1
Data 1
Divide into $Bx \times By$ blocks
Block Scrambling
$\{K^R{}_{c(1)}, K^G{}_{c(1)}, K^B{}_{c(1)}\}$
RDH 2
Data 2
$\{K^R{}_{e(2)}, K^G{}_{e(2)}, K^B{}_{e(2)}\}$
Block Rotation and Inversion
$\{K^R{}_{c(2)}, K^G{}_{c(2)}, K^B{}_{c(2)},$ $K^R{}_{c(3)}, K^G{}_{c(3)}, K^B{}_{c(3)}\}$
RDH 3
Data 3
$\{K^R{}_{e(3)}, K^G{}_{e(3)}, K^B{}_{e(3)}\}$
Negative – Positive Transformation
$\{K^R{}_{c(4)}, K^G{}_{c(4)}, K^B{}_{c(4)}\}$
RDH 4
Data 4
$\{K^R{}_{e(4)}, K^G{}_{e(4)}, K^B{}_{e(4)}\}$
Color Component Shuffling
$\{K^B{}_{c(5)}\}$
$\{K^R{}_{e(5)}, K^G{}_{e(5)}, K^B{}_{e(5)}\}$
RDH5
Data 5
Integration
$I_E = \{IER, IEG, IEB\}$
(b)

Figure 3. Key derivation. (**a**) Key derivation scheme; (**b**) decryption extraction process.

In encryption process, $K^R_{\mathrm{c}(1)}$ can be obtained by the result of the XOR operation between $K^R_{\mathrm{e}(1)}$ and a^R_{c}. The key derivation process is described as follows.

$$K^R_{\mathrm{c}(1)} = H(K^R_{\mathrm{e}(1)} \oplus a^R_{\mathrm{c}}), \tag{8}$$

$$K^G_{\mathrm{c}(1)} = H(K^R_{\mathrm{c}(1)} \oplus a^G_{\mathrm{c}}), \tag{9}$$

$$K^B_{\mathrm{c}(1)} = H(K^G_{\mathrm{c}(1)} \oplus a^B_{\mathrm{c}}), \tag{10}$$

Similarly,

$$K^R_{\mathrm{c}(u)} = H^{u-1}(K^R_{\mathrm{c}(1)}), \tag{11}$$

$$K^G_{\mathrm{c}(u)} = H^{u-1}(K^G_{\mathrm{c}(1)}), \tag{12}$$

$$K^B_{c(u)} = H^{u-1}(K^B_{c(1)}), \tag{13}$$

where $u = 2, 3, 4$. Regarding the key for the color component shuffling, it is single for each image. Therefore, it is derived by:

$$K^B_{c(5)} = H(K^B_{c(4)}). \tag{14}$$

3.3. Decryption and Extraction Process

As shown in Figure 3, there are different hierarchical levels. Therefore, the access rights for each hierarchical level can be easily controlled for the decryption and extraction. The user with high permission is able to extract and decrypt more confidential data and images, respectively, than the user with low permission.

Table 1 shows the total embedding capacity of Japan Image32 at each level for single embedding (one time) and double embedding (two times). Tables 2 and 3 illustrate the total embedding capacity and the Peak Signal-to-Noise Ratio (PSNR) values of test images for single embedding and double embedding, respectively. Figures 4 and 5 show the simulation results of Japan Image32 and Japan Image22 obtained by different permissions in the case of single embedding. The following sections describe the decryption and extraction process for different permissions.

Table 1. Embedding capacity at each level of Japan Image32.

Level	**Embedding Capacity (bits)**	
	Single Embedding	**Double Embedding**
Data 1	18,709	37,233
Data 2	18,524	36,866
Data 3	18,476	36,090
Data 4	16,827	31,934
Data 5	15,297	27,396
Total	87,833	169,519

Table 2. Total embedding capacity (bits) and Peak Signal-to-Noise Ratio (PSNR) values for single embedding.

