Reversible Data Hiding over Encrypted Images via Preprocessing-Free Matrix Secret Sharing

Zhongyun Hua, Xingyu Liu, Yifeng Zheng, Shuang Yi, and Yushu Zhang

***Abstract*—Cloud service is a natural choice to store and manage the exponentially produced images. Data privacy is one of the most concerned points in cloud-based image services. Reversible data hiding over encrypted images (RDH-EI) is an effective technique to securely store and manage confidential images in the cloud. However, existing RDH-EI schemes have obvious weaknesses such as reliable key management system dependence and single point of failure. To securely store and manage confidential images in the cloud, in this study, we propose a new reversible data hiding strategy via image secret sharing. We first design a secure (r, n)-threshold preprocessing-free matrix secret sharing (PFMSS) technique. It can directly share m-bit data by matrix multiplication without preprocessing. Using the PFMSS, we further design a secure (r, n)-threshold reversible data hiding scheme over encrypted images. The content owner divides a confidential image into n shares without accessing to a secret encryption key, and then sends the n shares to n cloud-based image servers from competing providers. For each share, some additional data, e.g., integrity and identification of the image, can be embedded into it and these data can also be losslessly extracted. An authorized receiver can recover the confidential image from r shares. By designing, the content owner doesn't need to access a secret key when encrypting the image and the scheme can withstand $n - r$ points of failure. Simulation results show that our scheme can ensure image content confidentiality and has a much larger embedding capacity compared to state-of-the-art schemes.**

***Index Terms*—Reversible data hiding, encrypted image, multiple data hiders, secret sharing**

I. Introduction

With the fast development of information technology, a greatly large number of images are exponentially produced everyday by different kinds of imaging devices such as smartphones, digital cameras, and various IoT devices. The tremendous growth of images causes heavy pressure for image storage and management and cloud service is a natural choice to store and manage the exponentially produced images [1]. When images are stored in the cloud, cloud server may embed some additional data into the images, e.g., time stamps, remarks and copyright data, for authentication management, copyright protection, and so on. Some reversible data hiding (RDH) schemes were developed using lossless compression [2], histogram shifting [3], [4], difference expansion [5], or prediction-error expansion [6]. Especially, deep learning shows powerful ability in RDH, since it can help find the optimal embedding positions in an image [7], [8].

Data privacy is a concerned point in cloud-based image services, since many image owners are unwilling to share their image content to cloud servers or any other authorized one [9]. Recently, many researches focus on reversible data hiding over encrypted images (RDH-EI), which is an effective technique to securely store and manage confidential images in cloud servers [10]–[12]. In an RDH-EI scheme, the content owner encrypts the confidential image into encrypted image to protect its confidentiality. The cloud server can embed additional data over the encrypted image for the purpose of management, identification, and etc., without accessing the image content, while the data embedding operation is reversible. Thereafter, the authorized receiver can completely recover the original image.

The existing RDH-EI schemes can be divided into three categories based on the embedding room vacation technique: reserving room before encryption (RRBE) [13]–[15], vacating redundancy in encryption (VRIE) [16]–[18], and vacating room after encryption (VRAE) [19]–[21]. For the RRBE-based schemes, the image owner first compresses an original image to vacate embedding room, and then encrypts the compressed image using any existing encryption algorithm. These schemes can achieve a high embedding capacity [14], [22], while the encrypted results have high security level. However, since room reservation is performed in the original image, it may cause heavy computation cost to the image owner. For the VRIE-based schemes, the image owner encrypts an original image using special public key cryptosystems (e.g., learning with errors-based cryptography). These cryptosystems can bring some data redundancy during the encryption such that some modifications to the encrypted image cannot affect the correctness of decryption [23]. Thus, the data hider can embed data by slightly modifying the encrypted image. The encrypted image is highly secure. But these cryptosystems cause large data expansion and the size of the encrypted image is much larger than that of the original image, as discussed in [23].

This work was supported by the National Natural Science Foundation of China under Grants 62071142 and 62002301, by the Guangdong Basic and Applied Basic Research Foundation under Grants 2021A1515110027, 2021A1515011406 and 2023A1515010714, by the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies under Grant 2022B1212010005, and by the Shenzhen Science and Technology Program (Grant No.ZDSYS20210623091809029). (Corresponding author: Shuang Yi.)

Zhongyun Hua is with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Guangdong 518055, China, and also with the Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, Shenzhen 518055, China (e-mail: huazyum@gmal.com).

Xingyu Liu and Yifeng Zheng are with School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Guangdong 518055, China (e-mail: liuxyuh@gmail.com; yifeng.zheng@hit.edu.cn).

Shuang Yi is with Engineering Research Center of Forensic Science, Chongqing Education Committee, College of Criminal Investigation, Southwest University of Political Science and Law, Chongqing 401120, China (e-mail: yishuang@swupl.edu.cn).

Yushu Zhang is with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, Jiangsu 210016, China (e-mail: yushu@nuaa.edu.cn).

For the VRAE-based schemes, the image owner uses some special encryption methods to retain some data redundancy of the original image in the encrypted image and the data hider uses the preserved redundancy to vacate room for data hiding [24], [25]. This strategy vacates embedding room in the encrypted image and can greatly reduce the computation cost to the image owner. It is suitable for cloud services, where cloud server acts as the data hider that has large computation resource. But special encryption techniques are required to preserve data redundancy and ensure image confidentiality.

In this study, we propose a novel RDH-EI scheme using image secret sharing. We first develop an (r, n)-threshold preprocessing-free matrix secret sharing (PFMSS) technique. In contrast to the existing secret sharing techniques used in [26], [27] that share image pixels in prime finite fields, PFMSS can directly share m-bit pixels without any preprocessing. Using the PFMSS, we further propose an RDH-EI scheme called PFMSS-RDHEI, in which a novel most significant bit (MSB) prediction method, called error class and value encoding (ECVE), is developed to encode image pixels in the encrypted domain. Compared to the existing MSB prediction methods introduced in [13], [14], the developed ECVE can separate more compressible significant bits for the same prediction error range, achieving a large embedding room. The PFMSS-RDHEI allows the content owner to encrypt its image into n shares for n cloud servers. For each share, some additional data can be embedded into it and the embedding operation is reversible. An authorized receiver can recover the original image from r shares. The contributions and novelty of this study are concluded as follows:

- We present the first RDH-EI scheme that securely stores and manages confidential images without a reliable key management system (KMS). The content owner can encrypt an image into n encrypted shares without a secret key involved, while an authorized receiver can losslessly recover the image with at least r shares. Thus, our scheme can withstand $n - r$ points of failure, and the collusion attack of $r - 1$ shares as well.
- We develop an (r, n)-threshold image secret sharing method called PFMSS, which can directly encrypt m-bit pixels without any preprocessing. Formal analysis is provided to prove its correctness and justify its security.
- We propose a novel encoding method ECVE that can fully separate the compressible bits from image pixels, and thus can vacate a large room for data embedding.
- Extensive evaluation results show that our RDH-EI scheme can protect the image content confidentiality while achieving a superior embedding capacity in contrast to the state-of-the-art RDH-EI schemes.

Our scheme is suitable for many scenarios, for instance, the remote medical consultation, which is shown in Fig. 1. The IoT device generates medical images which are to be outsourced to the cloud for storage and management. For privacy protection, each medical image is first encrypted into n shares by the IoT gateway [28], such as a medical center, and then outsourced to n cloud servers from competitive server providers. Each cloud server can embed some additional data into the image share, e.g., time stamps, remarks and copyright data, for authentication management, copyright protection, and so on. The authorized hospital, doctor or patient can receive r image shares to recover the medical image. Since the n cloud servers are from different competitive providers, they are unwilling to have the collusion operations. Actually, our scheme can resist the collusion attack of $r - 1$ cloud servers.

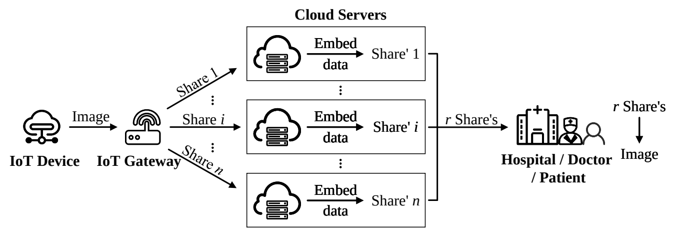


Fig. 1: Secure storage and management of medical images in remote medical consultation.

The remainder of this paper is organized as follows. Section II reviews the related work of the MSB prediction techniques and RDH-EI schemes. Section III introduces the PFMSS. Section IV presents our RDH-EI scheme. Section V discusses the property of our RDH-EI scheme and presents the experimental results, and Section VI evaluates its performances and compares it with the state-of-the-art schemes. Finally, Section VII concludes the paper.

II. Related Work

In this section, we introduce the MSB prediction techniques for data embedding, and then review the RDH-EI schemes.

A. MSB Prediction Techniques

A natural image usually has high pixel redundancy, and one can make use of these pixel redundancy to compress the image or vacate some room for hiding data. The MSB prediction technique is an effective method to utilize the pixel redundancy by predicting the MSBs of a pixel using its neighbor pixels. Then the MSBs can be vacated for data embedding. Puteaux $et\ al.$ [12] developed a MSB prediction method by recovering the MSB of a pixel using its prediction value. This method can embed one bit into each pixel at most, but may suffer from recovery failure since there is no mechanism to guarantee the correctness of flag blocks. Besides, its maximum embedding rate is only 1 bpp. To improve the embedding capacity, Chen $et\ al.$ [29] proposed an iterative prediction method on bit planes. Then the significant bit planes have large proportion of '0', which is beneficial for embedding room vacation. Yin $et\ al.$ [30] proposed a new MSB prediction method that vacates embedding room by calculating the number of the same bits between a pixel and its prediction value. The pixel can be recovered losslessly using the number of the same bits.

Most of the MSB prediction methods introduced above vacate embedding room from the experience aspect. To exploit pixel redundancy sufficiently, some works were devoted to theoretically calculate the maximum number of significant bits that can be vacated for data embedding [13], [14].

Puteaux $et\ al.$ [12] proved that the MSB of a pixel can be vacated for data embedding using bit replacement if the absolute value of prediction error is less than 64. Inspired by this conclusion, Mohammadi $et\ al.$ [13] embedded data into multiple significant bit planes of an image and then recovered the bit planes using more accurate prediction error ranges. Later, Yu $et\ al.$ [14] enhanced the embedding capacity by designing a new prediction error division. Compared with the MSB prediction methods in [13], [14], our proposed ECVE can separate more embeddable bits within the same prediction error, and thus can achieve larger embedding capacity.

B. RDH-EI Schemes

Existing RDH-EI schemes can be divided into RRBE-based schemes [13]–[15], VRIE-based schemes [16]–[18] and VRAE-based schemes [19]–[21].

1) RRBE-Based RDH-EI Schemes: In the RRBE-based schemes [14], [22], the content owner first reserves embedding room in the plain image by image compression, and then encrypts the compressed image using an existing stream cipher. The data hider can directly embed some secret data into the reserved room, and the receiver with related keys can recover the original image and embedded data. Thus, the key point of an RRBE-based scheme is how to reserve embedding room in the plain image. Recently, the main trend of RRBE-based RDH-EI schemes is to reserve large embedding room by compressing the prediction errors of plain images. For example, the authors of [22] and [31] adopted the median edge detector (MED) predictor to predict a plain image, and then separately utilized the bit-plane encoding method in [10] and the pixel encoding method in [32] to encode the prediction errors, achieving a large embedding capacity. Qiu $et\ al.$ [33] calculated the prediction errors of pixels and then compressed the pixels whose prediction errors are within the range of $[-T, T]$, while preserving the original values of the other pixels. However, the threshold T is determined by exhaustive enumeration, which is time-consuming. The RRBE-based RDH-EI schemes can achieve a high embedding capacity, because the high pixel redundancy of plain images can be fully exploited. However, these embedding capacity reserving operations are usually very complex and time-consuming, and cause heavy computation costs to the content owners, especially for some performance-limited terminal devices.

2) VRIE-Based RDH-EI Schemes: In the VRIE-based schemes [16], [17], the content owner uses special public key encryption strategies to encrypt an original image and these encryption strategies can bring data redundancy such that some modifications to the encrypted image cannot affect the correctness of decryption. The data hider can embed data by slightly modifying the encrypted image. Ke $et\ al.$ [23] proposed such a scheme by encrypting an original image using learning with errors-based cryptography. The data hider first determines the modification range that cannot affect the correctness of decryption and divides the range into several sub-ranges. Then, the data hider can embed different data to the encrypted image by adding the corresponding modifications to it. Wang $et\ al.$ [17] proposed a VRIE-based scheme using McEliece cryptography. The data hider can embed data into the encrypted image by slightly changing the encrypted image, which cannot affect the image decryption. As the used cryptosystems are secure, the image content in these VRIE-based schemes can be well protected. However, these cryptosystems cause large data expansion and the size of the encrypted image is much larger than that of the original image [23].

