

# Northumbria Research Link

Citation: Khelifi, Fouad, Brahim, Tahar, Han, Jungong and Li, Xuelong (2018) Secure and privacy-preserving data sharing in the cloud based on lossless image coding. Signal Processing, 148. pp. 91-101. ISSN 0165-1684

Published by: Elsevier

URL: <https://doi.org/10.1016/j.sigpro.2018.02.016>  
<<https://doi.org/10.1016/j.sigpro.2018.02.016>>

This version was downloaded from Northumbria Research Link:  
<http://nrl.northumbria.ac.uk/id/eprint/33492/>

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: <http://nrl.northumbria.ac.uk/policies.html>

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)

# Secure and Privacy-preserving Data Sharing in the Cloud based on Lossless Image Coding

Fouad Khelifi<sup>1</sup>, Tahar Brahimi<sup>2</sup>, Jungong Han<sup>1</sup>, Xuelong Li<sup>3</sup>

---

## Abstract

Image and video processing in the encrypted domain has recently emerged as a promising research area to tackle privacy-related data processing issues. In particular, reversible data hiding in the encrypted domain has been suggested as a solution to store and manage digital images securely in the cloud while preserving their confidentiality. However, although efficiency has been claimed with reversible data hiding techniques in encrypted images (RDHEI), reported results show that the cloud service provider cannot add more than 1 bit per pixel (bpp) of additional data to manage stored images. This paper highlights the weakness of RDHEI as a suggested approach for secure and privacy-preserving cloud computing. In particular, we propose a new, simple, and efficient approach that offers the same level of data security and confidentiality in the cloud without the process of reversible data hiding. The proposed idea is to compress the image via a lossless image coder in order to create space before encryption. This space is then filled with a randomly generated sequence and combined with an encrypted version of the compressed bit stream to form a full resolution encrypted image in the pixel domain. The cloud service provider uses the created room in the encrypted image to add additional data and produces an encrypted image containing additional data in a similar fashion. Assessed with the lossless Embedded Block Coding with Optimized Truncation (EBCOT) algorithm on natural images, the proposed scheme has been shown to exceed the capacity of 3 bpp of additional data while maintaining data security and confidentiality.

*Keywords:* Reversible data hiding, encryption, confidentiality, security, capacity, privacy-preserving cloud.

---

## 1. Introduction

Over the last decade, there has been a growing body of research on data analytics and processing technologies that deal with large amounts of data of different types and acquired from various sources [1, 2, 3]. However, privacy concerns have also been widely raised in the literature in signal processing applications that deal with user-related data. In fact, digital multimedia such as images and videos can be viewed as privacy-sensitive information in many service-based applications and hence can be abused if the processing or storage is conducted directly on plaintext data in the cloud. Such concerns about privacy protection have urged researchers to develop privacy-preserving systems in biometrics [4, 5], data compression [6, 7], and watermarking [8, 9]. On the other hand, Reversible Data Hiding (RDH) has emerged as an approach for embedding additional data into digital images imperceptibly with the ability to recover the host image

---

<sup>1</sup>Department of Computer and Information Sciences, Northumbria University, NE2 1XE, UK. E-mail: fouad.khelifi@northumbria.ac.uk; jungong.han@northumbria.ac.uk

<sup>2</sup>Department of Electronics, Jijel University, Jijel 18000, Algeria. E-mail:tbrahimi@univ-jijel.dz

<sup>3</sup>Center for OPTical IMagery Analysis and Learning (OPTIMAL), State Key Laboratory of Transient Optics and Photonics, Xi'an Institute of Optics and Precision Mechanics, Chinese Academy of Sciences, Xi'an 710119, Shaanxi, P. R. China. E-mail:xuelong\_li@opt.ac.cn

losslessly after extracting the additional data [10, 11, 12]. This differs from older versions of data hiding and watermarking where the process generates robust watermarks at the expense of some loss of information in the image content [13, 14]. A number of attempts have also considered RDH in the compressed domain [15, 16, 17]. Moreover, inspired by existing works on information processing in the encrypted domain, encryption was initially used in RDH systems to protect the content of the host image [18, 19, 20]. An interesting feature of such systems, called separability, has then been introduced for potential use in areas where the confidentiality of the host image is protected from the data hiding/extraction system while the confidentiality of the hidden message is protected from the image encryption/decryption system [21]. That is, the data hider can only process an encrypted version of the host image in order to embed additional data without accessing the image visual content while the encryption/decryption system can only encode and recover the original image without knowing the additional data. Therefore, the challenge from the data hider's perspective lies in the security and reversibility of the embedding process as well as the amount of data to embed in the encrypted domain. It is meant by reversibility here the ability of the decoder to recover the original host image in a lossless fashion.

A possible scenario for such systems in cloud computing has been suggested in many research papers [17, 22, 23, 24, 25]. Indeed, sensitive visual data such as biometric (face, fingerprint, iris, etc.) and medical images can be encrypted before they are sent to the cloud for storage. This ensures the confidentiality of the data since only the genuine cloud user can disclose the content of images through decryption. On the other hand, the Cloud Service Provider (CSP) may insert some meta data about the user and the date of upload in encrypted images to facilitate their management in the database (i.e. storage, search and retrieval). In the next section, we elaborate on this idea and provide a new model with complete analysis of security issues in cloud computing and how these can be addressed by such systems.

Existing RDH techniques in encrypted images, denoted here by RDH-EI, can be categorized in two classes depending on whether space for embedding the message is allocated before or after encryption [22]. The first attempts in the literature tend to find the embedding room after encryption. In [18], the authors adopted a block cipher method using the Advanced Encryption Standard (AES) algorithm to encrypt non-overlapping blocks, each consists of 16 gray level pixels. Each encrypted block is then used to embed one message bit by replacing a bit of the cipher text with the hidden bit. This system is, however, not separable because the location of the hidden bit in each block is required to recover the image at the decryption stage. The host image can be recovered by analyzing the local standard deviation of each block for two possible hidden bits (0 and 1). The idea of analyzing the neighborhood correlation to recover the host image was also explored in [19] and [20] where the stream cipher was used at the encryption stage. In [21], a separable RDH scheme operating in the encrypted domain was introduced. The technique creates room for embedding the additional data by compressing the least significant bits of the encrypted image. In [26], the idea of compressing half of the 4<sup>th</sup> Least Significant Bit (LSB) values of the encrypted image with the Low Density Parity Check (LDPC) codes method [27] has been shown to offer higher image recovery quality when compared with similar techniques. In a similar technique proposed in [28], the LDPC method has also been employed to compress the Most Significant Bits (MSB). In [29], the idea is to select a pixel among its four neighboring pixels in the encrypted image to embed a single bit. The reconstruction

of the original image relies on an improved context adaptive interpolation technique. However, the results show that encrypted images could be recovered with some loss of information when the embedding rate increases up to 0.16 bit per pixel (bpp). By adopting the bit flipping idea to embed the additional data and the statistical analysis of decrypted pixels for image recovery [19], the authors in [24] proposed to use a test on each block of encrypted pixels in order to decide on whether a bit could be embedded or not. This has been shown to improve the quality of the reconstructed image if the receiver has only the encryption/decryption key. However, their system is not separable as it requires the decrypted image before the extraction of the hidden data. In [30], the authors adopted an asymmetric encryption system, called Paillier homomorphic encryption, to develop a RDH-EI technique. The technique operates on pairs of pixels and encrypts the nearest and smaller even number of each pixel with the corresponding parity bit separately. Nonetheless, although the asymmetric encryption offers higher security when compared to the widely used stream cipher in RDH, this technique is not separable and produces an encrypted marked image with larger bits representation<sup>4</sup> than the original one due to the bandwidth overhead of the Paillier encryption system. The homomorphic encryption has also been adopted in [31] for separable RDH-EI but the problem of increasing the image bit budget was not addressed. In [32], the authors presented a non separable RDH technique. In this technique, both the content owner and data hider use stream cipher encryption in a cascade way (encryption by content owner followed by another encryption by the data hider). The authors used a number of public keys to conduct the data embedding process on encrypted image blocks by *re-encrypting* each block with one of the keys. That is, each key is linked to a sequence of bits which will represent a portion of the message. Given the fact that only one of the public keys can decrypt an encrypted block, the message can be extracted by analyzing the uniformity of blocks decrypted with all possible keys where an SVM classifier, trained on encrypted and unencrypted blocks, is used in order to find the right public key. Once the right key is determined for each block, the message can be deduced and the image can be recovered through a double decryption (i.e., using the public key and the encryption key). More recently, the authors in [33] proposed a histogram shifting method to hide additional data in the encrypted image. The technique divides the encrypted image into non-overlapping blocks and selects two pixels in each block as the block histogram peaks. The histogram is then expanded from the peaks towards the borders so that a bit can be embedded by shifting the pixels that are equal to one of the peak values accordingly.

