Reversible Data Hiding in Encrypted Images Using Cipher-Feedback Secret Sharing

Zhongyun Hua, *Member, IEEE*, Yanxiang Wang, Shuang Yi, Yicong Zhou, *Senior Member, IEEE*, and Xiaohua Jia, *Fellow, IEEE*

***Abstract*— Reversible data hiding in encrypted images (RDH-EIs) has attracted increasing attention since it can protect the privacy of original images while exactly extracting the embedded data. In this paper, we propose an RDH-EI scheme with multiple data hiders. First, we introduce a cipher-feedback secret sharing (CFSS) technique using the cipher-feedback strategy of the Advanced Encryption Standard. Then, using the CFSS technique, we devise a new (r, n)-threshold $(r \leq n)$ RDH-EI scheme with multiple data hiders called CFSS-RDHEI. It can encrypt an original image into n encrypted images with reduced size using an encryption key and sends each encrypted image to one data hider. Each data hider can independently embed secret data into the encrypted image to obtain a marked encrypted image. The embedded data can be extracted from each marked encrypted image using the data hiding key, and the original image can be completely recovered from r marked encrypted images using the encryption key. Performance evaluations show that our CFSS-RDHEI scheme has a higher embedding rate and that its generated encrypted images are much smaller, while still being well protected, compared to existing secret sharing-based RDH-EI schemes.**

***Index Terms*— Reversible data hiding, encrypted image, cipher-feedback secret sharing, multiple data hiders.**

I. Introduction

WITH the rapid development of cloud computing, there is an increasing need for privacy protection. To protect the data privacy of holders while performing necessary operations, multimedia signal processing in the encrypted domain has attracted increasing attention [1]–[4]. In many applications, such as cloud storage, to protect the image contents, an image owner would like to encrypt the image to be a cipher image before uploading to the cloud server. After receiving the cipher image, the cloud server should embed some data into it for the purposes of storage management, image labeling, and so on. Thus, reversible data hiding in encrypted image (RDH-EI) technology has been developed [5], [6]. The image owner encrypts an image into an encrypted image. The data hider can embed secret data into the encrypted image without access to the image contents. The image and embedded data can be completely recovered using the related secret keys.

In 2008, Puech *et al.* first proposed an RDH-EI scheme [5], in which one bit can be embedded into an image block of a cipher image encrypted by the Advanced Encryption Standard (AES), and the embedded bit can be extracted by analyzing the local standard deviation of the image block. Several years later, Zhang proposed another new RDH-EI scheme by first encrypting an image using a stream cipher and then embedding additional data through the bit flipping technique [6]. Inspired by these works, many new RDH-EI schemes have been developed using different techniques [7]–[9]. All the RDH-EI schemes can be roughly classified into two types: vacating room after encryption (VRAE) [6]–[9] and reserving room before encryption (RRBE) [10]–[14]. The VRAE strategy first encrypts an original image and then embeds secret data into the encrypted image, while the vacated room for data embedding in the RRBE strategy is prereserved before encryption. Compared with the VRAE strategy, the RRBE strategy can take advantage of the spatial correlation of the original image and usually provides a larger embedding capacity than the VRAE strategy. Some typical examples of the RRBE strategy are as follows. Puteaux and Puech proposed an RDH-EI scheme by predicting the most significant bit (MSB) of the pixels [14]. This scheme can make full use of the local correlation of the plain image and achieves a high embedding capacity. Using this strategy, Yin *et al.* [15] designed a variable length MSB prediction method using Huffman coding, and Mohammadi *et al.* [16] extended the MSB prediction method by employing other significant bits.

To keep redundant information for data embedding, existing RDH-EI schemes using the VRAE or RRBE strategy usually encrypt the original images using some lightweight encryption methods such as block-based permutation [17], [18] and the XOR operation [3], [13] with a fixed key. Although these operations can achieve noise-like cipher images, the encrypted

Manuscript received 7 November 2021; revised 21 December 2021; accepted 22 December 2021. Date of publication 6 January 2022; date of current version 4 August 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62002301, Grant 62071142, and Grant 61732022; in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2021A1515011406; in part by the Natural Science Foundation of Chongqing under Grant cstc2019jcyj-msxmX0393 and Grant cstc2018jscx-msybX0115; in part by the University Innovation Research Group project of Chongqing (Evidence Science Technology Innovation and Application) under Grant CXQT21033; in part by the Shenzhen College Stability Support Plan under Grant GXWD20201230155427003-20200824210638001; and in part by the Research Committee at University of Macau under Grant MYRG2018-00136-FST. This article was recommended by Associate Editor S. Zhu. *(Corresponding author: Shuang Yi.)*

Zhongyun Hua, Yanxiang Wang, and Xiaohua Jia are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Shenzhen 518055, China (e-mail: huazyum@gmail.com; wyxhitsz@gmail.com; csjia@cityu.edu.hk).