3) VRAE-Based RDH-EI Schemes: The VRAE-based schemes vacate embedding room in the encrypted image by data hider [24], [25]. These schemes are more suitable for cloud platforms because the cloud servers act as data hiders and they usually have high performance. To retain data redundancy in the encrypted domain for data embedding, most VRAE-based schemes encrypt an image using lightweight symmetric encryption methods, such as bitwise XOR [34] and block-based or in-block permutation [32], [35], [36]. However, these VRAE-based RDH-EI schemes using only bitwise XOR as encryption method are frail under the ciphertext-only attack [37] while the block permutation and co-modulation encryption has a low ability to resist the known-plaintext attack [38]. To obtain a higher security level, some RDH-EI schemes [39], [40] encrypt images using some public key encryption strategies and the secret data are embedded using the homomorphic property between the plain images and encrypted images. For example, the authors in [39] encrypted plain images using the Paillier and EIGamal cryptosystems, and then embeded secret data into the encrypted images using the additive and multiplicative homomorphisms for modular multiplication, respectively. However, these encryption strategies usually suffer from high computation cost and large data expansion [40].

Recently, several RDH-EI schemes have been developed using secret sharing techniques [26], [27], [41]. Some of them can resist single point of failure by encrypting the confidential image to be multiple shares for decentralized storage. However, these schemes only partly solve the above issues and have the following properties. (1) Most of their used secret sharing techniques process image pixels in prime finite fields so that they cannot be directly applied to m-bit image pixels [26], [27]; (2) Each pixel is encrypted independently without involving any random number in the encryption, thus only achieving limited security [26]; (3) Some schemes don't consider the data redundancy in the encrypted domain well enough and achieve a small embedding capacity [26], [41].

C. Discussions

From the analysis above, the existing MSB prediction techniques cannot sufficiently separate all the compressible significant bits from image pixels. For the RDH-EI techniques, the RRBE-based schemes may cause heavy computation costs to the content owner, the VRIE-based schemes may cause large data expansion, and the VRAE-based schemes may have security drawbacks, and also large data expansion. Besides, all the existing RDH-EI schemes require a reliable key management system (KMS) [42] to manage the secret keys. However, it is practically costly to implement a reliable KMS in such a

TABLE I: Important notations in this paper.

Notation	Description
$F(y)$	A prime polynomial in $GF(2^8)$
$\mathbf{X}$	A matrix of size $n \times r$ for sharing
$\mathbf{h}$	A vector $[h_1, h_2, \cdots, h_r]^T$ to be shared
h_1	The secret value s
$\mathbf{h}'$	A random integer vector $[h_2, \cdots, h_r]^T$
$\mathbf{s}$	Sharing results with n elements
$\mathbf{X}_r$	Any $r \times r$ sub-matrix of $\mathbf{X}$
$\mathbf{s}_r$	Sharing results with r elements
$f(x)$	A degree-$(r-1)$ polynomial passing through the point $(0, s)$
$\mathbf{I}$	The original 8-bit image with size $M \times N$
$\mathbf{I}(j)$	The j-th pixel of image $\mathbf{I}$
$\mathbf{B}$	A block of image $\mathbf{I}$ with size $S_1 \times S_2$
$\mathbf{B}(j)$	The j-th pixel of block $\mathbf{B}$
$\mathbf{EI}_i$	The i-th encrypted image
$p(i,j)$	The pixel at position (i,j) in an image or a block
$\hat{p}(i,j)$	The prediction value of $p(i,j)$
$e(i,j)$	The prediction error of $p(i,j)$
p_L	The incompressible LSBs of a pixel
p_M	The compressible MSBs of a pixel
$\mathbf{L}$	The total encoded result of an encrypted image except for each block's first pixel
$\mathbf{EM}_i$	The i-th marked encrypted image
$\otimes$	Polynomial multiplication
$\oplus$	Bitwise XOR operation

multiparty environment over insecure public networks [42]. In this study, we aim to propose a new secret sharing-based RDH-EI scheme that can solve the above issues. The important notations used in this paper are listed in Table I.

III. PFMSS

Herein, we present an (r, n)-threshold PFMSS technique ($2 \le r \le n$) and analyze its correctness and security.

A. (r, n)-Threshold PFMSS

A polynomial with integer coefficients that cannot be factorized into polynomials of lower degree with integer coefficients is called a prime polynomial, for example, the $y^2 + y + 1$. The algebraic operations in the finite field can be extended to polynomial operations. Following the polynomial operations in $GF(2^m)$, an $n \times r$ coefficient matrix in the (r, n)-threshold PFMSS can be constructed.

Algorithm 1 presents the pseudocode for generating the coefficient matrix using n random integers generated by a pseudo-random number generator (PRNG). Note that $\oplus$ indicates bitwise XOR operation and $\otimes$ indicates polynomial multiplication. After constructing an $n \times r$ coefficient matrix $\mathbf{X}$, the sharing process can be expressed as

$$\mathbf{s} = \mathbf{X} \otimes \mathbf{h} \mod F(y), \tag{1}$$

where $F(y)$ is a prime polynomial in $GF(2^m)$, $\mathbf{h} = [h_1, h_2, \cdots, h_r]^T$, where h_1 is the secret to be shared, and $h_2, h_3, \cdots, h_r$ are $r-1$ random numbers in $GF(2^m)$.

B. Correctness

The correctness of a secret sharing scheme indicates that the secret can be recovered from any r shares with their identities [43]. Because the sharing operation in Eq. (1) is performed in the finite field using matrix multiplication, the secret can be obtained by the matrix inverse operation. Then we first present Theorem 1 [44] to indicate the sufficient condition for the matrix inverse operation using polynomial operations in the finite field.

```
Algorithm 1. Generation of the coefficient matrix.
Input: Random integer sequence $\mathbf{b} = \{b_k\}_{k=1}^{n}$, where $\{b_k\}$
    are $n$ different elements and $0 < b_k < 2^m$.
1: Initialize an $n \times r$ matrix $\mathbf{X} \in \mathbb{R}^{n \times r}$.
2: Set the elements in the first column of $\mathbf{X}$ as 1.
3: for $j = 2$ to $r$ do
4:     for $i = 1$ to $n$ do
5:         $x(i,j) = x(i,j-1) \otimes (b_i \oplus (j-2))$.
6:     end for
7: end for
Output: Coefficient matrix $\mathbf{X}$.
```

Theorem 1. *For an $r \times r$ square matrix $\mathbf{X}_r$ and a prime polynomial $F(y)$ in $GF(2^m)$, the polynomial matrix operation*

$$\mathbf{s}_r = \mathbf{X}_r \otimes \mathbf{h} \mod F(y)$$

is reversible. Furthermore, its inverse operation is

$$\mathbf{h} = \mathbf{X}_r^{-1} \otimes \mathbf{s}_r \mod F(y),$$

if $det(\mathbf{X}_r)$ and $F(y)$ are two coprime polynomials.

Theorem 1 is a basic theorem of modern cryptography theory and gives the sufficient condition of matrix inverse operation in a finite field. Next, we propose Lemma 1 stating that the determinant of any $r \times r$ sub-matrix of the coefficient matrix generated by our proposed Algorithm 1 and the prime polynomial $F(y)$ are two coprime polynomials.

Lemma 1. *For the $n \times r$ matrix generated by Algorithm 1, the determinant of its any $r \times r$ sub-matrix is coprime with the prime polynomial $F(y)$.*

Proof: Assume that the i_1-th, i_2-th, $\cdots$, i_r-th rows are selected from the $n \times r$ matrix $\mathbf{X}$ generated by Algorithm 1. Then, these r rows can form an $r \times r$ sub-matrix $\mathbf{X}_r$, which is shown as

$$\mathbf{X}_r = \begin{bmatrix} 1 & b_{i_1} & \cdots & b_{i_1} \otimes (b_{i_1} \oplus 1) \otimes \cdots \otimes (b_{i_1} \oplus (r-2)) \\ 1 & b_{i_2} & \cdots & b_{i_2} \otimes (b_{i_2} \oplus 1) \otimes \cdots \otimes (b_{i_2} \oplus (r-2)) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & b_{i_r} & \cdots & b_{i_r} \otimes (b_{i_r} \oplus 1) \otimes \cdots \otimes (b_{i_r} \oplus (r-2)) \end{bmatrix}. \tag{2}$$

Now we calculate the determinant of $\mathbf{X}_r$ using the following steps.

Subtraction. For the j-th ($2 \le j \le r$) column, we update it by subtracting the multiplication of its previous column with $b_{i_1} \oplus (j-2)$. Take the element $\mathbf{X}_r(h, j)$ as an example. From Eq. (2), $\mathbf{X}_r(h, j) = \mathbf{X}_r(h, j-1) \otimes (b_{i_h} \oplus (j-2))$. Note that the subtraction and addition operations in $GF(2^m)$ are both XOR operation $\oplus$. After subtracting the multiplication of $\mathbf{X}_r(h, j-1)$ with $b_{i_1} \oplus (j-2)$, we can update $\mathbf{X}_r(h, j)$ as

$$\begin{aligned}
&\mathbf{X}_r(h, j) \\
&= \mathbf{X}_r(h, j-1) \otimes (b_{i_h} \oplus (j-2)) \oplus \mathbf{X}_r(h, j-1) \otimes (b_{i_1} \oplus (j-2)) \\
&= \mathbf{X}_r(h, j-1) \otimes ((b_{i_h} \oplus (j-2)) \oplus (b_{i_1} \oplus (j-2))) \\
&= \mathbf{X}_r(h, j-1) \otimes (b_{i_h} \oplus b_{i_1}) \\
&= \mathbf{X}_r(h, j-2) \otimes (b_{i_h} \oplus (j-3)) \otimes (b_{i_h} \oplus b_{i_1}) \\
&= \mathbf{X}_r(h, 1) \otimes (b_{i_h} \oplus 0) \otimes \cdots \otimes (b_{i_h} \oplus (j-3)) \otimes (b_{i_h} \oplus b_{i_1}) \\
&= b_{i_h} \otimes (b_{i_h} \oplus 1) \otimes \cdots \otimes (b_{i_h} \oplus (j-3)) \otimes (b_{i_h} \oplus b_{i_1}).
\end{aligned} \tag{3}$$

Obviously, $\mathbf{X}_r(1, j) = 0$ $(2 \leq j \leq r)$. Besides, $\mathbf{X}_r(h, 2) = \mathbf{X}_r(h, 1) \otimes (b_{i_h} \oplus b_{i_1}) = b_{i_h} \oplus b_{i_1}$ $(1 \leq h \leq r)$. Since the linear transformations of a matrix do not change its determinant, we can obtain the determinant of $\mathbf{X}_r$ as

$$\begin{aligned}
&det(\mathbf{X}_r) \\
&= \begin{vmatrix}
1 & 0 & \cdots & 0 \\
1 & b_{i_2} \oplus b_{i_1} & \cdots & b_{i_2} \otimes \cdots \otimes (b_{i_2} \oplus (r-3)) \otimes (b_{i_2} \oplus b_{i_1}) \\
\vdots & \vdots & \ddots & \vdots \\
1 & b_{i_r} \oplus b_{i_1} & \cdots & b_{i_r} \otimes \cdots \otimes (b_{i_r} \oplus (r-3)) \otimes (b_{i_r} \oplus b_{i_1})
\end{vmatrix}.
\end{aligned} \tag{4}$$

Expansion. Expanding the above determinant along the first row, we can obtain that

$$\begin{aligned}
&det(\mathbf{X}_r) \\
&= \begin{vmatrix}
b_{i_2} \oplus b_{i_1} & \cdots & b_{i_2} \otimes \cdots \otimes (b_{i_2} \oplus (r-3)) \otimes (b_{i_2} \oplus b_{i_1}) \\
b_{i_3} \oplus b_{i_1} & \cdots & b_{i_3} \otimes \cdots \otimes (b_{i_3} \oplus (r-3)) \otimes (b_{i_3} \oplus b_{i_1}) \\
\vdots & \vdots & \vdots \\
b_{i_r} \oplus b_{i_1} & \cdots & b_{i_r} \otimes \cdots \otimes (b_{i_r} \oplus (r-3)) \otimes (b_{i_r} \oplus b_{i_1})
\end{vmatrix}.
\end{aligned} \tag{5}$$

Factor extraction. Observe that all entries in the h-th $(1 \leq h \leq r-1)$ row have a factor of $b_{i_{h+1}} \oplus b_{i_1}$. After extracting the factor for each row, one can then obtain

$$\begin{aligned}
det(\mathbf{X}_r) = &\prod_{2 \leq k \leq r} (b_{i_k} \oplus b_{i_1}) \otimes \\
&\begin{vmatrix}
1 & b_{i_2} & \cdots & b_{i_2} \otimes (b_{i_2} \oplus 1) \otimes \cdots \otimes (b_{i_2} \oplus (r-3)) \\
1 & b_{i_3} & \cdots & b_{i_3} \otimes (b_{i_3} \oplus 1) \otimes \cdots \otimes (b_{i_3} \oplus (r-3)) \\
\vdots & \vdots & \ddots & \vdots \\
1 & b_{i_r} & \cdots & b_{i_r} \otimes (b_{i_r} \oplus 1) \otimes \cdots \otimes (b_{i_r} \oplus (r-3))
\end{vmatrix}.
\end{aligned} \tag{6}$$

Repeating the above subtraction, expansion, and factor extraction steps. Note that the multiplier in the subtraction step is always $b_{i_m} \oplus (j-2)$, where m is 1 in the first round, 2 in the second round, and so on. After $r-2$ rounds, one can eventually obtain

$$\begin{aligned}
det(\mathbf{X}_r) &= \prod_{2 \leq k \leq r} (b_{i_k} \oplus b_{i_1}) \otimes \cdots \otimes \prod_{r-1 \leq k \leq r} (b_{i_k} \oplus b_{i_{r-2}}) \\
&\quad \otimes \begin{vmatrix} 1 & b_{i_{r-1}} \\ 1 & b_{i_r} \end{vmatrix}, \\
&= \prod_{1 \leq j < k \leq r} (b_{i_k} \oplus b_{i_j}).
\end{aligned} \tag{7}$$

According to Algorithm 1, all the parameters satisfy $0 < b_k < 2^m, (k = 1, 2, \cdots, n)$, where all b_ks are different from each other. Because $F(y)$ is a prime polynomial in $GF(2^m)$, all factors of $det(\mathbf{X}_r)$ are coprime with $F(y)$. Thus, $det(\mathbf{X}_r)$ and $F(y)$ are coprime polynomials. Because the i_1-th, i_2-th,$\cdots$, i_r-th rows are any r rows of $\mathbf{X}$, the determinant of any $r \times r$ sub-matrix is coprime with $F(y)$. ■

Thus, $\mathbf{s}_r = \mathbf{X}_r \otimes \mathbf{h} \mod F(y)$ is reversible, and $\mathbf{h}$ can be completely recovered via

$$\mathbf{h} = \mathbf{X}_r^{-1} \otimes \mathbf{s}_r \mod F(y), \tag{8}$$

where $\mathbf{s}_r = [s_{i_1}, \cdots, s_{i_r}]^T$.