In the other category of RDH-EI, the embedding space is created before encryption. The techniques that fall under this category rely on a bit saving representation of the image to create room using a lossy compression-like technique that explores inter-pixel and psycho-visual redundancy. In [22], the authors classify the image into two parts A and B where each part is described by one type of blocks, i.e., textured or smooth blocks. Then, the LSB-planes of the textured part, denoted by A, are reversibly embedded into B by using a traditional RDH algorithm. The image is then encrypted with a standard stream cipher and additional data can finally be embedded in the available bit-planes of A. A similar system has also been proposed in [34] where a predictive filter was used to estimate a number of selected pixels from their

---

<sup>4</sup>The image size increases in bits because the encryption system maps the range of pixels into a larger space.

neighborhood. The estimated pixels are then replaced with the corresponding estimation errors. Finally, the histogram of these errors is shifted to create the embedding space accordingly before encryption. In [25], the authors build upon the work of [34] and use an interpolation filter instead to predict pixels from their neighborhood. They also replace the predicted pixels with the corresponding prediction errors before encryption but the data hiding technique combines the idea of histogram shifting with expansion to create more room. The technique has been shown to deliver high capacity performance but this comes at the cost of slightly disclosing the visual content in the stego image due to un-encrypted interpolation errors. In [35], the authors adopted a sparse coding technique to create space for data hiding. The idea is based on the assumption that a patch in the image can be represented by a few atoms in an over-complete dictionary. Because sparse coding provides an approximate version of patches, each selected patch is described by its sparse code and a binary representation of the residual error in addition to a portion of the dictionary bits. The remaining bit space in each representation of the patch code is reserved for embedding the additional data.

Although the approach of RDH-EI appears to meet design requirements for secure and privacy-preserving data storage and sharing through the cloud, it still fails to reach high capacity of additional data insertion because the spatial redundancy in digital images is not fully exploited. In this paper, a model for secure data storage and sharing through the cloud is presented. To meet the design requirements of the cloud model, a new, simple, and efficient approach that does not require the process of RDH is proposed. We particularly argue that our approach exhibits the properties of data security and confidentiality in the cloud with higher capacity of additional data insertion as compared to RDH-EI. The technique relies on the idea of reserving room before encryption via a wavelet-based lossless image coder. Compared with state-of-the-art reversible data hiding systems in the encrypted domain, the proposed approach has been shown to achieve a significantly higher capacity of additional data insertion with suitable properties for the presented cloud. More specifically, the system offers lossless reconstruction of the host image without the CSP key while maintaining the bit budget of the original uncompressed image. Such a feature does not exist in current RDH-EI systems to the best of our knowledge. It is worth mentioning that the techniques in [15, 16] differ from the proposed system and RDHEI techniques in that a small payload is embedded in the JPEG file header without altering the compressed image content. The novelty of this work can be summarized as follows. (i) A new cloud model for secure and privacy-preserving data storage and sharing is presented. The model addresses a number of security issues and can be used in various practical applications. (ii) Unlike existing RDH-EI systems that reserve room *before* encryption, e.g. [25, 22, 34, 35], we propose to create room *losslessly* as image pixels do not undergo any lossy operations. This allows the content owner in the separable scheme to have a perfect reconstruction of the image using only the encryption key. It also saves the computational complexity since the process of self-embedding the residual error or a portion of the image content with a RDH technique as reported in [22, 34, 35] is no longer needed here. (iii) An efficient wavelet-based image coder is used to reserve room before encryption. This technique creates a significantly higher capacity than the current state-of-the-art. (iv) A shuffling scheme is used with stream cipher encryption on the compressed bit-stream for high randomness and de-correlation of samples. By doing so, the system's security can be significantly enhanced, and it turns out to be more suitable for cloud computing applications

than the conventional pixel domain-based stream cipher method in current RDH-EI systems.

The paper is structured as follows. Section 2 presents the proposed model for secure cloud computing along with design requirements. In section 3, the proposed approach for secure and privacy-preserving data sharing and management in the cloud is described in detail. Section 4 gives an experimental evaluation of the proposed system in comparison with RDH-EI systems. Section 5 summarizes the contributions and concludes the paper.

## 2. Problem formulation

In order to describe the system architecture and derive design requirements, we first highlight the features of the proposed framework for potential use in cloud computing. Fig. 1(a) illustrates our proposed model where a cloud user sends an image along with a message in encrypted form to the cloud. In health care applications, the image could be medical and the message may contain the medical record of the corresponding patient. In the field of law enforcement, the image could be biometric (face, fingerprint, iris, etc.) and the message would represent the criminal record of the corresponding suspect. To serve authorized users, the CSP acts as a secure data insertion system by adding different

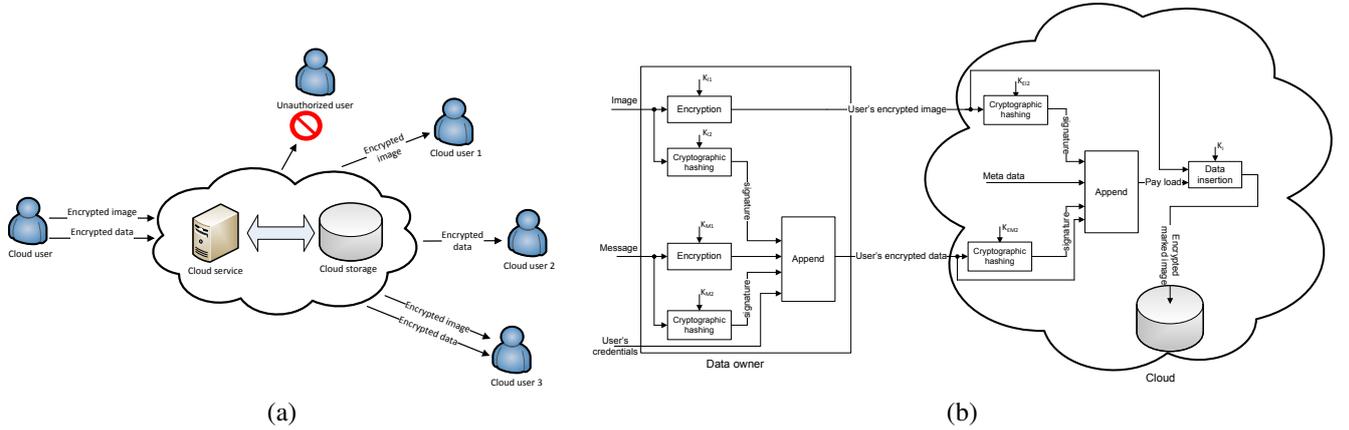


Figure 1: Proposed model for cloud computing. (a) Full and restricted access to data by cloud users. (b) Encryption and data insertion in the cloud.

types of data including the user's encrypted data into the encrypted image. In RDH-EI, this process is referred to as data embedding but as will be seen later, this is not required in our approach and hence, the term *data insertion* is used throughout the paper. Upon request, the CSP provides other cloud users with the encrypted image and/or the encrypted message depending on the access level they have been granted. It is clear that in order for the CSP to extract the inserted data and for the cloud user to recover the original image from the same source, i.e. the encrypted image containing the message, the message extraction and image recovery processes must be *separable*. Now let us describe how such a system can solve a number of security issues in cloud computing. First, let us recall from existing works [36][37] the security challenges in cloud computing which will be addressed in this paper.

- *Data confidentiality*: The data located in the cloud should be kept private. Indeed, any possible leakage of the stored data should not compromise its confidentiality. In the proposed model, confidentiality is guaranteed via encryption since all the data being handled in the cloud is already encrypted.