Image	**Single Embedding**	
(768 × 512 pixels)	**Total Embedding Capacity (bits)**	**PSNR**
Japan Image22	41,576	38.34
Japan Image27	78,682	43.93 (max)
Japan Image32	87,833	40.41
Australia Image01	75,360	43.80
Australia Image03	86,720	41.69
Australia Image05	41,965	38.75
Indonesia Image01	285,821	37.91
Indonesia Image35	240,071	37.42 (min)
Iran Image13	525,427 (max)	38.29
Iran Image49	132,792	38.41
(512 × 768 pixels)	**Total Embedding Capacity (bits)**	**PSNR**
Japan Image26	38,418 (min)	38.81
Japan Image31	55,594	39.61
Australia Image02	41,212	42.77
Australia Image06	65,859	40.55
Indonesia Image20	89,424	40.98
Indonesia Image26	112,831	39.69
Iran Image10	218,532	42.29
Iran Image15	62,670	41.94

Table 3. Total embedding capacity (bits) and PSNR values for double embedding (Japan).

Image (768 × 512 pixels)	**Double Embedding**	
	Total Embedding Capacity (bits)	**PSNR**
Japan Image01	131,529	37.44
Japan Image13	85,156	34.64
Japan Image32	169,519	34.65
Japan Image17	165,832	35.55
Japan Image15	88,180	32.93
Japan Image22	79,477	32.43
Japan Image27	142,136	39.68
Japan Image08	90,016	34.53
Japan Image20	101,275	35.66
Japan Image02	111,144	37.65

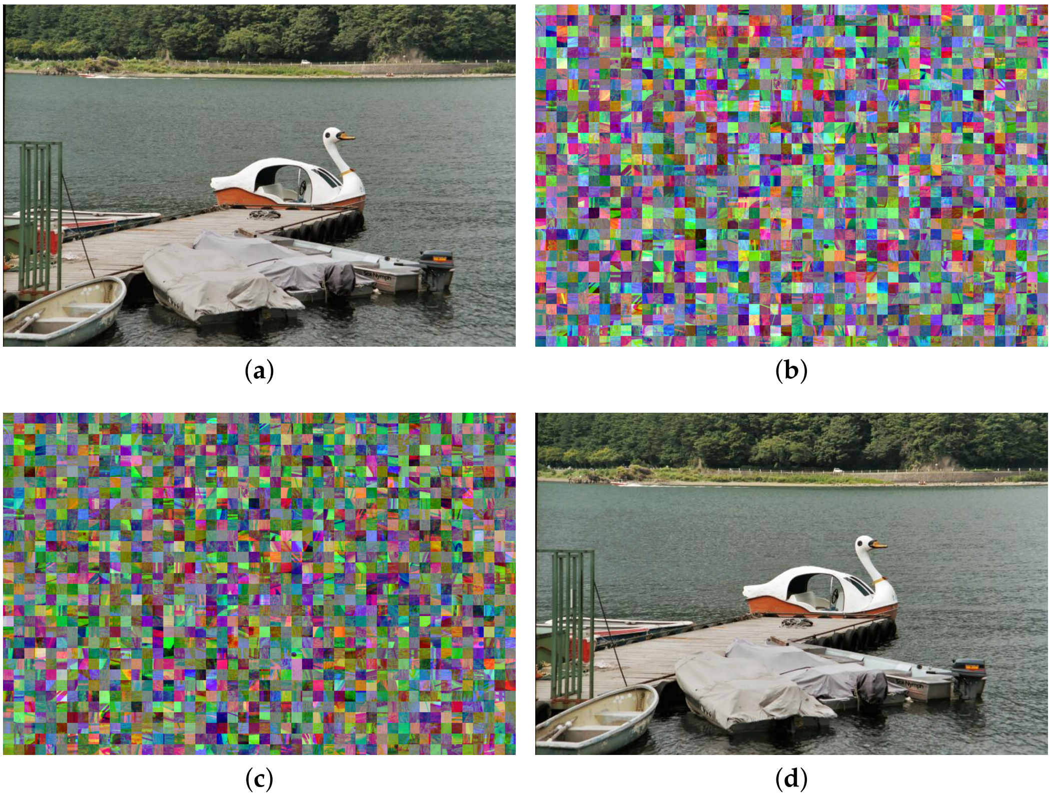
(**a**)
(**b**)
(**c**)
(**d**)