C. Security

First, we present Definition 1 [45] to define the security of a secret sharing scheme. The security of a secret sharing scheme indicates that any t $(t < r)$ shares with their identities cannot reveal anything about the secret [46].

Definition 1. *An (r, n)-threshold sharing scheme (Share, Reconstruct) over $GF(2^m)$ is perfectly secure if: $\forall x, x' \in GF(2^m), \forall S \subseteq \{1, \cdots, n\}$ s.t. $|S| < r$, and for every possible $|S|$-tuple of shares $\alpha = (\alpha_1, \cdots, \alpha_{|S|})$, we have*

$$\begin{aligned}
&\Pr_{Share(x) \to (s_1, \cdots, s_n)} [(s_i)_{i \in S} = \alpha], \\
&= \Pr_{Share(x') \to (s'_1, \cdots, s'_n)} [(s'_i)_{i \in S} = \alpha].
\end{aligned}$$

Definition 1 indicates that if the probability of obtaining any $|S|$ shares from any secret value is the same, then every unauthorized set S, i.e., $|S| < r$, can learn nothing about the secret and the (r, n)-threshold sharing scheme is secure.

Then, we construct a degree-$(r-1)$ polynomial $f(x)$ as follows

$$f(x) = s \oplus \sum_{k=2}^{r} (h_k \otimes (x \oplus 0) \otimes \cdots \otimes (x \oplus (k-2))) \mod F(y), \tag{9}$$

where s is a constant, and $h_2, \cdots, h_r$ are random numbers in $GF(2^m)$.

Corollary 1. *When every element of $\mathbf{h}' = [h_2, \cdots, h_r]^T$ in Eq. (9) is uniformly distributed, $f(x)$ is a random degree-$(r-1)$ polynomial passing through the point $(0, s)$.*

Proof: Suppose that $g(x)$ is a degree-$(r-1)$ polynomial passing through $(0, q_1)$ as follows

$$g(x) = q_1 \oplus (\sum_{k=2}^{r} (q_k \otimes x^{k-1})) \mod F(y). \tag{10}$$

When the coefficients of every degree in $f(x)$ and $g(x)$ are the same, $f(x) = g(x)$, namely,

$$\begin{aligned}
f(x) &= s \oplus \sum_{k=2}^{r} (h_k \otimes (x \oplus 0) \otimes \cdots \otimes (x \oplus (k-2))) \\
&= s \oplus (h_2 \otimes x) \oplus (h_3 \otimes x \otimes (x \oplus 1)) \\
&\quad \oplus \cdots \oplus (h_r \otimes x \otimes (x \oplus 1) \otimes \cdots \otimes (x \oplus (r-2))) \\
&= q_1 \oplus (q_2 \otimes x) \oplus \cdots \oplus (q_r \otimes x^{r-1}) = g(x).
\end{aligned} \tag{11}$$

Then, we can obtain the relationships between the coefficients $\mathbf{q} = [q_1, \cdots, q_r]^T$ and $\mathbf{h} = [s, h_2, \cdots, h_r]^T$. For example, $q_1 = s$ and

$$\begin{aligned}
q_2 = &h_2 \oplus (1 \otimes h_3) \oplus (1 \otimes 2 \otimes h_4) \\
&\oplus \cdots \oplus (1 \otimes 2 \otimes \cdots \otimes (r-2) \otimes h_r).
\end{aligned} \tag{12}$$

As a result, we can obtain the relationship between $\mathbf{q}$ and $\mathbf{h}$ as follows:

$$\mathbf{q} = \begin{bmatrix} 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & \prod_{i=1}^{r-3} i & \prod_{i=1}^{r-2} i \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & \sum_{i=1}^{r-2} i \\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \otimes \mathbf{h} \mod F(y). \tag{13}$$

The $r \times r$ matrix in the above equation is an upper triangular matrix, whose diagonal elements are equal to 1. Thus, its determinant is 1. Because $F(y)$ is a prime polynomial in $GF(2^m)$, 1 and $F(y)$ are coprime polynomials. Then according to Theorem 1, Eq. (13) is reversible. Thus, there is a one-to-one map between $\mathbf{h}$ and $\mathbf{q}$, and $q_1 = s$ is a constant. Set $\mathbf{q}' = [q_2, \cdots, q_r]^T$. Then, there is a one-to-one map between $\mathbf{h}'$ and $\mathbf{q}'$ as well.

Since every element in $\mathbf{h}' = [h_2, \cdots, h_r]^T$ is uniformly distributed, $\mathbf{q}' = [q_2, \cdots, q_r]^T$ is uniformly distributed in $GF(2^m)$. Thus, $g(x)$ is a random degree-$(r-1)$ polynomial passing through $(0, s)$. Since $f(x) = g(x)$, $f(x)$ is also a random degree-$(r-1)$ polynomial passing through the point $(0, s)$. ∎

Next, we propose Lemma 2 stating the sufficient condition to ensure the security of our PFMSS scheme over $GF(2^m)$.

Lemma 2. *The PFMSS scheme over $GF(2^m)$ is secure if every element of $\mathbf{h}' = [h_2, \cdots, h_r]^T$ is uniformly distributed in the $GF(2^m)$.*

Proof: Suppose that

$$\begin{bmatrix} 1 & b_1 & \cdots & b_1 \otimes \cdots \otimes (b_1 \oplus (r-2)) \\ 1 & b_2 & \cdots & b_2 \otimes \cdots \otimes (b_2 \oplus (r-2)) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & b_n & \cdots & b_n \otimes \cdots \otimes (b_n \oplus (r-2)) \end{bmatrix} \otimes \begin{bmatrix} s \\ h_2 \\ \vdots \\ h_r \end{bmatrix} = \begin{bmatrix} s_1 \\ s_2 \\ \vdots \\ s_n \end{bmatrix}, \tag{14}$$

which is equivalent to the degree-$(r-1)$ polynomial $f(x)$ described in Eq. (9) passing through $(n+1)$ points, $(0, s), (b_1, s_1), \cdots, (b_n, s_n)$. Because every element of $\mathbf{h}' = [h_2, \cdots, h_r]^T$ is uniformly distributed in $GF(2^m)$, $f(x)$ is a random degree-$(r-1)$ polynomial that passes through a fixed point $(0, s)$, according to Corollary 1.

The total number of degree-$(r-1)$ polynomials is $(2^m)^r$ in the finite field $GF(2^m)$. According to the Lagrange interpolation in $GF(2^m)$, the number of degree-$(r-1)$ polynomials that pass through any t different points is $(2^m)^{r-t}$ (Corollary 3.10 on page 57 of [47]).

The number of degree-$(r-1)$ polynomials $f(x)$ passing through the point $(0, s)$ is $(2^m)^{r-1}$. For $\forall S \subseteq \{1, \cdots, n\}$ s.t. $|S| < r$, the number of degree-$(r-1)$ polynomials $f(x)$ passing through $|S|+1$ different points $(0, s)$, $\{(b_i, s_i) | i \in S\}$ is $(2^m)^{r-|S|-1}$. Thus, for the $f(x)$ that passes through the point $(0, s)$, the probability of passing through the $|S|+1$ different points $(0, s)$, $\{(b_i, s_i) | i \in S\}$ is

$$\begin{aligned} Pr &= \frac{Pr[f(0) = s, f(b_i) = s_i (i \in S)]}{Pr[f(0) = s]} \\ &= \frac{\frac{(2^m)^{r-|S|-1}}{(2^m)^r}}{\frac{(2^m)^{r-1}}{(2^m)^r}} = (\frac{1}{2^m})^{|S|}, \end{aligned} \tag{15}$$

which denotes the probability of obtaining any $|S|$ shares from the secret s. Notice that the possibility is independent of s, and $\{b_i | i \in S\}$. Thus, the PFMSS scheme over $GF(2^m)$ is secure, according to Definition 1. ∎

D. Comparisons with Prior Secret Sharing Schemes

Recently, some secret image sharing schemes based on matrix have been developed and we compare our proposed PFMSS with these schemes from the aspects of extra processing, security, pixel expansion and recovery quality. Table II lists the comparison results. The schemes in [48], [49] share and recover image pixels using matrix multiplication. The scheme in [48] shares an image over Z_{257} finite field. When processing an 8-bit image, the encrypted image may exist pixels with value 256 and these pixels should be post-processed to avoid data size overflow. The scheme in [49] shares an image in the ring of integers modulo 256 (Z_{256}) using an $n \times r$ coefficient matrix K. The matrix is randomly generated until it satisfies two conditions: 1) Any r row vectors of K are linearly independent; 2) The determinant of any $r \times r$ submatrix of K is coprime with 256. These two conditions are necessary and sufficient to ensure that the secret value can be recovered losslessly with any r shares. However, the success of this method relies on experiment trying and has no theoretical foundation. Besides, the security of the sharing process hasn't been analyze and cannot be guaranteed.

The schemes in [50], [51] share an image into n shares by directly dividing it into n parts according to a generated matrix with elements 0 and 1. The binary matrix ensures that the whole secret image can be recovered losslessly with at least r shares. However, the schemes cause information leakage since each share contains partial pixels of the secret image. To protect image confidentiality, a secure stream cipher is required to encrypt the image [51].

The schemes in [52], [53] are visual secret sharing schemes. The scheme in [52] shares a binary image to n color shares. Since more memory units are required to store a pixel after sharing, the scheme suffers from pixel expansion. Besides, it is a lossy method, due to its bitwise-or operation in the recovering process. The scheme in [53] shares an 8-bit grayscale image to n shares, and offers two decoding options: the stacking-to-see decryption and lossless recovery. Since each share has larger file size than the original image, the scheme suffers from pixel expansion.

For our proposed PFMSS, it can directly encrypt m-bit data without extra processing, such as 8-bit grayscale image pixels. Its security has been proved in Section III-C. In addition, each share has the same size as the secret image with no pixel expansion. Thus, although some previous matrix-based secret sharing schemes are secure, our proposed PFMSS shows superiority in the RDH-EI applications.

TABLE II: Comparisons with different matrix-based (r, n)-threshold secret image sharing schemes.

Methods	No extra processing	Security guarantee	No pixel expansion	Lossless recovery
Hua *et al.* [48]	×	✓	✓	✓
Yu *et al.* [49]	✓	×	✓	✓
Chen *et al.* [50]	×	×	✓	✓
Bao *et al.* [51]	×	×	✓	✓
Liu *et al.* [52]	✓	✓	×	×
Wu *et al.* [53]	✓	✓	×	✓
Our PFMSS	✓	✓	✓	✓

E. Discussions

For an m-bit image $\mathbf{I}$, the (r, n)-threshold PFMSS for the j-th pixel $\mathbf{I}(j)$ can be expressed as

$$\mathbf{s}(j) = \mathbf{X} \otimes \mathbf{h} \mod F(y), \tag{16}$$

where $\mathbf{h} = [\mathbf{I}(j), h_2 \cdots, h_r]^T$ and $h_2, \cdots, h_r$ are $r-1$ random numbers. The sharing result $\mathbf{s}(j) = [s_1(j), s_2(j), \cdots, s_n(j)]^T$ includes the n shares of the j-th pixel. A party P_i holds the share $(i, s_i(j))$, where i is the identity.