Shuang Yi is with the 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).

Yicong Zhou is with the Department of Computer and Information Science, University of Macau, Macau, China (e-mail: yicongzhou@um.edu.mo).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCSVT.2022.3140974.

Digital Object Identifier 10.1109/TCSVT.2022.3140974

results cannot defend against many security attacks [19]. To enhance security, some new RDH-EI schemes have been developed using homomorphic encryption methods [2], [20]–[22]. In these schemes, an original image is encrypted by a homomorphic encryption algorithm, and the secret data are embedded into the encrypted image using homomorphic properties. However, since all homomorphic encryption algorithms have extremely high computational costs and data expansion, the encryption efficiency is greatly reduced, and the size of the generated encrypted image is expanded.

Recently, some RDH-EI schemes have been proposed using secret sharing techniques [23]–[26]. For the schemes in [23], [25], [26], the image owner encrypts an image into n encrypted images using secret sharing techniques and then uploads each encrypt image to one cloud server. Each cloud server can embed data into the encrypted image individually. When collecting r encrypted images, the original image can be completely recovered. The secret sharing-based RDH-EI technologies can be applied in multi-party cloud storage. For an important image, the image owner would like to share it into n parties and store each party in a cloud server. Only collecting at least r parties, the original image can be recovered. Thus, the image security is determined by r parties, which can resist the collusion attack of $r-1$ parties. In addition, even if some encrypted images are lost or destroyed, the original image can still be completely recovered if r ones are undamaged. In [23], Wu *et al.* first adopted the secret sharing technique for the RDH-EI scheme, in which the secret data are then embedded into the encrypted image using difference expansion or difference histogram shifting. Later, Chen *et al.* [24] proposed another secret sharing-based RDH-EI method. In this work, the data hider embeds secret data using difference expansion and the addition homomorphism. However, this work has only one data hider, and the embedding rate is not high. To achieve a larger embedding capacity, Chen *et al.* [25] proposed a new method with multiple data hiders. In the data hiding phase, each data hider can embed secret data by substituting one pixel in every n pixels. The embedding rate is reduced when the number of data hiders increases, and it is not applicable when $n > 7$.

In this paper, we propose a new RDH-EI scheme using a novel secret sharing technique. First, we present a novel cipher-feedback secret sharing (CFSS) technique. Using the CFSS, we further propose an RDH-EI scheme with multiple data hiders. In an (r, n)-threshold ($r \leq n$) scheme, the proposed RDH-EI can encrypt an original image into n encrypted images and sends each encrypted image to one data hider. Since $r-1$ pixels are processed in one sharing, each encrypted image is the $1/(r-1)$ size of the original image. A multi-MSB prediction method is used to embed secret data. The main contributions and novelty of this paper are summarized as follows:

1) We develop a cipher-feedback secret sharing (CFSS) technique applying the cipher-feedback strategy of AES to the polynomial-based secret sharing scheme. The CFSS can share images to be encrypted images with much smaller size than existing secret sharing techniques.
2) Using the CFSS, we propose an (r, n)-threshold ($r \leq n$) RDH-EI scheme called CFSS-RDHEI. Different from other secret sharing-based schemes [23]–[26] that independently share image pixels, our CFSS-RDHEI shares several pixels at one time using the cipher-feedback strategy.
3) A multi-MSB prediction method is developed to embed secret data, and it achieves a large embedding capacity.
4) Experimental results show that our CFSS-RDHEI can achieve a larger embedding rate and that its encrypted images are much smaller, while still being well protected, compared with existing secret sharing-based RDH-EI schemes.

The rest of this paper is organized as follows. Section II introduces the CFSS. Section III presents the CFSS-RDHEI in detail. Section IV simulates the CFSS-RDHEI and analyzes its embedding performance. Section V analyzes the security of the CFSS-RDHEI, and Section VI concludes this paper.

II. Cipher-Feedback Secret Sharing

In this section, we introduce the concept of the polynomial-based secret sharing technique, discuss its properties, and develop cipher-feedback secret sharing (CFSS).