Figure 4. Simulation results of Japan Image32 obtained by different permissions (single embedding). (**a**) Original image (Japan Image32); (**b**) final encrypted image; (**c**) half encrypted image (decryption: negative-positive transformation and color component shuffling; extraction: Data 3, 4, and 5); (**d**) decryption-only image.

3.3.1. Full Permission

Let us consider that a user has the full access right to extract all the embedded data and to entirely decrypt the images. Therefore, the user has the full permission to get a managed key K_M as given in Figure 3a. If a user were to obtain a key K_M, the user would be able to derive all twenty eight keys, retrieve the original image from the final encrypted image as shown in Figures 4b and 5b and also extract all the embedded data, as shown in Table 1.

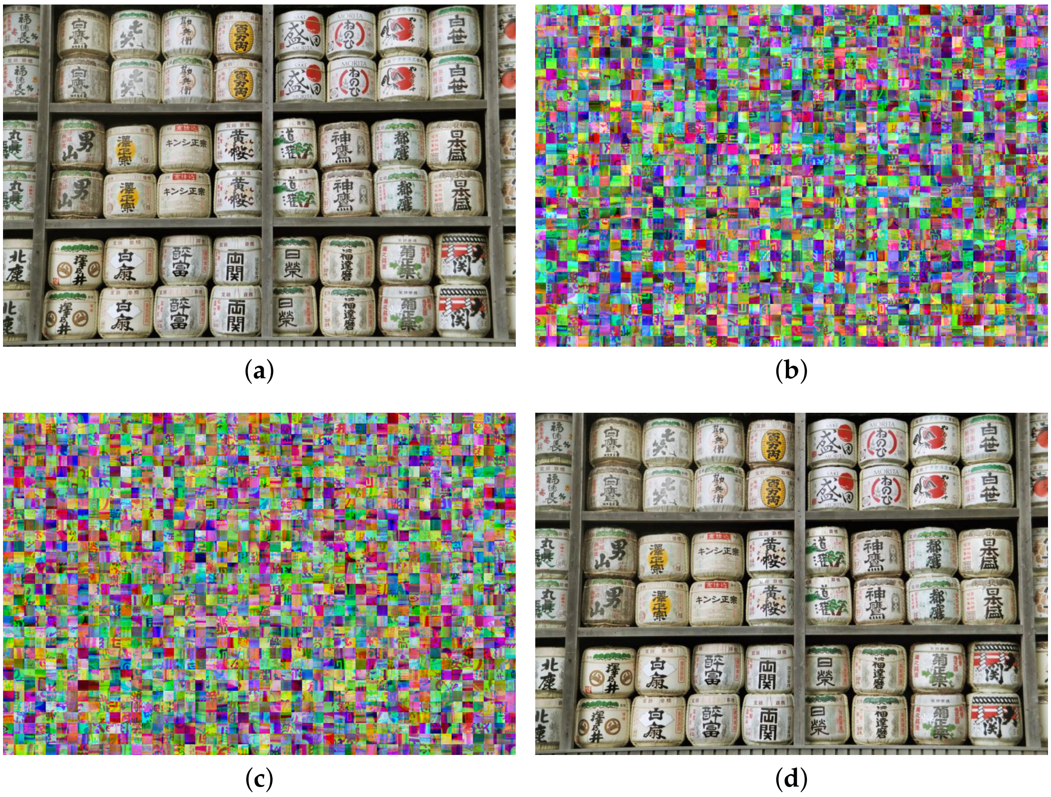
(**a**) (**b**) (**c**) (**d**)

Figure 5. Simulation results of Japan Image22 obtained by different permissions (single embedding). (**a**) Original image (Japan Image22); (**b**) final encrypted image; (**c**) half encrypted image (decryption: negative-positive transformation and color component shuffling; extraction: Data 3, 4, and 5); (**d**) decryption-only image.