- *Data integrity from the user's perspective:* Images and digital messages might be altered in the cloud without the owner's consent. With the proposed solution, the image owner attaches a digital signature of the image and another signature of the message to the encrypted message data which will then be inserted in the reserved room of the encrypted image by the CSP. When the encrypted image and encrypted message are retrieved, the cloud user decrypts them and verifies their authenticity.
- *Data integrity from the CSP's perspective:* If a cloud user claims that images and/or messages have been altered, the CSP should verify this by his own means to protect the service from false claims. To this end, the CSP creates a signature of the encrypted image and a signature of the encrypted message and inserts them in the reserved room of the encrypted image. This enables the CSP to check the integrity of the data transmitted to cloud users.
- *Data access control:* The data owner, who is a cloud user at one side, might designate other cloud users who can share his data at the other side. To this end, the CSP manages authorized access to the data and prevents unauthorized users from illegal access. This can be guaranteed by inserting credentials in the reserved room of the encrypted image. Such credentials describe the authorized users and the level of access they have been granted (i.e. access to original encrypted image and/or encrypted message). A cloud user is required to send credentials to the cloud in order to access the data. If the credentials match the ones inserted in the reserved room of the encrypted image, data access is authorized.

A detailed demonstration of the aforementioned encryption and data insertion process for cloud computing is given in Fig. 1(b). Note that the data insertion system also adds some meta data (tagging information) to facilitate the storage and management of images in the cloud. The features of the proposed cloud computing model suggest the following design requirements of our system:

- *Bit budget constancy:* One of the main design requirements the presented model is that the encrypted image containing additional data must be represented at the same bit cost as the original uncompressed image.
- *High capacity:* As shown before, the data insertion system adds different types of data in the encrypted image. The room for adding such data must be sufficiently large.
- *Reversibility for authorized users with full access:* Upon request, the CSP must reconstruct the original encrypted image and the owner's encrypted data from the encrypted image containing additional data. This will allow authorized users to recover the original image and the message via decryption.
- *Separability:* Since authorized users have different levels of access, the system should ensure that the original image can be recovered<sup>5</sup> from the encrypted image containing additional data without the CSP key. On the other hand, the inserted data can be extracted without the image encryption key.

---

<sup>5</sup>Image recovery here stands for lossy or lossless reconstruction.

- *Lossless image recovery without the CSP key*: From the encrypted image containing additional data and by just using the image encryption key, authorized users who have been granted access only to the owner’s encrypted image must be able to reconstruct a lossless version of the original image. A lossy version would not be suitable in medical and biometric applications.

In the next section, the proposed system is described in detail.

### 3. Proposed approach

The proposed reversible and separable privacy-preserving data sharing scheme is illustrated by Fig. 2. First, the data

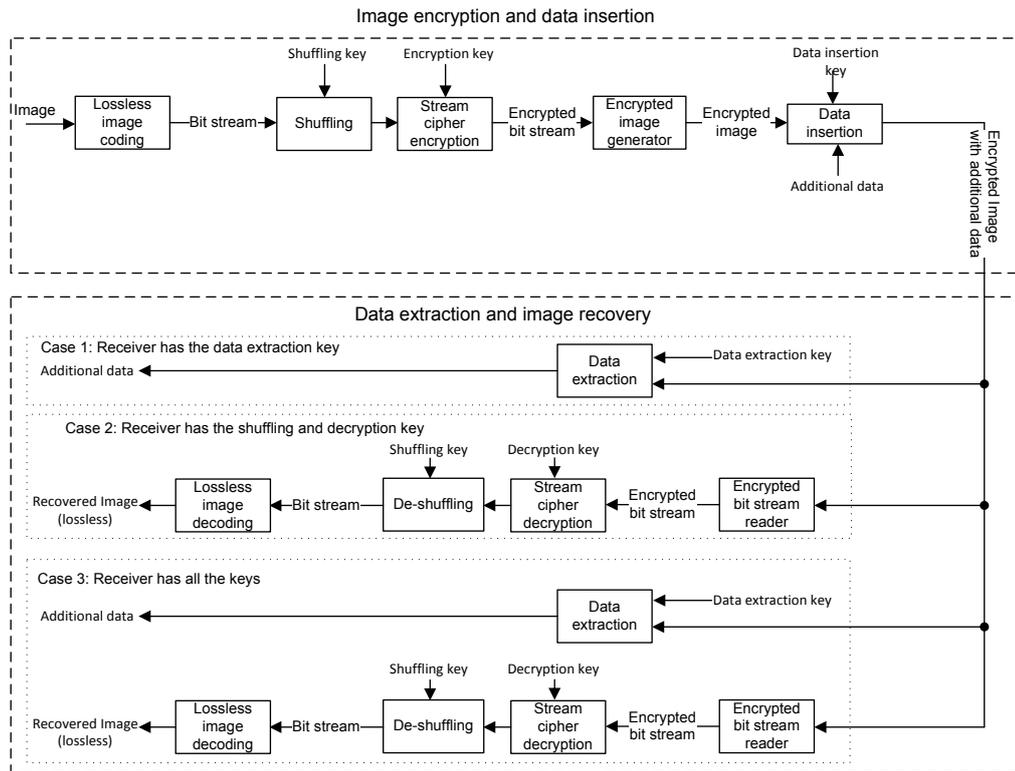


Figure 2: Proposed system for secure and privacy-preserving data management and sharing through the cloud.

insertion room is created before encryption via a lossless image coder. To secure the image content, the compressed bit stream is then shuffled and encrypted prior to the construction of the encrypted image. Since the compressed bit stream is expected to be smaller than the original one, the reserved space for data insertion corresponds to the difference in size. That is, the encrypted image is created by using a representation in bytes <sup>6</sup> of the compressed bit stream and filling the remaining space (reserved insertion space) with a randomly generated sequence to obtain the same size in terms of resolution as the original one. It is worth noting here that the length of the bit stream is also included in the encrypted image so that the CSP knows the room reserved for inserting the additional data <sup>7</sup>. The CSP securely uses

<sup>6</sup>Here, a byte stands for 8 bits

<sup>7</sup>In this paper, we use the term "additional data" to refer to the payload added to the encrypted image as illustrated by Fig. 1(b).

the reserved space to insert the additional data using a secret CSP key. At the receiver end, the process is reversed to extract the additional data and/or recover the image depending on the available keys. Note that the proposed system uses a *lossless* operation (i.e. lossless image compression) to reserve room. Indeed, the *data insertion* room is created *without modifying* the image content. This is one of the key differences between our work and existing RDH-EI systems that reserve room before encryption such as [25, 22, 34, 35]. It also supports our claim in this paper that the RDH process is not required to achieve the design requirements for the cloud model listed above. The advantage of this property is twofold. First, the computational complexity is reduced since the reversible self-embedding of a portion of the image content is no longer needed for lossless image recovery. Second, the authorized cloud user can recover the original image without distortions using only the decryption key. Moreover, the current state-of-the-art lossless image coding constitutes an efficient alternative that can offer a significantly larger *insertion* space than the *embedding* room in the conventional RDH-EI approach.

### 3.1. Image encryption and data insertion

#### 3.1.1. Lossless wavelet-based image coding

The proposed approach can operate with any lossless image coder. However, for high performance, the Embedded Block Coding with Optimized Truncation (EBCOT) algorithm, developed by Taubman [38] is adopted in this paper. Other powerful coders such as the Context-based Adaptive Lossless Image Coder (CALIC) [39] and Low COMplexity LOSSless COMpression for Images (LOCO-I) [40] could also be used but without the desirable properties of resolution and quality scalability offered by the wavelet transform in EBCOT. While exhibiting superior compression performance to existing standards and state-of-the-art coding techniques, EBCOT uses the wavelet transform and offers distinctive features such as lossless and lossy compression with a single bit-stream, progressive transmission (in quality and/or in resolution), region-of-interest (ROI) coding, open architecture, robustness to bit errors, and protective image security [41]. The wavelet transform has a de-correlating and compressive nature in the sense that the signal energy is mostly concentrated in a small number of coefficients in the low-frequency sub-band. In this work, we use EBCOT and adopt the reversible integer  $5/3$  wavelet implemented in the lifting scheme (LS), since it enables to losslessly compress the image. The lifting structure, which is a spatial domain wavelet construction, simplifies the construction process of reversible wavelet transforms and ensures efficient hardware implementation without using the Fourier transform. Also, the concepts of translation and dilation of the mother wavelet are no longer used. In comparison with classical wavelets, the lifting scheme offers distinct advantages since it carries out a reduced number of arithmetical operations, with a significantly lower computational complexity [42]. This helps to speed up the implementation process, and regain the memory usage. Furthermore, the inverse wavelet transform can be achieved directly by undoing the operations of the forward transform. The integer version of the LS gains a significant advantage especially for designing efficient lossless progressive compression systems requiring a perfect reconstruction by means of rounding operators combined with lifting constructions. This ensures reversibility in the transform which maps integers to integers. Let  $x$  be the input signal which can be split into two sample groups denoted by even and odd index signals, i.e.,  $x_e = x(2j)$  and  $x_o = x(2j + 1)$ , respectively. Each group comprises