When one collects r shares, he/she can recover the original pixel without any data loss. Suppose that the i_1-th, i_2-th, $\cdots$, i_r-th shares have been collected. First, an $n \times r$ matrix $\mathbf{X}$ is generated using Algorithm 1. Then, an $r \times r$ matrix $\mathbf{X}_r$ is constructed using the i_1-th, i_2-th, $\cdots$, i_r-th rows of $\mathbf{X}$. The r shares of the j-th pixel are obtained as $\mathbf{s}_r(j) = [s_{i_1}(j), s_{i_2}(j), \cdots, s_{i_r}(j)]^{\mathrm{T}}$. Finally, the $r-1$ random numbers and $\mathbf{I}(j)$ can be recovered using Eq. (8).

Most existing secret sharing schemes can only process values in prime finite fields and thus a preprocessing operation is required to process m-bit data. The PFMSS scheme can achieve the following properties: (1) It can directly process m-bit data without preprocessing. (2) The secret sharing and reconstruction processes are all performed using matrix multiplication. (3) It is a nondeterministic system because $r-1$ random numbers $h_2, \cdots, h_r$ are used. Then, even share a secret several times using the same coefficient matrix, the obtained shares are completely different.

IV. PFMSS-Based RDH-EI Scheme

Using PFMSS, this section develops a new RDH-EI scheme called PFMSS-RDHEI and Fig. 2 demonstrates its structure. The content owner encrypts an original image into n encrypted image $\mathbf{EI}_i$, which are sent to n cloud servers from competitive providers. The cloud server is the data hider and can embed additional data into the encrypted image to generate the marked encrypted image $\mathbf{EM}_i$. The receiver can extract the embedded data in each marked encrypted image and can completely recover the original image from r marked encrypted images.

In the existing RDH-EI schemes, the content owner uses a secret key to encrypt the original image, while the receiver should use a related key to recover the original image. Thus, a reliable KMS should be used to manage and distribute these secret keys, which is practically costly to implement such a multiparty environment over insecure public networks [42]. However, in our PFMSS-RDHEI, the security of the image

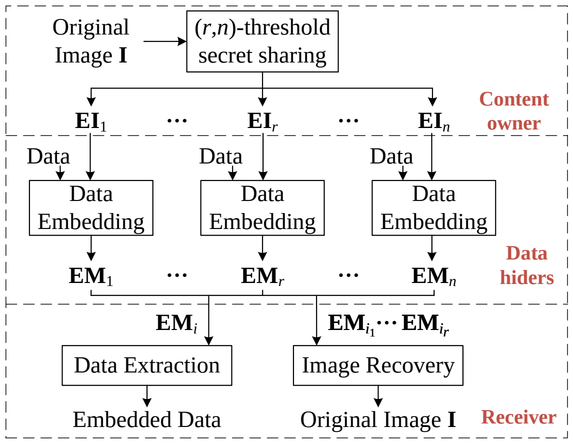


Fig. 2: The structure of our PFMSS-RDHEI.

depends on the (r, n)-threshold secret sharing mechanism, and does not need a secret key.

A. Content Owner

The content owner first encrypts the image $\mathbf{I}$ into n image shares block-by-block, and then embeds some extra information into the shares using a bit-pair encoding method. Finally, n encrypted images are generated and sent to n data hiders.

1) Block-Based (r, n)-Threshold Image Sharing: In our RDH-EI scheme, the content owner encrypts the image block-by-block. Suppose that the 8-bit grayscale image $\mathbf{I}$ is of size $M \times N$ and the block size is $S_1 \times S_2$. For each image block, a coefficient matrix $\mathbf{X}$ is generated using Algorithm 1.

The pixels are shared block-by-block, and the coefficient matrix $\mathbf{X}$ for sharing a block is the same. Then, a total number of $\lceil M/S_1 \rceil \times \lceil N/S_2 \rceil$ coefficient matrix $\mathbf{X}$s are generated using a pseudo-random number generator. For each image block $\mathbf{B}$ of size $S_1 \times S_2$, $r-1$ random integers in $GF(2^8)$, namely $h_2, \cdots, h_r$, are generated. Then the (r, n)-threshold sharing operation to its j-th pixel $\mathbf{B}(j)$ is expressed as

$$\begin{bmatrix} s_1(j) \\ s_2(j) \\ \vdots \\ s_n(j) \end{bmatrix} = \begin{bmatrix} 1 & x_{12} & \cdots & x_{1r} \\ 1 & x_{22} & \cdots & x_{2r} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & x_{n2} & \cdots & x_{nr} \end{bmatrix} \otimes \begin{bmatrix} \mathbf{B}(j) \\ h_2 \\ \vdots \\ h_r \end{bmatrix} \mod F(y), \tag{17}$$

where $F(y) = y^8 + y^4 + y^3 + y + 1$. Note that $h_2, \cdots, h_r$ are completely different and randomly generated when sharing the pixels in different blocks. After sharing all the $\lceil M/S_1 \rceil \times \lceil N/S_2 \rceil$ blocks, n image shares can be generated. Because $F(y)$ is a prime polynomial in $GF(2^8)$, these image shares are also 8-bit grayscale images.

2) Bit-Pair Encoding: In addition, some extra information, including the block sizes S_1 and S_2, parameter r, and identity of the image share, should be embedded into the fixed positions such that the data hider and receiver can extract these information. We use 8 bits to encode each of them and embed them into the last 32 pixels using LSB replacement.

The original LSBs of the last 32 pixels should be embedded into the image share and we develop a bit-pair encoding method to embed them as follows. (1) Divide the image of size $M \times N$ into $MN/2$ adjacent pixel pairs. (2) For each pixel pair, extract the two most significant bits (MSBs) and encode '00', '11', '01' and '10' as '0','10','110' and '111', respectively. (3) Combine all the codes of the MSBs to obtain a bit sequence B. (4) Process t pixel pairs until $2t \geq 32 + \lceil \log_2(MN/2) \rceil + L_B$, where we use $\lceil \log_2(MN/2) \rceil$ bits to represent t and L_B is the length of B. (5) Embed the original LSBs of the last 32 pixels, t with $\lceil \log_2(MN/2) \rceil$ bits and bit sequence B into the $2t$ MSBs of the t processed pixel pairs. Finally, n encrypted images $\mathbf{EI}_1, \mathbf{EI}_2, \cdots, \mathbf{EI}_n$ are generated and sent to n data hiders.

B. Data Hider

After receiving an encrypted image, a data hider can embed data into it to obtain a marked encrypted image. First, the data hider extracts the block sizes S_1 and S_2, parameter r, and the identity of the encrypted image from the LSBs of the last 32 pixels, and then recovers the LSBs of the last 32 pixels using the inverse process of bit-pair encoding. Then, for each block of the encrypted image, the data hider calculates its prediction errors $e(i,j)$s using an improved MED predictor and then encodes the pixels using our ECVE method. Note that the first pixel $p(1,1)$ in each block is not processed. Finally, embed some extra information into the LSBs of the last 32 pixels and embed the encoding result into the image, except for the first pixel of each block. The rest pixels can be used to embed additional data.

1) Improved MED predictor: For each block of the encrypted image, we first predict its pixels to generate the prediction errors using an improved MED predictor. For a pixel $p(i,j)$, its prediction value $\hat{p}(i,j)$ is calculated as

$$\hat{p}(i,j) = \begin{cases} \alpha, & \text{for } i = 1, j \neq 1; \\ \beta, & \text{for } i \neq 1, j = 1; \\ \max(\alpha,\beta), & \text{for } \gamma \leq \min(\alpha,\beta); \\ \min(\alpha,\beta), & \text{for } \gamma \geq \max(\alpha,\beta); \\ \alpha + \beta - \gamma, & \text{otherwise}, \end{cases} \tag{18}$$

where $\alpha = p(i, j-1)$, $\beta = p(i-1, j)$, and $\gamma = p(i-1, j-1)$, which are the adjacent pixels of the current pixel $p(i,j)$ in the left, upper, and upper-left directions, respectively. Then the prediction error $e(i,j)$ is calculated as

$$e(i,j) = p(i,j) - \hat{p}(i,j). \tag{19}$$

2) Concept of ECVE: The ECVE method can sufficiently separate the compressible bits from a pixel according to its prediction error. In the next description, for simplicity, we denote $p(i,j)$, $\hat{p}(i,j)$ and $e(i,j)$ as p, $\hat{p}$ and e, respectively. For an 8-bit pixel p, its prediction error is $e = p - \hat{p}$. According to the range of e, the pixel p can be divided into two parts: the k least significant bits and $(8-k)$ most significant bits, denoted by p_L and p_M, respectively. The p_M are compressible while the p_L are incompressible.

When the prediction error $e = -1$, 0 or 1, the $k = 0$ and all the 8 bits of the pixel p are compressible bits. Otherwise, when $e \in [2^k, 2^{k+1})$ or $e \in (-2^{k+1}, -2^k] (1 \leq k \leq 7)$, the k least significant bits of the pixel, i.e., p_L, should be reserved. Thus, according to the range, all the prediction errors can be divided into 17 classes and Table III shows how to encode these 17 classes. Suppose that a pixel p can be decomposed into $(p_8p_7p_6p_5\ p_4p_3p_2p_1)_2$, where p_8 and p_1 are the most and least significant bits, respectively. For example, a pixel $p = 202 = 11001010_2$ and its prediction value $\hat{p} = 211$. Then its prediction error $e = p - \hat{p} = -9 \in (-2^4, -2^3]$. According to Table III, the code of e is 00002, and $p_L = p_3p_2p_1 = 010_2$.

TABLE III: Codes of the prediction error e.

	e	k	Code of e	p_L
1	$e \in (-2^8, -2^7]$	7	2	$p_7p_6p_5\ p_4p_3p_2p_1$
2	$e \in (-2^7, -2^6]$	6	02	$p_6p_5\ p_4p_3p_2p_1$
3	$e \in (-2^6, -2^5]$	5	002	$p_5\ p_4p_3p_2p_1$
4	$e \in (-2^5, -2^4]$	4	0002	$p_4p_3p_2p_1$
5	$e \in (-2^4, -2^3]$	3	0000 2	$p_3p_2p_1$
6	$e \in (-2^3, -2^2]$	2	0000 02	p_2p_1
7	$e \in (-2^2, -2^1]$	1	0000 002	p_1
8	$e = -1$	0	0000 0002	$*$
9	$e = 0$	0	0000 0000	$*$
10	$e = 1$	0	0000 0001	$*$
11	$e \in [2^1, 2^2)$	1	0000 001	p_1
12	$e \in [2^2, 2^3)$	2	0000 01	p_2p_1
13	$e \in [2^3, 2^4)$	3	0000 1	$p_3p_2p_1$
14	$e \in [2^4, 2^5)$	4	0001	$p_4p_3p_2p_1$
15	$e \in [2^5, 2^6)$	5	001	$p_5\ p_4p_3p_2p_1$
16	$e \in [2^6, 2^7)$	6	01	$p_6p_5\ p_4p_3p_2p_1$
17	$e \in [2^7, 2^8)$	7	1	$p_7p_6p_5\ p_4p_3p_2p_1$

3) Reversibility of ECVE: Here, we conclude that a pixel p can be recovered using the code of e, its predication value $\hat{p}$, and the p_L. For the pixel p, it consists of the k least significant bits p_L and $(8-k)$ most significant bits p_M. Correspondingly, suppose that its prediction value $\hat{p}$ can be divided into two parts, $\hat{p}_L$ and $\hat{p}_M$. The recovery process of the pixel p is as follows.

First, according to the code of e and Table III, we can obtain the range of e. Then, based on the range, the pixel p or its $(8-k)$ most significant bits p_M can be recovered as follows.

- When $e \in [-1, 1]$, we can directly obtain that $p = \hat{p} + e$.
- When $e \in [2^k, 2^{k+1}) (1 \leq k \leq 7)$, the p_M can be obtained by

$$p_M = \begin{cases} \hat{p}_M + 1, & p_L \geq \hat{p}_L; \\ \hat{p}_M + 2, & p_L < \hat{p}_L. \end{cases} \tag{20}$$

- When $e \in (-2^{k+1}, -2^k] (1 \leq k \leq 7)$, the p_M can be obtained by

$$p_M = \begin{cases} \hat{p}_M - 1, & p_L \leq \hat{p}_L; \\ \hat{p}_M - 2, & p_L > \hat{p}_L. \end{cases} \tag{21}$$

When $e \in [2^k, 2^{k+1})$ or $e \in (-2^{k+1}, -2^k]$, we can recover the pixel p by combining the $(8-k)$ most significant bits p_M and the k least significant bits p_L.