3.3.2. Partial Permission

In this case, let us assume that another user is only permitted to extract Data 3, 4 and 5, as shown in Figure 3b. The user would obtain six keys, i.e., $K^R_{\mathrm{e}(3)}$, $K^G_{\mathrm{e}(3)}$, $K^B_{\mathrm{e}(3)}$, $K^R_{\mathrm{c}(4)}$, $K^G_{\mathrm{c}(4)}$ and $K^B_{\mathrm{c}(4)}$. Hence, the user is able to derive the seven keys, i.e., $K^R_{\mathrm{e}(4)}$, $K^R_{\mathrm{e}(5)}$, $K^G_{\mathrm{e}(4)}$, $K^G_{\mathrm{e}(5)}$, $K^B_{\mathrm{e}(4)}$, $K^B_{\mathrm{e}(5)}$, $K^B_{\mathrm{c}(5)}$, extract Data 3, 4, and 5 and obtain the half encrypted image as given in Figures 4c and 5c.

3.3.3. Decryption-Only Permission

Let us assume that a user is only allowed to decrypt the image, but is restricted in extracting the embedded data. If a user were to obtain a key, i.e., $K^R_{\mathrm{c}(1)}$, then he/she would be able to derive the twelve keys, i.e., $K^G_{\mathrm{c}(1)}$, $K^B_{\mathrm{c}(1)}$, $K^R_{\mathrm{c}(2)}$, $K^G_{\mathrm{c}(2)}$, $K^B_{\mathrm{c}(2)}$, $K^R_{\mathrm{c}(3)}$, $K^G_{\mathrm{c}(3)}$, $K^B_{\mathrm{c}(3)}$, $K^R_{\mathrm{c}(4)}$, $K^G_{\mathrm{c}(4)}$, $K^B_{\mathrm{c}(4)}$ and $K^B_{\mathrm{c}(5)}$. Figures 4d and 5d show the simulation results for decryption-only permission. From the experimental results of 100 images, the maximum and minimum values of PSNR are 43.93 (Japan Image27) and 37.42 (Indonesia Image35), respectively, as shown in Table 2. Hence, there is approximately only 6.51 dB of variation in PSNR values. The maximum embedding capacity for ‘Iran Image13’ is 525,427 bits with its corresponding PSNR value as 38.29 dB. Similarly, the minimum embedding capacity for ‘Japan Image26’ is 38,418 bits with its corresponding PSNR value as 38.81 dB. In this case, our scheme is effective for ‘Iran Image13’, as it has comparatively higher embedding capacity than ‘Japan Image26’.

4. Experimental Results and Analysis

4.1. Key Space

Generally, there are various types of attacks on encryption such as differential attacks, brute-force attacks, statistical attacks, and so on. A brute-force attack is a kind of a trial-and-error method that is used to obtain the possible combination. Here, we evaluate the size of key space assuming the brute-force attacks. The conventional BPBE scheme [13] has four encryption processes, namely block scrambling, block rotation/inversion, negative-positive transformation and color component shuffling, respectively. It performs the encryption with the identical keys for all color components. The key space can be determined by the number of the divided blocks n. The four encryption processes are independent of each other. Therefore, the total key space is calculated by multiplying the key spaces of each encryption process as described below.