half the size of the original signal. In this paper, the lossless implementation takes advantage of the bi-orthogonal Le Gall 5/3 wavelet. The 5/3 transform belongs to the class of symmetric, bi-orthogonal wavelets, constructed from the interpolating Deslauriers-Dubuc scaling functions [43]. The lifting structure of the 5/3 wavelet filter performs two lifting steps only. The forward transform is obtained as follows

$$d(j) = x_o(j) - \left\lfloor \frac{1}{2}(x_e(j) + x_e(j+1)) + \frac{1}{2} \right\rfloor \quad (1)$$

and

$$s(j) = x_e(j) + \left\lfloor \frac{1}{4}(d(j-1) + d(j)) + \frac{1}{2} \right\rfloor \quad (2)$$

where  $\lfloor \cdot \rfloor$  indicates the floor operation. The multiplierless reversible 5/3 transform is also known as (2, 2) with reference to its 2 vanishing moments for both the analyzing and the synthesizing high pass filters and exhibits a very low computational complexity since only 6 additions and 2 shifts are involved. Compared to the Daubechies filter of the same order, the 5/3 transform has been shown to offer better approximation properties [44]. To form a 2-D pyramid structure, the transform is obtained by successively applying the one-dimensional transform in (1) and (2) along the rows and columns of the input image.

Denote by  $M \times N$  the size of an input image<sup>8</sup> encoded with  $\lambda$  bpp where  $\lambda$  is an integer. In the rest of the paper, the length of the original binary sequence is denoted by  $\ell$  and is equal to  $M \times N \times \lambda$ . The length of the compressed bit stream is denoted by  $\ell'$ . Obviously, if the image is compressible<sup>9</sup>,  $\ell' < \ell$ .

### 3.1.2. Shuffling and stream cipher

A shuffling process is conducted on the compressed bit stream so that the sequence bits are rearranged randomly according to a secret key  $K_{sh}$ . The shuffled sequence  $S_{sh}(i)$ , ( $i = 1, \dots, \ell'$ ) can be expressed as

$$S_{sh} = \Phi(S_c, K_{sh}) \quad (3)$$

where  $\Phi$  denotes the shuffling process and  $S_c$  is the compressed bit stream. If  $S_{sh}$  is arranged column wise, then (3) can be re-written as

$$S_{sh} = \Upsilon \times S_c \quad (4)$$

where  $\Upsilon$  is a sparse invertible matrix of size  $\ell' \times \ell'$  with integers in  $[0, 1]$  so that

$$\forall i \in \{1, 2, \dots, \ell'\}, \sum_{j=1}^{\ell'} \Upsilon(i, j) = 1 \quad (5)$$

<sup>8</sup>Without loss of generality, we consider gray level images in this paper.

<sup>9</sup>Theoretically speaking, natural images are compressible because they exhibit redundant information across neighboring pixels (a kind of correlation which can be exploited by the coder) and thus a non compressible image would correspond to a white Gaussian noise.

This matrix is pseudo randomly created and represents the shuffling key  $K_{sh}$ . It can be shown that  $\Upsilon \times \Upsilon^T = \mathbb{I}_{\ell'}$  where  $T$  stands for the transpose operation and  $\mathbb{I}_{\ell'}$  is the identity matrix. The shuffling process reduces the correlation that may exist in neighboring samples. For the sake of illustration, denote by  $X_h$  a shifted version of the original image  $X$  in the spatial domain through the horizontal direction. Likewise,  $X_v$  represents its shifted version in the vertical direction, i.e.,

$$X_h(i, j) = X(i, j + 1 \pmod N) \quad (6)$$

and

$$X_v(i, j) = X(i + 1 \pmod M, j) \quad (7)$$

where  $M$  and  $N$  represent the image size. Let  $\rho_h$  and  $\rho_v$  be the correlation coefficients between  $X$  and its shifted versions as

$$\rho_k = \frac{E[(X - \mu_X)(X_k - \mu_{X_k})]}{\sigma_X \sigma_{X_k}} \quad (8)$$

where  $\mu_X$  represents the statistical mean of  $X$  and  $\sigma_X$  is the corresponding standard deviation while  $k \in \{h, v\}$ . Table 1 depicts the values of  $\rho_h$  and  $\rho_v$  *before* and *after* shuffling a number of standard images. As can be seen, the inter pixel

Table 1: Correlation coefficient before and after shuffling.

Image	Coefficient	Before	After
Lena	$\rho_h$	0.9691	0.0006
	$\rho_v$	0.9841	-0.0036
Barbara	$\rho_h$	0.8940	-0.0013
	$\rho_v$	0.9572	-0.0023
Peppers	$\rho_h$	0.9755	-0.0011
	$\rho_v$	0.9808	-0.0002
Baboon	$\rho_h$	0.8653	-0.0007
	$\rho_v$	0.7523	0.0002

correlation decreases significantly. Next,  $S_{sh}$  is further subjected to encryption with a stream cipher as follows. By using the encryption key, a random binary sequence, also called the key stream,  $r(i)$ , ( $i = 1, \dots, \ell'$ ) can be generated. The encrypted bit stream  $S_{en}(i)$ , ( $i = 1, \dots, \ell'$ ) is given by

$$S_{en}(i) = S_{sh}(i) \oplus r(i) \quad (9)$$

where  $\oplus$  represents the bit-wise exclusive OR (XOR) operation.

Note that the Known Plaintext Attack [45] (KPA) cannot take place on the encryption scheme since the original image is not accessible by the CSP whereas both the original image and its encrypted version are needed to perform such an attack. The only attack that could be applied in this case is the Ciphertext Only Attack [46] where the attacker (i.e., CSP here) can access a number of encrypted images. This will be discussed in section 4.2. It is worth mentioning, however, that the process of shuffling followed by a stream cipher may not be sufficient for securing the data in other applications. Indeed, if the attacker had access to a number of original bit streams and their encrypted and shuffled versions, it would be possible to estimate the shuffling matrix  $\Upsilon$  and the key stream  $r(i)$  as follows. Let  $\omega$  be the number of pairs (i.e.,

original/shuffled sequences). Since  $\Upsilon$  is a sparse invertible matrix with values in  $\{0, 1\}$  and the stream cipher either keeps a shuffled bit or flips it, the attacker looks non zero values only in the shuffling matrix (i.e.  $\Upsilon(i, j) = 1$ ). Let us first define the probability that the value of the shuffling matrix at  $(i, j)$  corresponds to 1 under the assumption that the resulting bit is not flipped by the stream cipher

$$Prob(i, j) = \frac{1}{\omega} \sum_{\nu=1}^{\omega} \Theta_{i,j}(\nu) \quad (10)$$

where

$$\Theta_{i,j}(\nu) = \begin{cases} 1 & \text{if } S_{sh}^{\nu}(i) = S_c^{\nu}(j) \\ 0 & \text{Otherwise} \end{cases} \quad (11)$$

If the flipping process caused by encryption is taken into account, the probability in (10) could be either 1 or 0 for non zero values of  $\Upsilon(i, j)$  whereas for the values that are equal to zero it should be close to 0.5. Therefore, one can estimate the shuffling matrix as

$$\hat{\Upsilon}(i, j) = \begin{cases} 1 & \text{if } j = \arg(\max_{\alpha \in \{1, 2, \dots, \ell'\}} (Prob(i, \alpha), 1 - Prob(i, \alpha))) \\ 0 & \text{Otherwise} \end{cases} \quad (12)$$

where  $i \in \{1, 2, \dots, \ell'\}$ . Finally, the stream key  $\hat{r}(i)$  can be determined as

$$\hat{r}(i) = \begin{cases} 1 & \text{if } \max(Prob(i, \zeta(i)), 1 - Prob(i, \zeta(i))) < \max(1 - Prob(i, \zeta(i))) \\ 0 & \text{Otherwise} \end{cases} \quad (13)$$

where  $\zeta(i) = \arg(\hat{\Upsilon}(i, \alpha) = 1)$ .  
 $\alpha \in \{1, 2, \dots, \ell'\}$