To better show the process of our ECVE method, we provide an illustrate example using a block with size 2×2 and show the encoding and decoding processes in Fig. 3. Note that the first pixel $p(1,1)$ of the block is not processed. We first calculate the prediction values of all other pixels using the improved MED predictor in Eq. (18). Then, the encoding process to the three pixels $p(1,2)$, $p(2,1)$ and $p(2,2)$ is as follows:

- For the pixel $p(1,2)$, its prediction value $\hat{p} = 24$ according to Eq. (18) and then its prediction error $e = p - \hat{p} =$

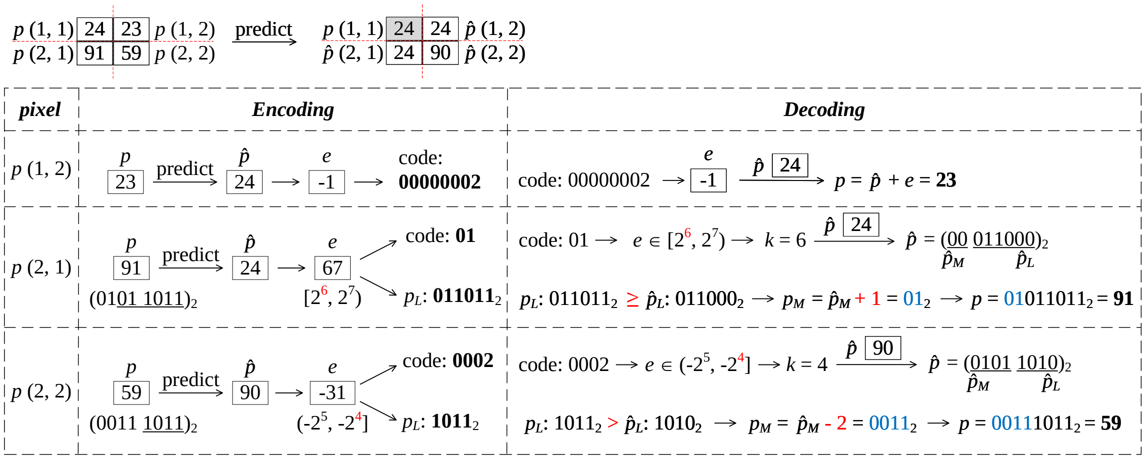

p (1, 1) 24 23 p (1, 2)
p (2, 1) 91 59 p (2, 2)
predict
p (1, 1) 24 24 p̂ (1, 2)
p̂ (2, 1) 24 90 p̂ (2, 2)
pixel
Encoding
Decoding
p (1, 2)
code: 00000002
code: 00000002 → -1 → p = p̂ + e = 23
p (2, 1)
code: 01
pL: 011011₂
(0101 1011)₂
[2⁶, 2⁷)
code: 01 → e ∈ [2⁶, 2⁷) → k = 6 → p̂ = (00 011000)₂
pL: 011011₂ ≥ p̂L: 011000₂ → pM = p̂M + 1 = 01₂ → p = 01011011₂ = 91
p (2, 2)
code: 0002
pL: 1011₂
(0011 1011)₂
(-2⁵, -2⁴]
code: 0002 → e ∈ (-2⁵, -2⁴] → k = 4 → p̂ = (0101 1010)₂
pL: 1011₂ > p̂L: 1010₂ → pM = p̂M - 2 = 0011₂ → p = 00111011₂ = 59

Fig. 3: Encoding and decoding examples of our ECVE method.

-1. According to Table III, the code of e is 00000002 and the pixel doesn't have p_L.

- For the pixel $p(2,1)$, its prediction value $\hat{p} = 24$ and then its prediction error $e = p - \hat{p} = 67 \in [2^6, 2^7)$. Then the code of e is 01 and the p_L of $p(2,1)$ is $p_6p_5p_4p_3p_2p_1 = 011011_2$.
- For the pixel $p(2,2)$, its prediction value $\hat{p} = 90$ and then its prediction error $e = p - \hat{p} = -31 \in (-2^5, -2^4]$. Then the code of e is 0002 and the p_L of $p(2,2)$ is $p_4p_3p_2p_1 = 1011_2$.

The decoding process to the three pixels is as follows:

- For the pixel $p(1,2)$, since the code of its prediction error e is 00000002, then $e = -1$ according to Table III. Besides, we can obtain its prediction value $\hat{p} = 24$. Then the pixel $p(1,2)$ is recovered as $p = \hat{p} + e = 23$.
- For the pixel $p(2,1)$, since the code of its prediction error e is 01, then $e \in [2^6, 2^7)$ and $k = 6$. Since $\hat{p} = 24$, $\hat{p}_M = 00_2$ and $\hat{p}_L = 011000_2$. Since we can obtain its 6 least significant bits as $p_L = 011011_2$ and $p_L \geq \hat{p}_L$, the 2 most significant bits are obtained as $p_M = \hat{p}_M + 1 = 00_2 + 1 = 01_2$. Thus, we can recover the pixel $p(2,1)$ as $p = p_M || p_N = 01011011_2 = 91$.
- For the pixel $p(2,2)$, since the code of its prediction error e is 0002, then $e \in (-2^5, -2^4]$ and $k = 4$. Since $\hat{p} = 90$, $\hat{p}_M = 0101_2$ and $\hat{p}_L = 1010_2$. Since we can obtain its 4 least significant bits as $p_L = 1011_2$ and $p_L > \hat{p}_L$, the 4 most significant bits are obtained as $p_M = \hat{p}_M - 2 = 0101_2 - 2 = 0011_2$. Thus, we can recover the pixel $p(2,2)$ as $p = p_M || p_N = 00111011_2 = 59$.

4) Encoding Process of ECVE: The ECVE method is used to encode the image pixels. Except for the first pixel of each block, the p_L of all pixels and their prediction errors' codes should be stored and encoded. We use a binary vector $\mathbf{P_L}$ to store all p_L and use L to store the corresponding length. Because the code of prediction error e may contain at most eight bits, we use eight vectors $\{\mathbf{C}_1, \mathbf{C}_2, \cdots, \mathbf{C}_8\}$ to store the codes of all the pixels' prediction errors in the 8-bit planes. Because these codes of prediction errors have different lengths, the eight vectors also have different lengths. Notice that the proportion of 0s is significantly larger than the sum of proportions of 1s and 2s. These eight vectors are sparse and can be significantly compressed. After compressing these eight vectors using the arithmetic code, we can obtain eight compressed vectors $\{\mathbf{CC}_1, \mathbf{CC}_2, \cdots, \mathbf{CC}_8\}$. To completely recover the original $\{\mathbf{C}_1, \mathbf{C}_2, \cdots, \mathbf{C}_8\}$, the numbers 0s, 1s and 2s should be stored. We use 24 integers $\mathbf{N} = \{N_1, N_2, \cdots, N_{24}\}$ to record them. In addition, we use eight integers $\mathbf{LC} = \{LC_1, LC_2, \cdots, LC_8\}$ to record the lengths of $\{\mathbf{CC}_1, \mathbf{CC}_2, \cdots, \mathbf{CC}_8\}$. L can be encoded using $3 + \lceil \log_2(MN) \rceil$ bits, whereas LC_i and N_i can be encoded using $\lceil \log_2(MN) \rceil + 1$ and $\lceil \log_2(MN) \rceil$ bits, respectively. Thus, all the pixels, except for the first pixel of each block, can be encoded as $\mathbf{L} = L \| \mathbf{N} \| \mathbf{LC} \| \mathbf{P_L} \| \mathbf{CC}_1 \| \cdots \| \mathbf{CC}_8$.

Finally, the block sizes S_1 and S_2, parameter r, and the identity of the encrypted image are embedded into the LSBs of the last 32 pixels. If the last 32 pixels contain the first pixels of some blocks, these first pixels can be put at the end of $\mathbf{L}$ for lossless recovery. The data $\mathbf{L}$ should be pre-embedded in the encrypted image. Except for the first pixel of each block, all pixels can be vacated for data embedding.

5) Data Hiding: The vacated room for data embedding is obtained from the pixels of all image blocks except for the first pixel. The data hider can determine the position of the first available space from the length of $\mathbf{L}$. Note that, if the additional data are confidential, the data hider can encrypt them using an existing cryptographic algorithm (e.g., AES) before embedding.

C. Receiver

A receiver can extract additional data from a marked encrypted image and recover the original image from r marked encrypted images.

1) Data Extraction: For a marked encrypted image, the receiver can extract the additional data as follows:

- *Step 1*: Extract the block sizes S_1 and S_2, parameter r, and the identity of the encrypted image from the LSBs of the last 32 pixels.
- *Step 2*: Divide the marked encrypted image into non-overlapping blocks with size $S_1 \times S_2$. Retrieve all bits

except for the first pixel of each block. These bits comprise the image codes and embedded data.
- *Step 3*: Exclude the codes of the image and obtain the embedded data.

2) Image Recovery: A receiver with r marked encrypted images can recover the original image as follows:

- *Step 1*: For each marked encrypted image, extract the block sizes S_1 and S_2, parameter r, and the identity from the LSBs of the last 32 pixels.
- *Step 2*: Decode the r marked encrypted images using ECVE method to recover the r encrypted images.
- *Step 3*: Generate the coefficient matrix $\mathbf{X}$ for each image block using Algorithm 1 with the public pseudo-random number seed. Suppose that the identities of the r marked encrypted images are $i_1, i_2, \cdots, i_r$. Construct matrix $\mathbf{X}_r$ using the i_1-th, i_2-th, $\cdots$, i_r-th rows of $\mathbf{X}$. Obtain the inverse matrix $\mathbf{X}_r^{-1}$.
- *Step 4*: For the k-th image block, its j-th pixel can be recovered using the j-th pixels of the k-th image blocks of the r encrypted images, namely $s_{i_1}(j), s_{i_2}(j), \cdots, s_{i_r}(j)$. The reconstruction process is defined as follows:

$$\begin{bmatrix}\mathbf{B}(j)\\ h_2\\ \vdots\\ h_r\end{bmatrix} = \mathbf{X}_r^{-1} \otimes \begin{bmatrix}s_{i_1}(j)\\ s_{i_2}(j)\\ \vdots\\ s_{i_r}(j)\end{bmatrix} \mod F(y). \tag{22}$$

Once all pixels have been recovered, the k-th image block can be obtained. After all image blocks are recovered, the original image $\mathbf{I}$ is acquired.

V. Property Discussion and Experimental Results

This section discusses the data embedding property of the proposed RDH-EI scheme and presents the simulation results.

A. Redundancy Preservation

In this section, we analyze that some data redundancy remains within each block in the encrypted images. In our scheme, an image is encrypted to be n image shares using our proposed secret sharing method. For an image $\mathbf{B}$ of size $S_1 \times S_2$, the encryption process of its j-th pixel $\mathbf{B}(j)$ is shown in Eq. (17). Thus, the i-th sharing result for the pixel $\mathbf{B}(j)$ can be calculated as

$$s_i(j) = \mathbf{B}(j) \oplus (x_{i2} \otimes h_2) \oplus \cdots \oplus (x_{ir} \otimes h_r) \mod F(y), \tag{23}$$

where $\oplus$ is the bitwise XOR operation. Denote that $A_i = (x_{i2} \otimes h_2) \oplus \cdots \oplus (x_{ir} \otimes h_r) \mod F(y)$. Then

$$s_i(j) = \mathbf{B}(j) \oplus A_i. \tag{24}$$

When encrypting the $S_1 \times S_2$ pixels in the image block $\mathbf{B}$, the coefficient matrix $\mathbf{X}$ and $r-1$ random integers $h_2, \cdots, h_r$ are kept the same. Then for the t-th pixel $\mathbf{B}(t)$ in the image block $\mathbf{B}$, its i-th share $s_i(t)$ can be generated using the same way as $s_i(j)$,

$$s_i(t) = \mathbf{B}(t) \oplus A_i. \tag{25}$$

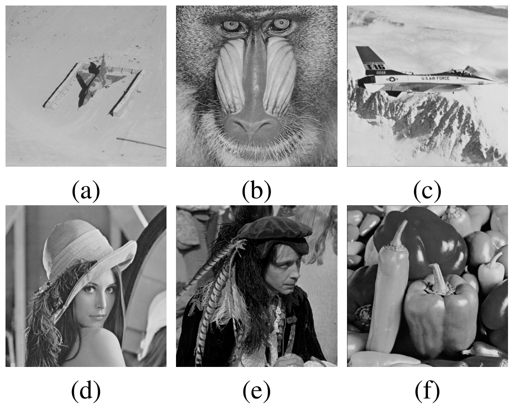


Fig. 4: Six classical images with size 512×512. (a) *Airplane*; (b) *Baboon*; (c) *Jetplane*; (d) *Lena*; (e) *Man*; (f) *Peppers*.

Since $(n_1 \oplus n_3) \oplus (n_2 \oplus n_3) = n_1 \oplus n_2$, we can get that

$$\begin{aligned} s_i(j) \oplus s_i(t) &= (\mathbf{B}(j) \oplus A_i) \oplus (\mathbf{B}(t) \oplus A_i) \\ &= \mathbf{B}(j) \oplus \mathbf{B}(t). \end{aligned} \tag{26}$$

This indicates that the XOR correlation of the two pixels $\mathbf{B}(j)$ and $\mathbf{B}(t)$ can be preserved in the generated image shares. Thus, when using the same parameters to share an image block, the generated n image shares have the same XOR correlation with the original image within the block. Because a natural image usually has high data redundancy, the generated n image shares can inherent some of the data redundancy when they have the same XOR correlation with the original image. Then our proposed ECVE method can well utilize these data redundancy to vacate room for data embedding.

B. Experimental Results

In this section, we simulate the proposed PFMSS-RDHEI, and evaluate its performance.

1) Experiment Settings: We first present the implementation settings in our experiments.

Datasets. We use six classical images and two image datasets "BOSSbase"[1] and the grayscaled "FFHQ"[2] as the test images. The six images are shown in Fig. 4. The dataset "BOSSbase" contains 10,000 grayscale images, and these images contain natural scenery, human face and all kinds of topics. The dataset "FFHQ" contains 52,001 images, and we use it to test the feasibility of our scheme. All the test images are grayscale images of size 512×512.