If an original image with $M \times N$ pixels is divided into n blocks with $B_x \times B_y$ pixels, n is calculated by:

$$n = \frac{M \times N}{B_x \times B_y}. \tag{15}$$

In block scrambling, the key space N_B is the number of permutations of n blocks, which is given by:

$$N_B = n!. \tag{16}$$

Similarly, while combining each of the four processes of rotation and inversion, there are some similar patterns generated. Therefore, the maximum possible patterns generated due to the rotation and the inversions are eight, respectively. The key space of the block rotation and inversion N_R is given as:

$$N_R = 8^n. \tag{17}$$

The number of patterns for the negative-positive transformation N_N and color component shuffling N_C is two and six, respectively. Hence, the total key spaces can be calculated by:

$$N_N = 2^n, \tag{18}$$

$$N_C = 6^n. \tag{19}$$

The total key space of the encrypted images N_A can be calculated by:

$$\begin{aligned} N_A &= N_B \times N_R \times N_N \times N_C \\ &= n! \times 8^n \times 2^n \times 6^n \\ &= n! \times 2^{5n} \times 3^n. \end{aligned} \tag{20}$$

On the other hand, the proposed scheme performs the encryption with the independent keys for all color components. Hence, the key spaces $N_B^{'}$, $N_R^{'}$ and $N_N^{'}$ are given by:

$$N_B^{'} = (n)!^3. \tag{21}$$

$$N_R^{'} = (8^n)^3. \tag{22}$$

$$N_N^{'} = (2^n)^3, \tag{23}$$

Therefore, the total key space of the encrypted image $N_A^{'}$ can be calculated by:

$$\begin{aligned} N_A^{'} &= N_B^{'} \times N_R^{'} \times N_N^{'} \times N_C \\ &= (n)!^3 \times (8^n)^3 \times (2^n)^3 \times 6^n \\ &= (n)!^3 \times 2^{13\,n} \times 3^n. \end{aligned} \tag{24}$$

From the above-mentioned analysis, the proposed scheme has a larger key space than the conventional scheme [13] due to the use of independent keys for R, G and B components in the encryption. Although the conventional scheme [10] utilizes the independent keys for encryption, the key space of the proposed scheme is more complex because of the embedding process. Therefore, the proposed scheme is more secured by confirming its large key space. Hence, the resilience against brute-force attacks can be improved.

4.2. Resilience Against Jigsaw Puzzle Solvers

JPS is a kind of attack that uses the correlation between the large numbers of pieces to obtain an original image. In an encrypted image, each block has a strong correlation to that of an original image. Hence, it is required to analyze the security of the proposed scheme with JPSs. According to [24,25], direct comparison D_c is the ratio of number of the pieces that are in correct position. Neighbor comparison N_c represents the ratio of the number of correctly joined blocks. Similarly, the largest component that is denoted as L_c is the ratio of the number of largest joined blocks that have correct adjacencies. As shown in Table 4, we have calculated the average scores of D_c, N_c and L_c of seven different standard images, i.e., Lena, Mandrill, Milkdrop, Pepper, Girl, Lake and Airplane of 512 × 512 pixels from the Signal and Image Processing Institute (SIPI) database [26]. The original images of 512 × 512 pixels are trimmed to 512 × 480 pixels to make a rectangular shape in the JPSs analysis. It is noted that a block size of 32 × 32 is chosen for the encryption. The scores of D_c, N_c and L_c are '1′s if the image is completely assembled by JPSs, whereas these scores are '0′s when the puzzles are not assembled at all. Hence, it is confirmed that the use of independent keys for encryption makes puzzle solvers almost impossible even when the number of blocks is 240.

4.3. Compression Efficiency

The compression efficiency is evaluated by calculating the bitrates, which is given by:

$$Bitrate\ (bpp) = \frac{Size\ of\ image\ file}{No.\ of\ pixels\ in\ original\ image}. \tag{25}$$

JPEG is a lossy compression algorithm. Therefore, if we use lossy compression in our scheme, the embedded data will be broken, and we cannot extract the embedded data. However, the lossless compression methods such as JPEG-LS can be applied for our scheme. As shown in Table 5, the compression performance of the proposed scheme is not severely degraded as compared to the original image. Hence, our scheme is somehow compatible with the JPEG-LS compression method.