A. Existing Polynomial-Based Secret Sharing

The (r, n)-threshold ($r \leq n$) polynomial-based secret sharing technique was first proposed by Shamir [27], where a dealer takes a secret as input and generates n shares for n parties (each party holds one share). A collection of any r shares can recover the original secret. The polynomial-based secret sharing technique can be used to solve many research issues, such as acting as a cryptographic scheme.

The fundamental theory of polynomial-based secret sharing is to construct a group of polynomial equations over a finite field F. Since n different positive integers within the finite field are required for generating n shares, n should be smaller than F. According to some known theorems, any r pairs $(x_i, y_i) \in F \times F$ with different $\{x_i\}$ can uniquely determine a polynomial f of degree $(k-1)$ such that $f(x_i) = y_i$ for $i \in \{1, 2, \cdots, k\}$.

The (r, n)-threshold secret sharing scheme to a secret S can be constructed as follows. First, let $\{x_1, x_2, \cdots, x_n\}$ be n different random positive integers that are all smaller than F. Then, an $r-1$ degree polynomial can be constructed by

$$f(x) = (a_0 + \sum_{k=1}^{r-1} a_k x^k) \mod F, \tag{1}$$

where the coefficients a_0 are taken from the secret S, $a_1 \cdots, a_{r-1}$ can be any random integer or the elements of the secret S and are all smaller than F. A party P_i holds $(x_i, f(x_i))$, where x_i is the identity and $f(x_i)$ is the share.

When any r shares with their identities have been collected (e.g., $(x_i, f(x_i)), i = 1, 2, \cdots, r$), one can reconstruct the polynomial f using the Lagrange interpolation as

$$L(x) = \sum_{j=1}^{r} (f(x_j) \times \prod_{\substack{0<k\leq r \\ k\neq j}} \frac{x - x_k}{x_j - x_k}) \mod F. \tag{2}$$

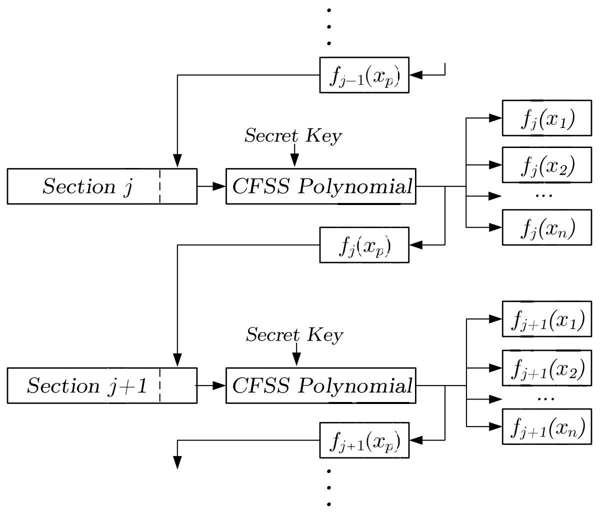


Fig. 1. Demonstration of the developed CFSS.

Then, the first coefficient a_0 can be obtained as $a_0 = L(0)$. Meanwhile, a completely expanded $L(x)$ can reveal the other coefficients $a_1, \cdots, a_{r-1}$ of the original polynomial as well. Then, the secret S can be obtained.

B. CFSS

In the original Shamir's secret sharing scheme [27], only the constant item a_0 is taken from the secret, and the other $r-1$ coefficients $a_1, \cdots, a_{r-1}$ are all randomly selected integers. In this scheme, each share has the same size as the original secret, and all the n shares have n times the size of the original secret, which causes large data expansion. In the Thien-Lin secret sharing scheme [28], all the r coefficients are taken from the secret. Then, each share has the $1/r$ size of the original secret, and all the n shares have the same size of the original secret. However, this scheme cannot achieve diffusion properties and does not use random numbers in the sharing process, leading to a limited security level.

To balance the share size and security, we introduce a new secret sharing scheme called CFSS using the cipher-feedback mode of AES. Fig. 1 shows the main concept of the CFSS. The last coefficient of the polynomial in the current sharing operation is taken from a random share (i.e., p is randomly selected among $1, 2, \cdots, n$ in each sharing) of the previous sharing result. The last coefficient in the first sharing operation is a random integer. Using this cipher-feedback strategy, any change in the secret image can completely change the obtained shares.