Evaluation Metrics. We evaluate our PFMSS-RDHEI scheme from the aspects of security level, embedding rate (ER), recovery ability and separation ability. A higher level of security enables better protection of the image content. The ER reflects the embedding capacity of each encrypted image by a data hider. The recovery ability indicates the recovery quality of the original image. The separation ability indicates whether the data extraction and image recovery are separate or not.

Implementation Environment. For all the competing methods, we implemented them strictly following the settings of

[1]http://agents.fel.cvut.cz/stegodata/
[2]https://www.kaggle.com/datasets/arnaud58/flickrfaceshq-dataset-ffhq

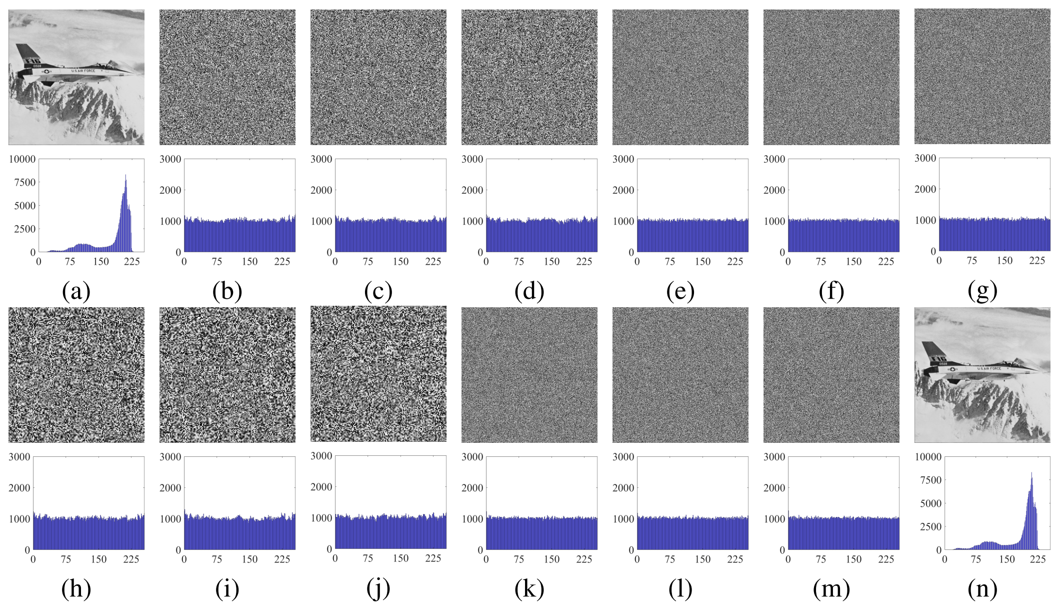


Fig. 5: Simulation results of the PFMSS-RDHEI scheme under $(3, 3)$-threshold with block sizes 2×2 and 4×4. (a) Original image $Jetplane$; (b)-(d) Three encrypted images with block size 2×2; (e)-(g) Three marked encrypted images with secret data embedded in (b)-(d); (h)-(j) Three encrypted images with block size 4×4; (k)-(m) Three marked encrypted images with secret data embedded in (h)-(j); (o) Reconstructed image from (e)-(g) or (k)-(m).

TABLE IV: Embedding rates of $(3, 3)$-threshold PFMSS-RDHEI for different images with different block sizes ($S_1 = S_2 = S$).

Images	Encrypted Images	PFMSS-RDHEI				
		$S = 2$	$S = 4$	$S = 8$	$S = 16$	$S = 32$
Airplane	1^{st}	2.668	3.322	3.473	3.505	3.508
	2^{nd}	2.668	3.321	3.474	3.504	3.507
	3^{rd}	2.668	3.321	3.474	3.504	3.510
Baboon	1^{st}	0.401	0.515	0.548	0.557	0.565
	2^{nd}	0.401	0.515	0.549	0.558	0.567
	3^{rd}	0.401	0.515	0.548	0.559	0.562
Jetplane	1^{st}	1.877	2.386	2.527	2.571	2.589
	2^{nd}	1.877	2.386	2.528	2.570	2.583
	3^{rd}	1.877	2.386	2.527	2.570	2.588
Lena	1^{st}	1.568	1.973	2.083	2.111	2.120
	2^{nd}	1.568	1.974	2.083	2.112	2.123
	3^{rd}	1.568	1.973	2.083	2.114	2.117
Man	1^{st}	1.250	1.570	1.655	1.678	1.688
	2^{nd}	1.249	1.570	1.654	1.680	1.688
	3^{rd}	1.249	1.570	1.655	1.679	1.687
Peppers	1^{st}	1.418	1.762	1.842	1.856	1.861
	2^{nd}	1.418	1.763	1.842	1.857	1.863
	3^{rd}	1.418	1.762	1.842	1.856	1.859
"BOSSbase"	1^{st}	2.178	2.763	2.929	2.981	3.000
	2^{nd}	2.178	2.763	2.929	2.981	3.000
	3^{rd}	2.178	2.763	2.929	2.981	3.000
"FFHQ"	1^{st}	1.991	2.518	2.664	2.708	2.723
	2^{nd}	1.991	2.518	2.664	2.708	2.723
	3^{rd}	1.991	2.518	2.664	2.708	2.723

their original papers. All the programs are developed by CLion 2020 with the C++ programming language and run on 64-bit Windows 11 with AMD Ryzen 7 5800H @3.20GHz, 16GB RAM.

2) Simulation Results: Fig. 5 simulates the PFMSS-RDHEI scheme using image $Jetplane$ under $(3, 3)$-threshold with block sizes 2×2 and 4×4. The embedded data are some randomly generated bits. It is obvious that the PFMSS-RDHEI scheme can encrypt a meaningful image into unrecognized encrypted images with uniform distribution. Since the encryption is done block-by-block, some data redundancy retains within each block for data embedding. Thus, the visual effect is better with smaller block size, which can be seen from the figures. However, even when block artifact exists, all pixels in the encrypted images can be distributed very uniformly, as shown in their histograms. This means that the image content can be well protected. When embedding additional data into these encrypted images, the obtained marked encrypted images also have uniform pixel distributions.

3) Embedding Rate: The embedding capacity indicates the number of additional data that can be embedded into each encrypted image by a data hider. It can be evaluated using the embedding rate (ER) as bits per pixel (*bpp*), which is calculated as

$$ER = \frac{\text{Total embedding capacity - Payload}}{\text{Total number of pixels}}. \tag{27}$$

Since the first pixel of each image block is not encoded, the total embedding capacity is $8 \times (M \times N - \lceil M/S_1 \rceil \times \lceil N/S_2 \rceil)$. The payload includes the codes of pixels $\mathbf{L}$ and the extra information embedded in the LSBs of the last 32 pixels, as discussed in Section IV-B4. Thus, the ER can be obtained as

$$ER = \frac{8 \times (M \times N - \lceil M/S_1 \rceil \times \lceil N/S_2 \rceil) - \mathbf{L} - 32}{M \times N}. \tag{28}$$

There are four parameters in our (r, n)-threshold RDH-EI scheme: the parameters r, n, and block sizes S_1 and S_2. The proposed PFMSS encrypts an image into n encrypted images with the same size and it can be recovered using r ones. Then, the parameters r and n have a slight influence on the ER. Because the proposed ECVE method encodes the encrypted image block-by-block, the ER is significantly affected by the block sizes S_1 and S_2. Table IV lists the embedding rates of

TABLE V: Data expansion rates of the homomorphic encryption-based and secret sharing-based RDH-EI schemes.

Methods	Content owner	Each data hider
Zheng $et\ al.$ [39]	8, 16	8, 16
Chen $et\ al.$ [40]	64, 128	64, 128
Chen $et\ al.$ [26]	n	1
Chen $et\ al.$ [27]	n	1
Qin $et\ al.$ [41]	n	1
PFMSS-RDHEI	n	1

the $(3, 3)$-threshold scheme for different images with block sizes $2, 4, 8, 16, 32$. Notice that the ERs significantly improve with the increase of block size. One has great flexibility in selecting a proper block size according to different situations.

As can be seen from Table IV, the image *Airplane* has much higher ERs than other images under the same block size. This is because it has higher smoothness and thus its prediction errors are closer to 0. We only list the embedding rates of our PFMSS-RDHEI scheme under the $(3, 3)$-threshold, because the settings of r and n cause a slight change in the embedding rates. With different (r, n)-thresholds, we can achieve approximately the same embedding rates.

VI. Performance Analysis

This section evaluates the performance of our scheme and compares it with other secret sharing-based RDH-EI schemes.

A. Data Expansion

In an RDH-EI scheme, data expansion occurs when the total size of the marked encrypted image is larger than that of the original image. The expansion rate is introduced to calculate the data expansion, which is defined as the ratio between the total bits of the marked encrypted image and the total bits of the original image. For most traditional VRAE-based schemes, data expansion does not occur. However, they usually use some lightweight encryption methods to keep data redundant in the encrypted domain, and thus have limited security levels [37], [38]. Here, we evaluate the data expansion in the homomorphic encryption-based and secret sharing-based RDH-EI schemes because they have high security levels. Table V lists the data expansion comparison of the homomorphic encryption-based and secret sharing-based RDH-EI schemes. A secret sharing-based RDH-EI scheme encrypts an original image into n encrypted images and sends them to n data hiders for data embedding. However, the homomorphic encryption-based RDH-EI schemes only have one data hider. Data expansion occurs seriously in these schemes that apply homomorphic encryption [39], [40]. Note that for the two encryption schemes in [39], the data expansions are 8 times and 16 times, respectively, and the data expansions are determined by the security parameters set by users. Compared to the homomorphic encryption-based schemes, the secret sharing-based schemes in [26], [27], [41] and our PFMSS-RDHEI scheme have much less data expansion with acceptable expansion rates. The expansion rate of all secret sharing-based schemes is n for the entire scheme, however, is 1 for each data hider, indicating no data expansion. Note that for the scheme in [26], there is only one encrypted image, which means that $n = 1$.

TABLE VI: Embedding rates of our PFMSS-RDHEI scheme using different embedding methods with $(3, 3)$-threshold and block size 4×4.

Images	Encrypted Images	Methods [13]	[14]	ECVE
Airplane	1^{st}	2.3401	2.5878	**3.3211**
	2^{nd}	2.3400	2.5878	**3.3212**
	3^{rd}	2.3397	2.5876	**3.3210**
Baboon	1^{st}	0.3637	0.3255	**0.5145**
	2^{nd}	0.3637	0.3254	**0.5144**
	3^{rd}	0.3638	0.3255	**0.5145**
Jetplane	1^{st}	1.9074	1.9659	**2.3856**
	2^{nd}	1.9074	1.9660	**2.3857**
	3^{rd}	1.9072	1.9657	**2.3854**
Lena	1^{st}	1.4999	1.6705	**1.9735**
	2^{nd}	1.4999	1.6705	**1.9734**
	3^{rd}	1.4994	1.6703	**1.9732**
Man	1^{st}	1.2218	1.2667	**1.5702**
	2^{nd}	1.2218	1.2668	**1.5702**
	3^{rd}	1.2222	1.2669	**1.5704**
Peppers	1^{st}	1.3796	1.4775	**1.7628**
	2^{nd}	1.3795	1.4775	**1.7629**
	3^{rd}	1.3794	1.4774	**1.7628**

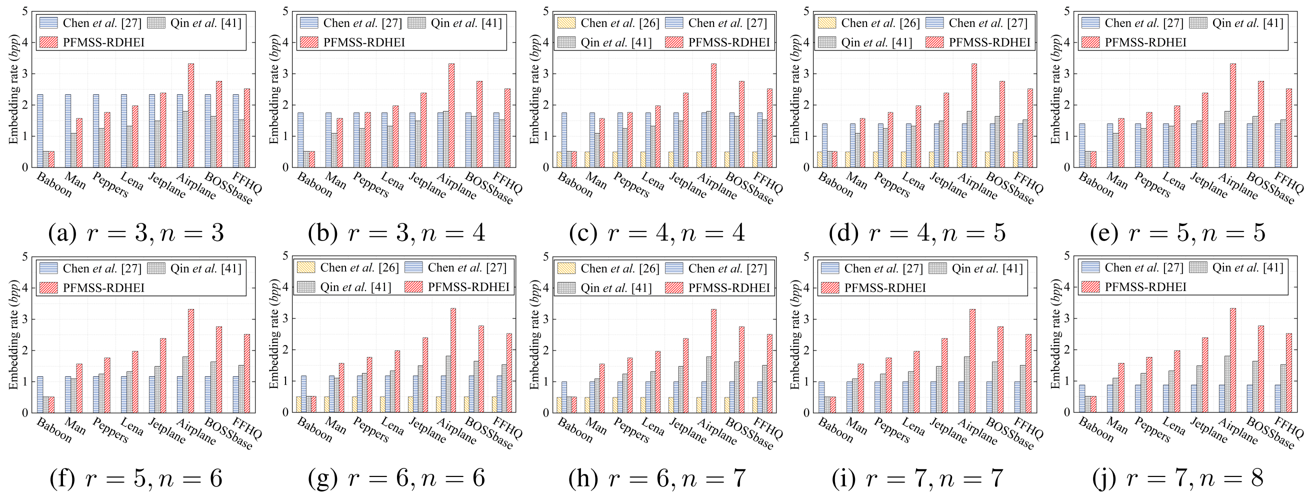
B. Performance of ECVE

Since our developed ECVE method can sufficiently separate the compressible and incompressible bits of a pixel, it can vacate a large room for data embedding. To show its high performance, we design experiment to compare it with two state-of-the-art encoding methods in [13], [14]. Specifically, we test the embedding rates of our PFMSS-RDHEI scheme using our ECVE and the encoding methods in [13], [14], respectively. The block size is set as 4×4 and the threshold is set as $(3, 3)$.