Table 4. Evaluation of Jigsaw Puzzle Solvers (JPSs) using standard images of 512 × 512 pixels.

Component	**BPBE Scheme (32 × 32)**	
	Identical	**Independent**
D_c(Avg)	0.102	0.002
N_c(Avg)	0.152	0.006
L_c(Avg)	0.197	0.008

Table 5. Calculation of bitrate after JPEG-LS compression (Iran Image13).

Image	Bitrate (bpp)
Original	13.10
Proposed scheme (independent)	13.73
Proposed scheme (identical)	13.70

5. Conclusions

In this paper, we proposed an integrated model of image protection techniques. The proposed scheme allows the hierarchical process for the encryption and the data embedding. Therefore, our scheme is suitable for the hierarchical access control system, where the permission is granted according to the various access rights. The compression performance is also not severely degraded as compared to the original image. In addition, we also posit an efficient key derivation scheme for managing the large amount of keys generated in the encryption and data embedding process. The size of the key space in the proposed scheme is larger than the conventional schemes due to the use of independent keys and the embedding process. Hence, our scheme is resilient against brute-force attacks. Similarly, we also assure that the proposed scheme is almost impossible for JPSs attacks. Our future work involves embedding the data that are related to the original image. For example, it may be possible to control the visibility of Regions of Interest (ROIs) via embedded data in a hierarchical manner.

Acknowledgments: This work was partially supported by Grant-in-Aid for Scientific Research(B), No.17H03267, from the Japan Society for the Promotion Science.

Author Contributions: Shoko Imaizumi and Takahiko Horiuchi designed the experiments; Anu Aryal performed the experiments and analyzed the results with Shoko Imaizumi; Hitoshi Kiya contributed the theoretical concept; Anu Aryal, Shoko Imaizumi, and Takahiko Horiuchi wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cox, I.J.; Kilian, J.; Leighton, F.T.; Shamoon, T. Secure spread spectrum watermarking for multimedia. *IEEE Trans. Image Process.* **1997**, *6*, 1673–1687.
2. Tirkel, A.Z.; Osborne, C.F.; Schyndel, R.G. Image watermarking-a spread spectrum application. In Proceedings of the IEEE 4th International Symposium on Spread Spectrum Techniques and Applications, Mainz, Germany, 25 September 1996; Volume 2, pp. 785–789.
3. Yeung, M.M.; Mintzer, F.C. Invisible watermarking for image verification. *Electron. Imaging* **1998**, *7*, 578–591.
4. Golijan, M.; Fridrich, J.J.; Du, R. Distortion-free data embedding. In Proceedings of the International Workshop on Information Hiding, Pittsburgh, PA, USA, 25–27 April 2001; pp. 27–41.
5. Xuan, G.; Zhu, J.; Chen, J.; Shi, Y.Q.; Ni, Z.; Su, W. Distortionless data hiding based on integer wavelet transform. *IEEE Electron. Lett.* **2002**, *38*, 1646–1684.
6. Tian, J. Reversible data embedding using a difference expansion. *IEEE Trans. Circuits Syst. Video Technol.* **2003**, *13*, 890–896.
7. Thodi, D.M.; Rodriguez, J.J. Expansion embedding techniques for reversible watermarking. *IEEE Trans. Image Process.* **2007**, *16*, 721–730.
8. Wu, H.T.; Dugelay, J.L.; Shi, Y.Q. Reversible image data hiding with contrast enhancement. *IEEE Signal Process. Lett.* **2015**, *22*, 81–85.
9. Ni, Z.; Shi, Y.Q.; Ansari, N.; Su, W. Reversible data hiding. *IEEE Trans. Circuits Syst. Video Technol.* **2006**, *16*, 354–362.
10. Imaizumi, S.; Ogasawara, T.; Kiya, H. Block-Permutation-Based-Encryption Scheme with Enhanced Color Scrambling. In Proceedings of the 20th Scandinavian Conference on Image Analysis, Tromsø, Norway, 12–14 June 2017; Volume 10269, pp. 562–573.