### 3.1.3. Creation of encrypted image

Recall that the original bit stream is composed of  $\ell$  bits. That is, the encrypted image generator needs to add  $(\ell - \ell')$  bits in order to construct an encrypted image with the same size as the original one. Also, the CSP must know the space reserved for inserting the additional data to ensure that the compressed bit stream does not get affected. We propose to arrange the bit stream in the following way. First, since  $\ell' < \ell$  the number of bits  $n_b$  to describe  $\ell'$  is fixed at

$$n_b = \lfloor \log_2(\ell) \rfloor \quad (14)$$

where  $\lfloor \cdot \rfloor$  denotes the floor operation. Let  $S_{\ell'}$  be the binary representation of  $\ell'$ . The encrypted image generator creates a sequence  $S'_{en}$  by selecting  $\ell - \ell' - n_b$  bits from  $S_{en}$  randomly. The bit stream used to construct the encrypted image is a concatenation of  $S_{en}$ ,  $S'_{en}$  and  $S_{n_b}$  as shown in Fig. 3. Note that the maximum number of bits which can be modified without affecting the image content is  $\ell - \ell'$ . Finally, each byte in the bit stream corresponds to a pixel of the encrypted image.

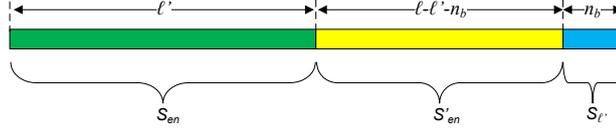


Figure 3: Structure of the bit stream used to generate the encrypted image.

### 3.1.4. Data insertion

Given a binary message (i.e. additional data)  $S_m$  of length  $q \leq \ell - \ell' - 2n_b$ , the CSP encrypts  $S_m$  with the secret key  $K_{di}$  to obtain a sequence  $S'_m$  as

$$S'_m = \Psi(S_m, K_{di}) \quad (15)$$

where  $\Psi$  represents the encryption function. It is worth mentioning that the encryption method described in section 3.1.2 (i.e. shuffling and stream cipher) cannot be used here to encrypt the message because the cloud user can apply a "Chosen Plaintext Attack" [47] (CPA) to estimate both the CSP key and the shuffling key since he has access to the encrypted image that contains additional data<sup>10</sup>. This is described as follows. If the shuffling and stream cipher are applied, then

$$S'_m = (\Gamma \times S_m) \oplus V \quad (16)$$

where  $S_m$  is the plaintext message arranged column wise and  $V$  is a random sequence (i.e. a random column vector) of length  $q$  generated with the stream cipher key.  $\Gamma$  is a sparse invertible matrix of size  $q \times q$  with integers in  $[0, 1]$  so that

$$\forall i \in \{1, 2, \dots, q\}, \sum_{j=1}^q \Gamma(i, j) = 1 \quad (17)$$

This is a pseudo random matrix that represents the shuffling key. Note that  $\Gamma \times \Gamma^T = \mathbb{I}_q$ . The matrix product  $(\Gamma \times S_m)$  in (16) describes the shuffling operation. Mathematically speaking, breaking the shuffling and stream cipher systems means the estimation of  $\Gamma$  and  $V$ . This is because from (16) one can estimate the plaintext message from the ciphertext as

$$S_m = \Gamma^T \times (S'_m \oplus V) \quad (18)$$

Because the cloud user (acting as an attacker here) can have full control over the data to insert (see Fig. 1(b)), he may choose a message as follows. Denote by  $S_m^n$ ,  $n \in \{1, 2, \dots, q\}$  a sequence derived from  $S_m$  as

$$S_m^n(i) = \begin{cases} S_m(i) & \text{if } i \neq n \\ \tilde{S}_m(i) & \text{Otherwise} \end{cases} \quad (19)$$

<sup>10</sup>Note that the cloud user creates additional data and sends it to the CSP (see Fig. 1) before he receives the encrypted image containing additional data.

where  $\tilde{\cdot}$  represents the bit flipping operation and  $i \in \{1, 2, \dots, q\}$ . Let  $S_m^{\prime n}$  be the shuffled and encrypted version of  $S_m^n$  with the same shuffling and encryption keys by using (16). That is,

$$S_m^{\prime n} = (\Gamma \times S_m^n) \oplus V \quad (20)$$

Because  $S_m^{\prime}$  and  $S_m^{\prime n}$  differ in only one bit (and so is the difference between  $S_m$  and  $S_m^n$ ), one can deduce the only non-zero value (that is equal to one) in the  $k^{th}$  row of  $\Gamma$  for each value of  $n$  as

$$\Gamma(k, j) = \begin{cases} 0 & \text{if } j \neq n \\ 1 & \text{Otherwise} \end{cases} \quad (21)$$

where  $k$  is the index for which  $S_m^{\prime}(k) \neq S_m^{\prime n}(k)$ . In view of (21), it is clear that by varying  $n$  in  $[1, q]$  the attacker can construct  $\Gamma$  completely. Once  $\Gamma$  is estimated, the attacker uses (16) to estimate  $V$  as follows

$$V = (\Gamma \times S_m) \oplus S_m^{\prime} \quad (22)$$

In this paper, the Advanced Encryption Standard (AES) [48] is used for encrypting  $S_m$  with a key length of 128. This is a symmetric block cipher with a block size of 128 bits. AES is nowadays broadly adopted and supported in both hardware and software. Furthermore, no practical cryptanalytic attacks against AES have been reported in the literature. The algorithm operates on blocks of  $4 \times 4$  bytes through a number of rounds. (10 rounds for a 128-bit key). Each of these rounds makes use of a different 128-bit round key, which is derived from the original AES key (This process is known as key expansion). In each round, the bytes are substituted using a fixed table called S-box and then shifted. Next, the columns of the  $4 \times 4$  bytes are combined using an invertible linear transformation before they are XORed to the 128 bits of the round key. The decryption process follows the same steps in the reverse order [48].

Denote by  $S_q$  the binary representation of  $q$ . We also propose to use  $n_b$  bits to represent  $S_q$ . The CSP then reads the encrypted image in binary form and substitutes the first  $q$  bits of  $S_{en}^{\prime}$  for  $S_m^{\prime}$ . Also, the last  $n_b$  bits of  $S_{en}^{\prime}$  are replaced with  $S_q$  so that the receiver knows the length of the inserted sequence. The encrypted marked bit stream is illustrated by Fig. 4. In the same way as described in subsection 3.1.3, the creation of the encrypted image that contains additional data

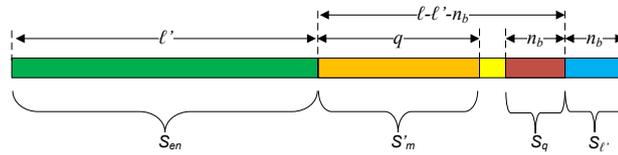


Figure 4: Structure of the bit stream used to generate an encrypted image containing additional data.

uses the idea of byte representation in the bit stream to make pixels. The Insertion Rate ( $IR$ ) measures the capacity of

data insertion in bits per pixel (bpp) and is defined as

$$IR = \frac{\lambda \times (\ell - \ell' - 2n_b)}{\ell} \quad (23)$$

### 3.2. Data extraction and image recovery

Upon receipt of the encrypted marked image, the receiver converts it into the corresponding bit stream and extracts the inserted data and/or recover the image according to the keys available as follows.

#### 3.2.1. Data extraction

The receiver reads  $S_{\ell'}$  and  $S_q$  to determine the start and end point of the encrypted sequence  $S'_m$  in the received bit stream. Here, the data extraction key represents the encryption key that was used to encrypt and append additional data. Therefore, the receiver applies the AES decryption algorithm using  $K_{di}$  to obtain the inserted sequence  $S_m$  as

$$S_m = \Psi^{-1}(S'_m, K_{dh}) \quad (24)$$

where  $\Psi^{-1}$  represents the AES decryption process.