Table VI shows the embedding rates of encrypted images using the three encoding methods with block size 4×4. As can be seen, our PFMSS-RDHEI scheme can achieve the largest embedding rates when using ECVE. This is because our ECVE divides prediction errors into 17 classes, which is shown in Table III. For a pixel p, when its prediction error e satisfies that $|e| \in [2^k, 2^{k+1})$ $(0 \leq k \leq 6)$, the methods in [13], [14] can separate at most $6 - k$ and $7 - k$ embeddable bits, respectively, while our ECVE method can separate $8 - k$ embeddable bits for data embedding. Besides, when $|e| \in [2^7, 2^8)$, our ECVE method can separate one embeddable bit; however, the embedding methods in [13], [14] cannot separate any embeddable bit. When $e = 0$, our ECVE and the method in [13] can use all the eight bits of the pixel for data embedding; however, the method in [14] only separates seven embeddable bits. As a result, for a same prediction error, our ECVE method can separate more compressible bits and thus can vacate more room for data embedding, compared to the encoding methods in [13], [14].

C. Comparison of Embedding Capacity

The developed PFMSS can encrypt images into encrypted images with high data redundancy within each block, and the proposed ECVE method can fully utilize the data redundancy. Thus, our RDH-EI scheme can achieve a large embedding



(a) $r = 3, n = 3$ (b) $r = 3, n = 4$ (c) $r = 4, n = 4$ (d) $r = 4, n = 5$ (e) $r = 5, n = 5$

(f) $r = 5, n = 6$ (g) $r = 6, n = 6$ (h) $r = 6, n = 7$ (i) $r = 7, n = 7$ (j) $r = 7, n = 8$

Fig. 6: Embedding rate comparisons between different secret sharing-based RDH-EI schemes under different settings of r and n.

TABLE VII: Theoretical time complexity of different secret sharing techniques.

(r, n)			Methods			
			Chen *et al.* [26]	Chen *et al.* [27]	Qin *et al.* [41]	PFMSS
$(2, n)$	One sharing for a pixel	Addition	3	1	1	1
		Multiplication	6	1	1	1
	One sharing for image of size $M \times N$	Addition	$3MN$	MN	MN	MN
		Multiplication	$6MN$	MN	MN	$\frac{1}{S^2}MN$
(r, n)	One sharing for a pixel	Addition	$2r - 1$	$r - 1$	$r - 1$	$2r - 3$
		Multiplication	$r(2r - 1)$	$\frac{r(r-1)}{2}$	$\frac{r(r-1)}{2}$	$2r - 3$
	One sharing for image of size $M \times N$	Addition	$(2r - 1)MN$	$(r - 1)MN$	$(r - 1)MN$	$\frac{(2r-4+S^2)}{S^2}MN$
		Multiplication	$r(2r - 1)MN$	$\frac{r(r-1)}{2}MN$	$\frac{r(r-1)}{2}MN$	$\frac{(2r-3)}{S^2}MN$

capacity. To demonstrate the high embedding capacity of our proposed scheme, we compare it with other secret sharing-based RDH-EI schemes. The block size in our PFMSS-RDHEI scheme is set as 4×4. Since the different encrypted images of our scheme have slightly different embedding rates, as shown in Table IV, we use the average embedding rates for comparison. The Chen *et al.* [26] method is not applicable when r is odd. The Qin *et al.* [41] method contains two versions. The version without preprocessing is selected and the parameter is set as the optimal value for each image.

Fig. 6 demonstrates the embedding rates of different secret sharing-based RDH-EI schemes with different settings of r and n. As can be seen, the embedding rates of Chen *et al.* [26] and Qin *et al.* [41] methods remain approximately the same with different r or n. The Chen *et al.* [27] method can achieve the best embedding rate when $n = 2$, since its embedding rate is $7/n$. It is obvious that for most test images, with an increase of r and n, our PFMSS-RDHEI scheme can achieve the largest embedding rates. The comparison results prove the superiority of our scheme in embedding capacity.

D. Security Evaluation

Our proposed scheme encrypts an image block-by-block. Our design intentionally uses the same matrix coefficients to share the pixels within each block such that some pixel correlation can remain for subsequent data embedding. But we use different and randomly generated matrix coefficients to share the pixels in different blocks. When sharing two pixels

TABLE VIII: Running time comparison (in seconds) of different secret sharing-based RDH-EI schemes for image *Lena*.

(r, n)	Parties	Methods			
		Chen *et al.* [26]	Chen *et al.* [27]	Qin *et al.* [41]	PFMSS-RDHEI
$(2, 2)$	Content owner	0.0018	0.0021	0.0013	0.0004
	Data hider	0.0069	0.0051	0.0079	0.0433
	Receiver	0.0155	0.0281	0.0722	0.0851
$(3, 3)$	Content owner	-	0.0037	0.0157	0.0008
	Data hider	-	0.0059	0.0057	0.0436
	Receiver	-	0.0472	0.1178	0.1318
$(4, 4)$	Content owner	0.0213	0.0192	0.0353	0.0126
	Data hider	0.0157	0.0066	0.0053	0.0444
	Receiver	0.0433	0.0630	0.1542	0.1780

using different and random parameters, the results are random and do not retain any information of the two pixels. We have theoretically discussed and proved this security property in Lemma 2 of Section III-C. Suppose that an original image size is $M \times N$ and the block size is $S_1 \times S_2$. Since the security of a block is dependent on its first pixel and the sharing result is within $[0, 255]$, the space of the sharing operation is $256^{\lceil M/S_1 \rceil \times \lceil N/S_2 \rceil}$. When setting a small block size such as 2×2, the space of the brute-force attack is $256^{\lceil 256/2 \rceil \times \lceil 256/2 \rceil} = 256^{128 \times 128}$ and the encrypted image can keep a high security level. When the block size increases, more pixel redundancy can be preserved by our ECVE method for embedding more data, but the image security may degrade accordingly. As a result, although some pixel correlation exists

TABLE IX: Comparisons of different VRAE-based RDH-EI schemes.

Methods	Encryption Strategies	Lossless Recovery	Separable	Free-Preprocessing	Security	Resistance to Single Point of Failure	Without KMS
Bhardwaj $et\ al.$ [34]	XOR	×	×	✓	Limited	×	×
Yi $et\ al.$ [32]	Block Permutation and Co-Modulation	✓	✓	✓	Limited	×	×
Liu $et\ al.$ [35]	In-Block Pixel Permutation and Co-XOR	✓	✓	✓	Limited	×	×
Wang $et\ al.$ [36]	Block Permutation and Co-XOR	✓	✓	✓	Limited	×	×
Zheng $et\ al.$ [39]	Homomorphic Encryption	✓	✓	✓	High	×	×
Chen $et\ al.$ [40]	Homomorphic Encryption	✓	✓	✓	High	×	×
Chen $et\ al.$ [26]	Secret Sharing	✓	×	×	Limited	×	×
Chen $et\ al.$ [27]	Secret Sharing	✓	✓	×	High	✓	×
Qin $et\ al.$ [41]	Secret Sharing	✓	✓	✓/×*	High	✓	×
PFMSS-RDHEI	Secret Sharing	✓	✓	✓	High	✓	✓

*indicating one version needs preprocessing while the other doesn't need.

in each block, our scheme can balance the trade-off between the security and embedding capacity by setting a proper block size. Besides, our scheme can keep the image confidentiality from the collusion attack of $r-1$ cloud servers, and can withstand $n-r$ points of failure.

E. Complexity Analysis

We analyze the complexity of our new secret sharing scheme PFMSS and the developed PFMSS-RDHEI scheme. Table VII lists the theoretical time complex of our PFMSS and the sharing schemes used in [26], [27], [41]. Note that in our PFMSS and the sharing scheme used in [41], the addition and multiplication operations denote the XOR and polynomial multiplication, respectively. When sharing a pixel, our PFMSS requires the minimum numbers of multiplication, because $2r-3 \leq \frac{r(r-1)}{2}$ with a positive integer r. Even our PFMSS requires more addition operations than the sharing schemes used in [27], [41], it can still has a lower time complexity, because the multiplication operation is much more time-consuming than the addition operation. Besides, since our PFMSS encrypts an image block-by-block using the same parameters, it requires to calculate only one matrix for encrypting a block. Thus, our PFMSS has the lowest time complexity in sharing an image when setting the block size S as an integer larger than 1.

We also test the actual time cost of different secret sharing-based RDH-EI schemes. We test the running time of different schemes under $(2,2)$, $(3,3)$ and $(4,4)$ thresholds to provide diverse results and set the embedding rate as 0.4 *bpp*. For fairness, set the block size S as 2 in our PFMSS-RDHEI scheme. Table VIII lists the running time of different parities of these schemes. Since the parameter r in the scheme [26] should be even, we only test its results under $(2,2)$-threshold and $(4,4)$-threshold. As can be seen from the results, our PFMSS-RDHEI requires more running time than others under data hider and receiver, because the arithmetic code is used to encode the image data during data hiding. But the data hiders in the cloud service are usually the cloud servers that have strong computation resources. On the other hand, our PFMSS-RDHEI requires the least time in the content owner, which usually has limited performance.

F. Property Comparisons with Prior Studies

Finally, we compare the features of different VRAE-based RDH-EI schemes from different aspects and Table IX lists the results. The scheme in [34] is lossy and the receiver cannot completely recover the original image, while the other schemes are lossless. The schemes in [26], [34] are not separable, which means that the embedded data can only be extracted after the marked encrypted images have been decrypted. Among all secret sharing-based schemes, our scheme and one scheme in [41] can directly encrypt the 8-bit pixels without preprocessing, while the other secret sharing-based schemes in [26], [27] need a preprocessing operation before encryption. To keep data redundant in the encrypted domain, the traditional VRAE-based RDH-EI schemes in [32], [34]–[36] usually use some lightweight encryption methods with limited security levels [37], [38]. The homomorphic encryption-based schemes in [39], [40] can achieve high security levels. However, they suffer from high computation costs and large data expansion. The Chen $et\ al.$ [26] method can only achieve lightweight security with proper parameter settings as reported in the original literature. Our scheme and other secret sharing-based RDH-EI schemes in [27], [41] can ensure image content confidentiality. The schemes in [26], [32], [34]–[36], [39], [40] encrypt an image to be only one encrypted image and thus cannot resist single point of failure. The image cannot be recovered if the only one encrypted image is damaged or lost. Besides, all these existing schemes rely on reliable KMS, which is practically costly to implement in such a multiparty environment over insecure public networks. As a result, our PFMSS-RDHEI scheme is lossless and separable, it can encrypt an image without a preprocessing operation and has a high security level. Besides, it can resist $n-r$ points of failure without relying on a reliable KMS.

VII. Conclusion

In this study, we proposed a new RDH-EI scheme with multiple data hiders. We first proposed an (r,n)-threshold preprocessing-free matrix secret-sharing (PFMSS) method, which can directly encrypt m-bit data using matrix multiplication without preprocessing. Formal analysis is provided to prove its correctness and justify its security. Besides, we further proposed a novel error class and value encoding (ECVE) method. Since it can sufficiently separate compressible significant bits from image pixels, and thus can obtain a large embedding capacity. An illustrate example is provided to show the encoding and decoding processes of ECVE. In our RDH-EI scheme, the content owner can encrypt its image block-by-block using the PFMSS and obtain n encrypted shares for n competing cloud servers. Each data hider can embed data into the encrypted share to obtain a marked encrypted image. The embedded data can be extracted from a marked encrypted image, and the original image is recovered from r marked encrypted images. Experimental results show that the proposed RDH-EI scheme can effectively protect the images and achieve a higher embedding capacity compared to the state-of-the-art RDH-EI schemes. Our future work aims to further enhance the embedding capacity in encrypted domain and investigate the data embedding in video since video has much more data redundancy than image.

References

[1] T. Wang, J. Zhou, X. Chen, G. Wang, A. Liu, and Y. Liu, "A three-layer privacy preserving cloud storage scheme based on computational intelligence in fog computing," *IEEE Trans. Emerg. Top. Comput. Intell.*, vol. 2, no. 1, pp. 3–12, 2018.

[2] M. Celik, G. Sharma, A. Tekalp, and E. Saber, "Lossless generalized-LSB data embedding," *IEEE Trans. Image Process.*, vol. 14, no. 2, pp. 253–266, 2005.

[3] X. Gao, L. An, Y. Yuan, D. Tao, and X. Li, "Lossless data embedding using generalized statistical quantity histogram," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 8, pp. 1061–1070, 2011.