Then, the (r, n)-threshold CFSS can be constructed as follows. (1) Divide the secret S into several nonrepeated sections, and each section has $r-1$ elements; (2) Generate n different integers $\{x_1, x_2, \cdots, x_n\}$ from the secret key; (3) for the j-th section, an $r-1$ degree polynomial is constructed by

$$f_j(x) = (a_0 + \sum_{k=1}^{r-2} a_k x^k + f_{j-1}(x_p)x^{r-1}) \mod F, \quad (3)$$

where $a_0, \cdots, a_{r-2}$ are $r-1$ elements from the j-th section of S. $f_{j-1}(x_p)$ is a randomly selected previous sharing result, in which x_p ($p \in \{1, 2, \cdots, n\}$) is an input selected from the previous sharing and p is random in each sharing. For the first section $j = 1$, $f_0(x_p)$ can be any random integer satisfying $f_0(x_p) < F$. When $x \in \{x_1, x_2, \cdots, x_n\}$, n shares can be generated as $\{f_j(x_1), f_j(x_2), \cdots, f_j(x_n)\}$. Then, a party P_i holds the identity i and $f_j(x_i)$, which is the j-th element of the share.

The (r, n)-threshold CFSS scheme can achieve the following properties. (1) It can achieve diffusion property. When a bit is changed in the secret S, all the elements can be randomly changed by the cipher-feedback structure and the random integer $f_0(x_p)$. (2) The security can be further improved because $f_{j-1}(x_p)$ is randomly selected from previous n sharing results, and p can be different in each sharing. (3) Since each section has $r-1$ elements and generates one element for each share, the size of the obtained share can be greatly reduced to $1/(r-1)$ size of the original secret S.

III. CFSS-Based RDH-EI

In this section, we propose a new RDH-EI scheme using the CFSS called CFSS-RDHEI. Fig. 2 shows the structure of the CFSS-RDHEI. MSB prediction can make full use of the high adjacent pixel correlations of natural images. We call the image shares before data embedding the encrypted images and after data embedding the marked encrypted images.

In the CFSS-RDHEI, the initialization process first calculates the proportion of precisely predictable pixel number P_c and then determines the optimal level l using P_c. Then, the final side information is calculated. After that, the final side information and the $(8-l)$ LSBs of the original image are encrypted into n shares using the CFSS scheme with an encryption key. After embedding the n shares of the final side information into n encrypted images, the encrypted images are sent to data hiders. Once a data hider receives an encrypted image, he/she can embed additional data into the encrypted image by substituting the multi MSBs of available pixels to obtain a marked encrypted image. At the receiver side, one can extract the embedded data from a marked encrypted image and the related data hiding key and losslessly reconstruct the original image from at least r marked encrypted images and the encryption key.

A. Final Side Information Generation

CFSS-RDHEI uses multi-MSB substitution to embed data. The multi MSBs of pixels should be correctly predicted when reconstructing the original image. However, some pixels with special values cannot be correctly predicted. Their values and positions should be stored.

The median edge detector (MED) predictor is a high-performance predictor [29], and it is used in our method to predict the multi-MSBs of image pixels. Using the previous adjacent pixels, the MED predictor can predict the current pixel value. The expression of the MED predictor used in the proposed method is shown as follows:

$$\tilde{x}_{i,j} = \begin{cases} b, & \text{for } j = 1, i \neq 1; \\ c, & \text{for } i = 1, j \neq 1; \\ \max(b, c), & \text{for } a \leq \min(b, c); \\ \min(b, c), & \text{for } a \geq \max(b, c); \\ b + c - a, & \text{otherwise,} \end{cases} \quad (4)$$