#### 3.2.2. Image recovery

The receiver reads  $\ell'$  to determine the end point of the shuffled and encrypted bit stream  $S_{en}$ . Here, the decryption key is the same as the encryption key (i.e., symmetric encryption). With the decryption key, the random binary sequence  $r(i), (i = 1, \dots, \ell')$  can be generated. The shuffled sequence  $S_{sh}$  can be obtained by decrypting  $S_{en}$  as

$$S_{sh}(i) = S_{en}(i) \oplus r(i) \quad (25)$$

Next,  $S_{sh}$  can be de-shuffled to get the compressed bit stream  $S_c$  as

$$S_c = \Phi^{-1}(S_{sh}, K_{sh}) \quad (26)$$

where  $\Phi^{-1}$  is the de-shuffling function. Finally, the lossless image decoder is used to decompress  $S_c$  and recover the original image.

#### 3.2.3. Data extraction and image recovery

If all the keys are available, the receiver can interchangeably extract the additional data and recover the original image. First, by reading  $S_{\ell'}$  and  $S_q$ , the receiver knows the start point and length of each of the sequences that represent the image and additional data. The same keys that were used to encrypt the image and additional data will serve for data extraction and image recovery as follows. The AES decryption algorithm is used with the key  $K_{di}$  to extract the sequence  $S_m$  as given by (24). Then, the random binary sequence  $r(i), (i = 1, \dots, \ell')$  can be generated from the decryption key and

the shuffled sequence  $S_{sh}$  can be obtained by decrypting  $S_{en}$  as in (25). Once  $S_{sh}$  is obtained, a de-shuffling process is conducted to deduce the compressed bit stream  $S_c$  as given by (26). Eventually, the decompression of  $S_c$  with the lossless image decoder gives the original image.

#### 4. Experimental analysis and discussion

In this section, the performance of the proposed system is analyzed in two different aspects: *data insertion capacity* and *system security*.

##### 4.1. Data insertion capacity

Here, the data insertion capacity, measured by  $IR$ , is evaluated on a number of  $512 \times 512$  gray level standard images. It is worth noting that the insertion rate strongly depends on the performance of the lossless coder, i.e. the better the compression performance, the higher the insertion rate. Table. 2 depicts the values of  $IR$  for images of different contents. As can be seen, the insertion rate exceeds 3 bpp for all images except the 'Baboon' image which is characterized by highly

Table 2: Obtained Insertion Rate on standard images.

Image	Lena	Barbara	Peppers	F-16	Baboon	Goldhill	Boat	Couple	Average
$ER$ (bpp)	3.69	3.22	3.38	4.02	1.89	3.16	3.60	3.16	3.26

textured and edged regions. Such images are difficult to compress and hence the room reserved for data insertion is smaller when compared to smooth and less textured images. We note, however, that the obtained insertion rate for 'Baboon' is significantly higher than the ones reached by existing RDH-EI systems as will be shown later. Fig. 5 illustrates the original 'Peppers' image, its encrypted version, and the encrypted one containing additional data at an insertion rate of 3.38 bpp with our proposed system. To illustrate the significance of our contributions, the features of the proposed system

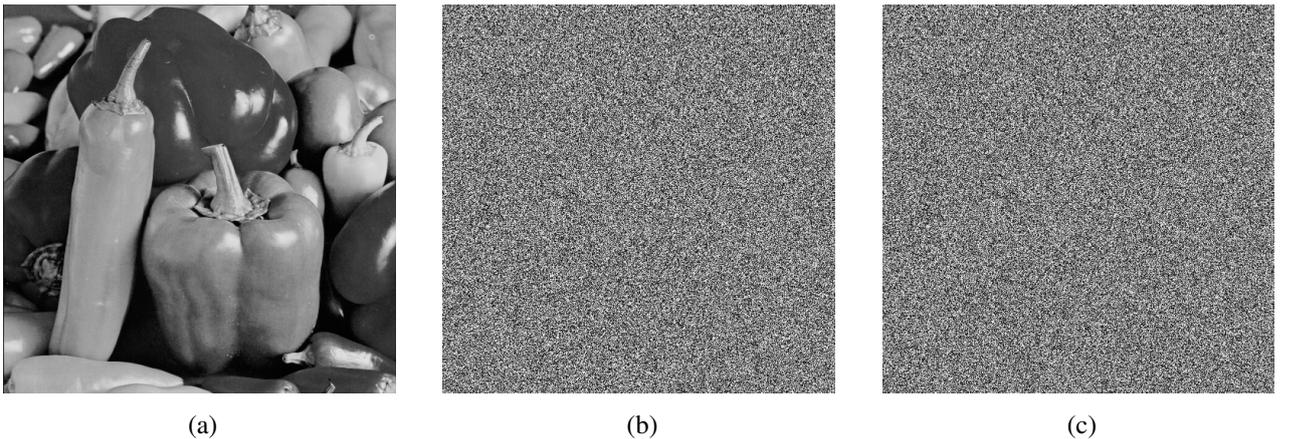


Figure 5: Data insertion in encrypted 'Peppers' at 3.38 bpp. (a) Original image. (b) Encrypted image. (c) Encrypted image with additional data.

as described in section 2, i.e., *bit budget constancy*, *capacity*, *reversibility*, *separability*, and *lossless image recovery* are listed in Tables 3-6 along with state-of-the-art techniques that adopt the RDH approach. We have also included the Peak Signal to Noise Ratio (PSNR) as a quality measure of the image recovered by using the encryption key only. The results

Table 3: Comparison between the proposed approach and existing RDH-EI systems on 'Lena'.

System	<i>bit budget constancy</i>	<i>capacity (bpp)</i>	<i>reversibility</i>	<i>separability</i>	<i>lossless image recovery</i>	PSNR (dB)
[18]	Yes	0.062	Yes	No	No	Not given
[19]	Yes	0.004	Yes	No	No	37.94
[20]	Yes	0.005	Yes	No	No	38.08
[21]	Yes	0.017	Yes	Yes	No	39.00
[22]	Yes	0.5	Yes	Yes	No	39.53
[24]	Yes	0.004	Yes	No	No	38.59
[25]	Yes	0.18	Yes	Yes	No	49.90
[26]	Yes	0.101	Yes	Yes	No	38.30
[28]	Yes	0.295	Yes	Yes	No	63.10
[29]	Yes	0.062	Yes	Yes	No	33.06
[30]	No	0.5	Yes	No	No	39.83
[31]	No	0.996	Yes	Yes	Yes	$\infty$
[34]	Yes	0.04	Yes	Yes	No	55.87
[35]	Yes	0.61	Yes	Yes	No	40.00
Proposed	<b>Yes</b>	<b>3.69</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>	$\infty$

Table 4: Comparison between the proposed approach and existing RDH-EI systems on 'F-16'.

System	<i>bit budget constancy</i>	<i>capacity (bpp)</i>	<i>reversibility</i>	<i>separability</i>	<i>lossless image recovery</i>	PSNR (dB)
[18]	Yes	0.062	Yes	No	No	Not given
[22]	Yes	0.5	Yes	Yes	No	42.19
[25]	Yes	0.18	Yes	Yes	No	49.92
[29]	Yes	0.156	Yes	Yes	No	32.38
[30]	No	0.5	Yes	No	No	39.84
[31]	No	0.996	Yes	Yes	Yes	$\infty$
[34]	Yes	0.04	Yes	Yes	No	58.25
[35]	Yes	1.0	Yes	Yes	No	40.00
Proposed	<b>Yes</b>	<b>4.02</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>	$\infty$

of the competing systems are given on four well-known  $512 \times 512$  standard images as reported in the literature. It is worth mentioning here that the systems which rely on pixel variation in small spatial regions to recover the original image such as [19], [20], and [29] fail to produce a lossless version of 'Baboon' and to extract additional data correctly at an embedding rate of 0.016 bpp even if the receiver possesses all the keys. This is mainly due to the content of this image which is highly textured. As a result, the difference in neighboring pixel values appears similar before and after data hiding. In the tables provided below, the data insertion capacity of such systems is calculated so that the reversibility is guaranteed. As can be seen, the proposed approach offers significantly better performance than the RDH-based approach and provides suitable features for the aforementioned model of cloud computing. Observe that [31] is the only system among existing ones in the literature that can provide lossless image recovery with the encryption key only.

Table 5: Comparison between the proposed approach and existing RDH-EI systems on 'Baboon'.