[4] Y. Jia, Z. Yin, X. Zhang, and Y. Luo, "Reversible data hiding based on reducing invalid shifting of pixels in histogram shifting," *Signal Processing*, vol. 163, pp. 238–246, 2019.

[5] J. Tian, "Reversible data embedding using a difference expansion," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 8, pp. 890–896, Aug. 2003.

[6] D. M. Thodi and J. J. Rodriguez, "Expansion embedding techniques for reversible watermarking," *IEEE Trans. Image Process.*, vol. 16, no. 3, pp. 721–730, 2007.

[7] R. Hu and S. Xiang, "Reversible data hiding by using CNN prediction and adaptive embedding," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 44, no. 12, pp. 10 196–10 208, 2022.

[8] J. Chang, G. Zhu, H. Zhang, Y. Zhou, X. Luo, and L. Wu, "Reversible data hiding for color images based on adaptive 3D prediction-error expansion and double deep Q-network," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 8, pp. 5055–5067, 2022.

[9] W. Shen, J. Qin, J. Yu, R. Hao, and J. Hu, "Enabling identity-based integrity auditing and data sharing with sensitive information hiding for secure cloud storage," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 2, pp. 331–346, 2019.

[10] K. Chen and C.-C. Chang, "High-capacity reversible data hiding in encrypted images based on extended run-length coding and block-based MSB plane rearrangement," *J. Vis. Commun. Image Represent.*, vol. 58, pp. 334–344, 2019.

[11] F. Huang, J. Huang, and Y.-Q. Shi, "New framework for reversible data hiding in encrypted domain," *IEEE Trans. Inf. Forensics Secur.*, vol. 11, no. 12, pp. 2777–2789, 2016.

[12] P. Puteaux and W. Puech, "An efficient MSB prediction-based method for high-capacity reversible data hiding in encrypted images," *IEEE Trans. Inf. Forensics Secur.*, vol. 13, no. 7, pp. 1670–1681, Jul. 2018.

[13] A. Mohammadi, M. Nakhkash, and M. A. Akhaee, "A high-capacity reversible data hiding in encrypted images employing local difference predictor," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2366–2376, 2020.

[14] C. Yu, X. Zhang, X. Zhang, G. Li, and Z. Tang, "Reversible data hiding with hierarchical embedding for encrypted images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 2, pp. 451–466, 2022.

[15] Y. Yang, H. He, F. Chen, Y. Yuan, and N. Mao, "Reversible data hiding in encrypted images based on time-varying huffman coding table," *IEEE Trans. Multimedia*, pp. 1–12, 2023, doi: 10.1109/TMM.2023.3238549.

[16] Y. Ke, M. Zhang, and J. Liu, "Separable multiple bits reversible data hiding in encrypted domain," in *Digital Forensics and Watermarking*, 2017, pp. 470–484.

[17] Z. Wang, M. Zhang, Y. Kong, Y. Ke, F. Di, and X. Yan, "Noise modulation-based reversible data hiding with McEliece encryption," *Sec. and Commun. Netw.*, 2022.

[18] Y. Ke, M. Zhang, X. Zhang, Y. Han, and J. Liu, "Reversible data hiding in encrypted domain with public key embedding mechanism," 2022, axXiv preprint arXiv:2208.14510.

[19] Y. Wang, Z. Cai, and W. He, "High capacity reversible data hiding in encrypted image based on intra-block lossless compression," *IEEE Trans. Multimedia*, vol. 23, pp. 1466–1473, 2021.

[20] H. Zou and G. Chen, "Reversible data hiding in encrypted image with local-correlation-based classification and adaptive encoding strategy," *Signal Processing*, vol. 205, p. 108847, 2023.

[21] C. Yu, X. Zhang, G. Li, S. Zhan, and Z. Tang, "Reversible data hiding with adaptive difference recovery for encrypted images," *Information Sciences*, vol. 584, pp. 89–110, 2022.

[22] Z. Yin, Y. Peng, and Y. Xiang, "Reversible data hiding in encrypted images based on pixel prediction and bit-plane compression," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 992–1002, 2022.

[23] Y. Ke, M. Zhang, J. Liu, T. Su, and X. Yang, "A multilevel reversible data hiding scheme in encrypted domain based on LWE," *J. Vis. Commun. Image Represent.*, vol. 54, pp. 133–144, 2018.

[24] H. Ge, Y. Chen, Z. Qian, and J. Wang, "A high capacity multi-level approach for reversible data hiding in encrypted images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 8, pp. 2285–2295, 2019.

[25] K. Gao, J.-H. Horng, and C.-C. Chang, "High-capacity reversible data hiding in encrypted images based on adaptive block encoding," *J. Vis. Commun. Image Represent.*, vol. 84, p. 103481, 2022.

[26] Y.-C. Chen, T.-H. Hung, S.-H. Hsieh, and C.-W. Shiu, "A new reversible data hiding in encrypted image based on multi-secret sharing and lightweight cryptographic algorithms," *IEEE Trans. Inf. Forensics Secur.*, vol. 14, no. 12, pp. 3332–3343, Dec. 2019.

[27] B. Chen, W. Lu, J. Huang, J. Weng, and Y. Zhou, "Secret sharing based reversible data hiding in encrypted images with multiple data-hiders," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 978–991, 2022.

[28] F. A. A. Lins and M. Vieira, "Security requirements and solutions for IoT gateways: A comprehensive study," *IEEE Internet Things J.*, vol. 8, no. 11, pp. 8667–8679, 2021.

[29] F. Chen, Y. Yuan, H. He, M. Tian, and H.-M. Tai, "Multi-MSB compression based reversible data hiding scheme in encrypted images," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 3, pp. 905–916, 2021.

[30] Z. Yin, Y. Xiang, and X. Zhang, "Reversible data hiding in encrypted images based on multi-MSB prediction and huffman coding," *IEEE Trans. Multimedia*, vol. 22, no. 4, pp. 874–884, 2020.

[31] Y. Wu, Y. Xiang, Y. Guo, J. Tang, and Z. Yin, "An improved reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multimedia*, vol. 22, no. 8, pp. 1929–1938, 2020.

[32] S. Yi and Y. Zhou, "Separable and reversible data hiding in encrypted images using parametric binary tree labeling," *IEEE Trans. Multimedia*, vol. 21, no. 1, pp. 51–64, 2019.

[33] Y. Qiu, Q. Ying, Y. Yang, H. Zeng, S. Li, and Z. Qian, "High-capacity framework for reversible data hiding in encrypted image using pixel prediction and entropy encoding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 9, pp. 5874–5887, 2022.

[34] R. Bhardwaj and A. Aggarwal, "An improved block based joint reversible data hiding in encrypted images by symmetric cryptosystem," *Pattern Recognit. Lett.*, vol. 139, pp. 60–68, Nov. 2020.

[35] Z.-L. Liu and C.-M. Pun, "Reversible data hiding in encrypted images using chunk encryption and redundancy matrix representation," *IEEE Trans. Dependable Secure Comput.*, vol. 19, no. 2, pp. 1382–1394, 2022.

[36] Y. Wang and W. He, "High capacity reversible data hiding in encrypted image based on adaptive MSB prediction," *IEEE Trans. Multimedia*, vol. 24, pp. 1288–1298, 2022.

[37] F. Khelifi, "On the security of a stream cipher in reversible data hiding schemes operating in the encrypted domain," *Signal Processing*, vol. 143, pp. 336–345, Feb. 2018.

[38] L. Qu, F. Chen, S. Zhang, and H. He, "Cryptanalysis of reversible data hiding in encrypted images by block permutation and co-modulation," *IEEE Trans. Multimedia*, vol. 24, pp. 2924–2937, 2022.

[39] S. Zheng, Y. Wang, and D. Hu, "Lossless data hiding based on homomorphic cryptosystem," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 2, pp. 692–705, 2021.

[40] B. Chen, X. Wu, W. Lu, and H. Ren, "Reversible data hiding in encrypted images with additive and multiplicative public-key homomorphism," *Signal Processing*, vol. 164, pp. 48–57, 2019.

[41] C. Qin, C. Jiang, Q. Mo, H. Yao, and C.-C. Chang, "Reversible data hiding in encrypted image via secret sharing based on GF(p) and GF(2^8)," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 32, no. 4, pp. 1928–1941, 2022.

[42] R. Chandramouli, M. Iorga, and S. Chokhani, "Cryptographic key management issues and challenges in cloud services," in *Secure Cloud Computing*. Springer, 2014, pp. 1–30.

[43] J. Ding, C. Lin, H. Wang, and C. Xing, "Communication efficient secret sharing with small share size," *IEEE Trans. Inf. Theory*, vol. 68, no. 1, pp. 659–669, 2022.

[44] W. Trappe, *Introduction to Cryptography with Coding Theory*. Pearson Education India, 2006.

[45] Q. Chen, C. Tang, and Z. Lin, "Efficient explicit constructions of multipartite secret sharing schemes," *IEEE Trans. Inf. Theory*, vol. 68, no. 1, pp. 601–631, 2022.

[46] I. Tjuawinata and C. Xing, "Leakage-resilient secret sharing with constant share size," *IEEE Trans. Inf. Theory*, vol. 68, no. 12, pp. 8228–8250, 2022.

[47] M. Rosulek, *The joy of Cryptography*, 2020, ch. secret sharing.

[48] Z. Hua, Y. Wang, S. Yi, Y. Zheng, X. Liu, Y. Chen, and X. Zhang, "Matrix-based secret sharing for reversible data hiding in encrypted images," *IEEE Trans. Dependable Secure Comput.*, pp. 1–18, 2022, doi: 10.1109/TDSC.2022.3218570.

[49] L. Yu, L. Liu, Z. Xia, X. Yan, and Y. Lu, "Lossless and efficient secret image sharing based on matrix theory modulo 256," *Mathematics*, vol. 8, no. 6, art. no. 1018, Jun. 2020.

[50] C.-C. Chen, C.-S. Lin, and J.-Z. Chen, "Boolean-based (k, n, m) multi-secret image sharing," *Axioms*, vol. 11, no. 5, art. no. 197, Apr. 2022.

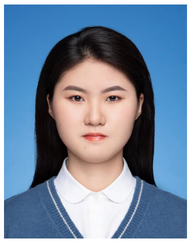
[51] L. Bao, S. Yi, and Y. Zhou, "Combination of sharing matrix and image encryption for lossless (k, n)-secret image sharing," *IEEE Trans. Image Process.*, vol. 26, no. 12, pp. 5618–5631, Dec. 2017.

[52] Z. Liu, G. Zhu, F. Ding, X. Luo, S. Kwong, and P. Li, "Contrast-enhanced color visual cryptography for (k, n) threshold schemes," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 18, no. 3s, Nov. 2022.

[53] X. Wu and P. Yao, "Boolean-based two-in-one secret image sharing by adaptive pixel grouping," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 19, no. 1, Jan. 2023.

Zhongyun Hua (Senior Member, IEEE) received the B.S. degree in software engineering from Chongqing University, Chongqing, China, in 2011, and the M.S. and Ph.D. degrees in software engineering from the University of Macau, Macau, China, in 2013 and 2016, respectively.

He is currently an Associate Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. His works have appeared in prestigious venues such as IEEE Transactions on Dependable and Secure Computing, IEEE Transactions on Image Processing, IEEE Transactions on Signal Processing, and ACM Multimedia. He has been recognized as a 'Highly Cited Researcher 2022'. His current research interests are focused on chaotic system, multimedia security, and secure cloud computing.



Xingyu Liu received her B.S. degree in Computer Science and Technology from Harbin Institute of Technology, Shenzhen, China in 2022. She is currently pursuing the M.S. degree in Computer Science and Technology from Harbin Institute of Technology, Shenzhen, China. Her research interests include reverible data hiding and secret sharing.

Yifeng Zheng is an Assistant Professor with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. He received the PhD degree in computer science from the City University of Hong Kong, Hong Kong, in 2019. He worked as a postdoc with the Commonwealth Scientific and Industrial Research Organization (CSIRO), Australia, and City University of Hong Kong, respectively. His work has appeared in prestigious conferences (such as ESORICS, DSN, AsiaCCS, and PERCOM) and journals (such as TDSC, TIFS, TSC, and TKDE). His current research interests are focused on security and privacy related to cloud computing, IoT, machine learning, and multimedia.

Shuang Yi received the B.S. degree in software engineering from Chongqing University, Chongqing, China, in 2011, and the Ph.D. degree in software engineering from the University of Macau, Macau, China, in 2018.

She is currently a Lecture with the College of Criminal Investigation, Southwest University of Political Science and Law, Chongqing. Her research interests include data hiding, secret sharing, image processing, and multimedia security.

Yushu Zhang (Senior Member, IEEE) received the B.S. degree from the School of Science, North University of China, Taiyuan, China, in 2010, and the Ph.D. degree from the College of Computer Science and Technology, Chongqing University, Chongqing, China, in 2014. He held various research positions with the City University of Hong Kong, Southwest University, the University of Macau, and Deakin University. He is currently a Professor with the College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, China. His research interests include multimedia security, blockchain, and artificial intelligence security. He is an Associate Editor of *Information Sciences*, *Journal of King Saud University-Computer and Information Sciences*, and *Signal Processing*.