11. Kurihara, K.; Imaizumi, S.; Shiota, S.; Kiya, H. An Encryption-then-Compression System for Lossless Image Compression Standards. *IEICE Trans. Inf. Syst.* **2017**, *E100-D*, 52–56.
12. Watanabe, O.; Uchida, A.; Fukuhara, T.; Kiya, H. An Encryption-then-Compression System for JPEG 2000 Standard. In Proceedings of the 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Brisbane, Australia, 19–24 April 2015.
13. Kurihara, K.; Kikuchi, M.; Imaizumi, S.; Shiota, S.; Kiya, H. An Encryption-then-Compression System for JPEG/Motion JPEG Standard. *IEICE Trans. Fundam.* **2015**, *E98-A*, 2238–2245.
14. Zhou, J.; Liu, X.; Au, O.C.; Tang, Y.Y. Designing an efficient image encryption-then-compression system via prediction error clustering and random permutation. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 39–50.
15. Erkin, Z.; Piva, A.; Katzenbeisser, S.; Lagendijk, R.L.; Shokrollahi, J.; Neven, G.; Barni, M. Protection and retrieval of encrypted multimedia content: When cryptography meets signal processing. *EURASIP J. Inf. Secur.* **2007**, *2007*, doi:10.1155/2007/78943.
16. Liu, W.; Zeng, W.; Dong, L.; Yao, Q. Efficient compression of encrypted grayscale images. *IEEE Trans. Image Process.* **2010**, *19*, 1097–1102.
17. Johnson, M.; Ishwar, P.; Prabhakaran, V.; Schnoberg, D.; Ramchandran, K. On compressing encrypted data. *IEEE Trans. Signal Process.* **2004**, *52*, 2992–3006.
18. Hu, R.; Li, X.; Yang, B. A new lossy compression scheme for encrypted gray-scale images. In Proceedings of the 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Florence, Italy, 4–9 May 2014; pp. 7387–7390.
19. Yi, S.; Zhou, Y. Adaptive code embedding for reversible data hiding in encrypted images. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017; pp. 4322–4326.
20. Chuman, T.; Kurihara, K.; Kiya, H. Security evaluation for block scrambling-based ETC systems against extended jigsaw puzzle solver attacks. In Proceedings of the 2017 IEEE International Conference on Multimedia and Expo (ICME), Hong Kong, China, 10–14 July 2017.
21. International Organization for Standardization. *ISO/ICE 14495-1:1999 Information Technology—Lossless and Nearlossless Compression of Continuous-Tone Still Images Baseline*; International Organization for Standardization: Geneva, Switzerland, 1999.
22. Content-Based Image Retrieval Database. Available online: http://imagedatabase.cs.washington.edu/groundtruth (accessed on 6 September 2017).
23. Imaizumi, S.; Fujiyoshi, M.; Kiya, H. An efficient access control method for composite multimedia content. *IEICE Electron. Express* **2010**, *7*, 1534–1538.
24. Gallagher, A. Jigsaw puzzles with pieces of unknown orientation. In Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Providence, RI, USA, 16–21 June 2012; pp. 382–389.
25. Cho, T.; Avidan, S.; Freeman, W. A probabilistic image jigsaw puzzle solvers. In Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), San Francisco, CA, USA, 13–18 June 2010; pp. 183–190.
26. SIPI Database. Available online: http://sipi.usc.edu/database (accessed on 6 September 2017).