System	<i>bit budget constancy</i>	<i>capacity (bpp)</i>	<i>reversibility</i>	<i>separability</i>	<i>lossless image recovery</i>	PSNR (dB)
[18]	Yes	0.062	Yes	No	No	Not given
[19]	Yes	0.001	Yes	No	No	37.92
[20]	Yes	0.001	Yes	No	No	38.37
[21]	Yes	0.017	Yes	Yes	No	31.86
[22]	Yes	0.5	Yes	Yes	No	30.19
[24]	Yes	0.004	Yes	No	No	39.03
[25]	Yes	0.054	Yes	Yes	No	49.51
[28]	Yes	0.295	Yes	Yes	No	46.20
[30]	No	0.5	Yes	No	No	39.84
[31]	No	0.996	Yes	Yes	Yes	$\infty$
[34]	Yes	0.012	Yes	Yes	No	55.76
Proposed	<b>Yes</b>	<b>1.89</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>	$\infty$

Table 6: Comparison between the proposed approach and existing RDH-EI systems on 'Peppers'.

System	<i>bit budget constancy</i>	<i>capacity (bpp)</i>	<i>reversibility</i>	<i>separability</i>	<i>lossless image recovery</i>	PSNR (dB)
[18]	Yes	0.062	Yes	No	No	Not given
[19]	Yes	0.005	Yes	No	No	38.03
[20]	Yes	0.005	Yes	No	No	38.05
[21]	Yes	0.034	Yes	Yes	No	31.82
[22]	Yes	0.5	Yes	Yes	No	36.87
[25]	Yes	0.126	Yes	Yes	No	49.76
[30]	No	0.5	Yes	No	No	39.83
[31]	No	0.996	Yes	Yes	Yes	$\infty$
Proposed	<b>Yes</b>	<b>3.38</b>	<b>Yes</b>	<b>Yes</b>	<b>Yes</b>	$\infty$

However, as mentioned earlier, this system uses an asymmetric encryption technique involving a bandwidth overhead and consequently the size of the ciphertext (i.e. encrypted image) is larger than the size of the plaintext (i.e. original image). The comparative results show that RDHEI techniques cannot offer an optimal solution for cloud applications that involve digital images. It is, however, worth mentioning that some techniques that use the RDH-based approach could be used effectively on uncorrelated signals, such as textual data, in which correlation within the adjacent samples cannot be exploited by compression-based methods. In this context, the techniques proposed in [30, 31, 32] could be applied effectively in such scenarios.

Further enhancements to the proposed system could be obtained in terms of the data insertion capacity. Indeed, the EBCOT lossless coder has been shown in [49][50] to perform better than CALIC and LOCO-I<sup>11</sup> on color images. It has also been adopted by the FBI for compression of high-resolution gray level fingerprint images. However, results reported in [51][52] have revealed the superiority of CALIC over EBCOT and LOCO-I on medical and biomedical images. Therefore, the type of images used in the cloud should be taken into consideration in order to optimize the performance of the system.

#### 4.2. Security analysis

There are two main components in the proposed system for cloud applications: the cloud user who sends an encrypted image along with additional data to the CSP on one hand. On the other hand, the CSP who inserts additional data in the encrypted image and grants access to the encrypted image that contains additional data upon request. Therefore, security requires that the image content cannot be disclosed by the CSP and the inserted data cannot be extracted by the cloud user [53]. Since the AES encryption system is adopted for encrypting additional data (see subsection 3.1.4), the cloud user will have to break the AES in order to extract the inserted data. To the best of our knowledge, the only attack on AES that has been reported in the literature is called the eXtended Sparse Linearization (XSL) attack [54] and this has been shown to be inefficient in practice [55]. Thus, the CSP side can be considered highly secure. As for the security of the cloud user's data (image), the proposed technique takes advantage of the randomness and de-correlation that the compressed bit stream exhibits to enhance security. In fact, from the attacker's perspective, encrypting a random uncorrelated sequence makes the task of estimating its content harder than a correlated and uniform sequence. In current

<sup>11</sup>Note that LOCO-I has been standardized as JPEG-LS.



Figure 6: Result of XORing two images. (a) Baboon. (b) Boat. (c) XORed images.

systems that adopt the RDH approach [19, 20, 21, 22, 24, 26, 28, 29, 35], the stream cipher is applied on the image using its spatial representation. However, because the CSP can observe multiple encrypted images, he can apply an attack as follows. Denote by  $S_o^i$ ,  $i \in \{1, 2\}$  the  $i^{th}$  original sequence and by  $S_e^i$  its encrypted version. Then

$$\begin{cases} S_e^1 = S_o^1 \oplus K_e \\ S_e^2 = S_o^2 \oplus K_e \end{cases} \quad (27)$$

where  $K_e$  is the key stream. In view of (27), we have

$$\begin{aligned} S_e^1 \oplus S_e^2 &= S_o^1 \oplus K_e \oplus S_o^2 \oplus K_e \\ &= S_o^1 \oplus S_o^2 \end{aligned} \quad (28)$$

That is, XORing two encrypted sequences cancels the key and produces the same result as if the original sequences are XORed. Now, knowing the XORing result of two images in the compressed domain does not allow the attacker to disclose their contents because it is meaningless to the decoder. On the contrary, this result is meaningful if the images are XORed in the spatial domain. In Fig. 6, the result of XORing two images in the spatial domain is displayed. As can be seen, the content of the two images is partially disclosed. This illustrates a clear security weakness in previous works which can be remedied here in the compressed domain.

## 5. Conclusion

In this paper, a model for storing and sharing data securely through the cloud has been presented. RDH-EI has been recently suggested as an approach to ensure secure and privacy-preserving applications for data sharing and management in the cloud, limitations have been reported on the efficiency of conducting RDH in such applications. To address security issues in the cloud more efficiently, a new approach that does not use the process of RDH has been proposed. This is based on the idea of reserving room before encryption via a wavelet-based lossless image coder. Compared with state-

of-the-art RDH-EI systems, the proposed approach has been shown to offer a significantly higher data insertion capacity with suitable features for the presented cloud model. Furthermore, our analysis shows that the CSP can perform an attack to partially disclose the content of images encrypted with a stream cipher in the existing RDH-EI approach. Alternatively, the proposed approach protects the image content in a more reliable way as the stream cipher is applied on the compressed bit-stream.

## References

- [1] H. Hu, Y. Wen, T.-S. Chua, X. Li, Toward scalable systems for big data analytics: A technology tutorial, *IEEE Access* 2 (2014) 652–687.
- [2] Y. Luo, D. Tao, K. Ramamohanarao, C. Xu, Y. Wen, Tensor canonical correlation analysis for multi-view dimension reduction, *IEEE Transactions on Knowledge and Data Engineering* 27 (2015) 3111–3124.
- [3] Y. Luo, Y. Wen, D. Tao, Heterogeneous multitask metric learning across multiple domains, To appear in *IEEE Transactions on Neural Networks and Learning Systems*, accessed on <http://ieeexplore.ieee.org/document/8058002/>.
- [4] Y. Wang, K. N. Plataniotis, An analysis of random projection for changeable and privacy-preserving biometric verification, *IEEE Transactions on Systems, Man, and Cybernetics-Part B, Cybernetics* 40 (2010) 1280–1293.
- [5] M. Lim, P. Yuen, Entropy measurement for biometric verification systems, *IEEE Transactions on Cybernetics* 46 (2016) 1065–1077.
- [6] M. Johnson, P. Ishwar, V. M. Prabhakaran, D. Schonberg, K. Ramchandran, On compressing encrypted data, *IEEE Transactions on Signal Processing* 52 (2004) 2992–3006.
- [7] W. Liu, W. Zeng, L. Dong, Q. Yao, Efficient compression of encrypted grayscale images, *IEEE Transactions on Image Processing* 19 (2010) 1097–1102.
- [8] S. Lian, Z. Liu, Z. Ren, H. Wang, Commutative encryption and watermarking in video compression, *IEEE Transactions on Circuits and Systems for Video Technology* 17 (2007) 774–778.
- [9] X. Gao, C. Deng, X. Li, D. Tao, Geometric distortion insensitive image watermarking in affine covariant regions, *IEEE Transactions on Systems, Man, and Cybernetics-Part C* 40 (2010) 278–286.
- [10] Z. Ni, Y.-. Shi, N. Ansari, W. Su, Reversible data hiding, *IEEE Transactions on Circuits and Systems for Video Technology* 16 (2006) 354–362.
- [11] L. An, X. Gao, X. Li, D. Tao, C. Deng, J. Li, Robust reversible watermarking via clustering and enhanced pixel-wise masking, *IEEE Transactions on Image Processing* 21 (2012) 3598–3611.
- [12] X. Li, W. Zhang, X. Gui, B. Yang, Efficient reversible data hiding based on multiple histograms modification, *IEEE Transactions on Information Forensics and Security* 10 (2015) 2016–2027.
- [13] C. Deng, X. Gao, X. Li, D. Tao, A local tchebichef moments-based robust image watermarking, *Signal Processing* 89 (2009) 1531–1539.
- [14] X. Gao, C. Deng, X. Li, D. Tao, Geometric distortion insensitive image watermarking in affine covariant regions, *IEEE Transactions on Systems, Man, and Cybernetics, Part C* 40 (2010) 278–286.
- [15] Z. Qian., X. Zhang, Lossless data hiding in JPEG bitstream, *The Journal of Systems and Software* 85 (2012) 309–313.
- [16] S.-W. Jung, Lossless embedding of depth hints in JPEG compressed color images, *Signal Processing* 122 (2016) 39–51.

- [17] J.-C. Chang, Y.-Z. Lu, H.-L. Wu, A separable reversible data hiding scheme for encrypted JPEG bitstreams, *Signal Processing* 133 (2017) 135–143.
- [18] W. Puech, M. Chaumont, O. Strauss, A reversible data hiding method for encrypted images, in: *Proc. SPIE 6819, Security, Forensics, Steganography, and Watermarking of Multimedia Contents, Vol. X-68191E*, San Jose, CA, USA, 2008.
- [19] X. Zhang, Reversible data hiding in encrypted image, *IEEE Signal Processing Letters* 18 (2011) 255–258.
- [20] W. Hong, T.-S. Chen, H.-Y. Wu, An improved reversible data hiding in encrypted images using side match, *IEEE Signal Processing Letters* 19 (2012) 199–202.
- [21] X. Zhang, Separable reversible data hiding in encrypted image, *IEEE Transactions on Information Forensics and Security* 7 (2012) 826–832.
- [22] K. Ma, W. Zhang, X. Shao, N. Yu, F. Li, Reversible data hiding in encrypted images by reserving room before encryption, *IEEE Transactions on Information Forensics and Security* 8 (2013) 553–562.
- [23] S. Yi, Y. Zhou, Binary-block embedding for reversible data hiding in encrypted images, *Signal Processing* 133 (2017) 40–51.
- [24] C. Qin, X. Zhang, Effective reversible data hiding in encrypted image with privacy protection for image content, *Journal of Visual Communications and Image Representation* 31 (2015) 154–164.
- [25] D. Xu, R. Wang, Separable and error-free reversible data hiding in encrypted images, *Signal Processing* 123 (2016) 9–21.
- [26] X. Zhang, Z. Qian, G. Feng, Y. Ren, Efficient reversible data hiding in encrypted images, *Journal of Visual Communications and Image Representation* 25 (2014) 322–328.
- [27] R. G. Gallager, Low-density parity-check codes, Ph.D. thesis, Massachusetts Institute of Technology (July 1963).
- [28] Z. Qian, X. Zhang, Reversible data hiding in encrypted image with distributed source encoding, *IEEE Transactions on Circuits and Systems for Video Technology* 26 (2016) 636–646.
- [29] X. Wu, W. Sun, High-capacity reversible data hiding in encrypted images by prediction error, *Signal Processing* 104 (2014) 387–400.
- [30] Y.-C. Chen, C.-W. Shiu, G. Horng, Encrypted signal-based reversible data hiding with public key cryptosystem, *Journal of Visual Communications and Image Representation* 25 (2014) 1164–1170.
- [31] X. Zhang, J. Long, Z. Wang, H. Cheng, Lossless and reversible data hiding in encrypted images with public key cryptography, *IEEE Transactions on Circuits and Systems for Video Technology* 26 (2016) 1622–1631.
- [32] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, Y. Y. Tang, Secure reversible image data hiding over encrypted domain via key modulation, *IEEE Transactions on Circuits and Systems for Video Technology* 26 (2016) 441–452.
- [33] Z. Yin, A. Abel, X. Zhang, B. Luo, Reversible data hiding in encrypted image based on block histogram shifting, in: *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Shanghai, China, 2016, pp. 2129–2133.
- [34] W. Zhang, K. Ma, N. Yu, Reversibility improved data hiding in encrypted images, *Signal Processing* 94 (2014) 118–127.
- [35] X. Cao, L. Du, X. Wei, D. Meng, X. Guo, High capacity reversible data hiding in encrypted images by patch-level sparse representation, *IEEE Transactions on Cybernetics* 46 (2016) 1132–1143.
- [36] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, Security and privacy for storage and computation in cloud computing, *Information Sciences* 258 (2014) 371–386.
- [37] M. Ali, S. U. Khan, A. V. Vasilakos, Security in cloud computing: Opportunities and challenges, *Information Sciences* 305 (2015) 357–383.

- [38] D. Taubman, High performance scalable image compression with EBCOT, *IEEE Transactions on Image Processing* 9 (2000) 1158–1170.
- [39] X. Wu, N. Memon, Context-based, adaptive, lossless image coding, *IEEE Transactions on Communications* 45 (1997) 437–444.
- [40] M. J. Weinberger, G. Seroussi, G. Sapiro, The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS, *IEEE Transactions on Image Processing* 9 (2000) 1309–1324.
- [41] A. Skodras, C. Christopoulos, T. Ebrahimi, The JPEG 2000 still image compression standard, *IEEE Signal Processing Magazine* 18 (2001) 36–58.
- [42] W. Zhang, Z. Jiang, Z. Gao, Y. Liu, An efficient VLSI architecture for lifting-based discrete wavelet transform, *IEEE Transactions on Circuits and Systems II* 59 (2012) 158–192.
- [43] A. R. Calderbank, I. Daubechies, W. Sweldens, B. L. Yeo, Wavelet transforms that map integers to integers, *Journal of Applied and Computational Harmonic Analysis* 5 (1998) 332–369.
- [44] M. Unser, T. Blu, Mathematical properties of the jpeg2000 wavelet filters, *IEEE Transactions on Image Processing* 12 (2003) 1080–1090.
- [45] E. Biham, P. C. Kocher, Fast Software Encryption, *Lecture notes in Computer Science*, Springer, 1995, Ch. A known plaintext attack on the PKZIP stream cipher, pp. 144–153.
- [46] A. Biryukov, E. Kushilevitz, *Advances in cryptology*, *Lecture notes in Computer Science*, Springer, 1998, Ch. From differential cryptanalysis to ciphertext-only attacks, pp. 72–88.
- [47] A. Biryukov, *Encyclopedia of Cryptography and Security*, Springer, 2011, Ch. Chosen Plaintext Attack, pp. 205–206.
- [48] J. Daemen, V. Rijmen, *The Design of Rijndael, AES - The Advanced Encryption Standard*, Springer-Verlag, 2002.
- [49] S. Kim, N. I. Cho, Hierarchical prediction and context adaptive coding for lossless color image compression, *IEEE Transactions on Image Processing* 23 (2014) 445–449.
- [50] Y. Liu, O. Deforges, K. Samrouth, LAR-LLC: a low complexity multiresolution lossless image codec, *IEEE Transactions on Circuits and Systems for Video Technology* 26 (2016) 1490–1501.
- [51] D. A. Clunie, Lossless compression of grayscale medical images: effectiveness of traditional and state-of-the-art approaches, in: *Proc. SPIE 3980, Medical Imaging*, Vol. 12-386389, San Diego, CA, USA, 2000.
- [52] J. Taquet, C. Labit, Hierarchical oriented predictions for resolution scalable lossless and near-lossless compression of CT and MRI biomedical images, *IEEE Transactions on Image Processing* 21 (2012) 2641–2652.
- [53] F. Khelifi, On the security of a stream cipher in reversible data hiding schemes operating in the encrypted domain, *Signal Processing* 143 (2018) 336–345.
- [54] N. T. Courtois, J. Pieprzyk, *Advances in Cryptology*, *Lecture notes in Computer Science*, Springer, 2002, Ch. Cryptanalysis of Block Ciphers with Overdefined Systems of Equations, pp. 267–287.
- [55] C.-W. Lim, K. Khoo, *Fast Software Encryption*, *Lecture notes in Computer Science*, Springer, 2007, Ch. An Analysis of XSL Applied to BES, pp. 242–253.