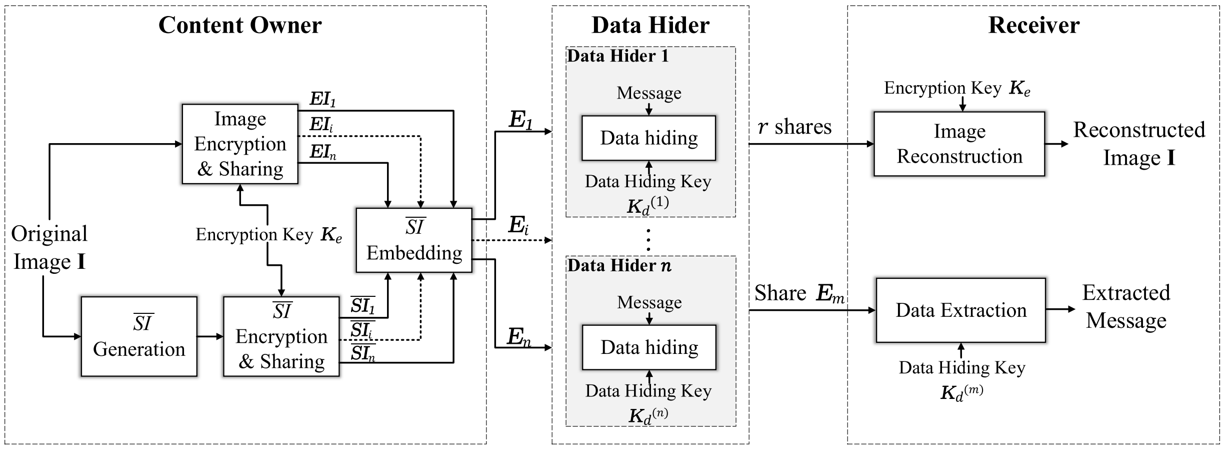


Fig. 2. Overview of the CFSS-RDHEI.

where a, b, and c are the upper-left, upper, and left adjacent pixels of $x_{i,j}$, respectively. Note that the first pixel $x_{1,1}$ is not processed.

Since different images may have different smoothness properties, their optimal embedding capacity can be achieved at different levels of MSBs. To achieve a large embedding rate for each image, an optimal level l for l-MSB prediction is first determined. For simplicity, the proportion of pixels that can be precisely predicted is used to detect l. For an image of size $M \times N$, this proportion P_c is calculated as

$$P_c = \frac{\sum\limits_{(i,j\in[1,M]\times[1,N])\cap(i,j)\neq(1,1)} P(i,j)}{M \times N}, \tag{5}$$

where $P(i, j)$ indicates whether pixel $x_{i,j}$ can be correctly predicted, and it is defined as

$$P(i,j) = \begin{cases} 1 & \text{for } x_{i,j} = \tilde{x}_{i,j}; \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

Since P_c is an image feature, the relationship between l and P_c can be learned from a large number of natural images. According to experience-based learning on the 10000 natural images in the BOW-2 image dataset [30], l is set as

$$l = \begin{cases} 4 & \text{for } P_c \leq 0.063; \\ 5 & \text{for } 0.063 < P_c \leq 0.102; \\ 6 & \text{for } P_c > 0.102. \end{cases} \tag{7}$$

Denote $x_{i,j}^{lMSB}$ and $\tilde{x}_{i,j}^{lMSB}$ as the l-MSB values of the original and predicted pixels, respectively. Their values can be calculated as

$$\begin{cases} x_{i,j}^{lMSB} = (x_{i,j} - (x_{i,j} \mod 2^{8-l})) \times 2^{-(8-l)}, \\ \tilde{x}_{i,j}^{lMSB} = (\tilde{x}_{i,j} - (\tilde{x}_{i,j} \mod 2^{8-l})) \times 2^{-(8-l)}. \end{cases} \tag{8}$$

The l-MSB prediction error between $x_{i,j}^{lMSB}$ and $\tilde{x}_{i,j}^{lMSB}$ is calculated as

$$Lpe = x_{i,j}^{lMSB} - \tilde{x}_{i,j}^{lMSB}. \tag{9}$$

Because a natural image usually has high adjacent pixel correlation, a pixel can be predicted using its adjacent pixels. Thus, when applying an efficient predictor to a natural image, the distribution of the prediction errors is a Laplacian-like distribution with a location parameter of 0 [31]. If the l MSBs of a pixel can be completely predicted, its l-MSB prediction error Lpe is 0. If the difference between $x_{i,j}^{lMSB}$ and $\tilde{x}_{i,j}^{lMSB}$ of a pixel is 1, its l-MSB prediction error Lpe is -1 or 1. Similarly, if the difference between $x_{i,j}^{lMSB}$ and $\tilde{x}_{i,j}^{lMSB}$ of a pixel is 2, its l-MSB prediction error Lpe is -2 or 2. Fig. 3 displays the histograms of the l-MSB prediction errors of two commonly used natural images. It can be seen that the numbers of l-MSB prediction errors -1, 0, and 1 are the largest. To fully utilize this statistical characteristic, we propose a new method to store the extra information. First, a location map is used to mark the pixels. A pixel is considered a predictable pixel if its prediction error $Lpe \in \{-1, 0, 1\}$ and its related position on the original location map ***OLM*** is marked as 0; otherwise, it is an unpredictable pixel and its position on the ***OLM*** is marked as 1. Since most pixels are predictable pixels, the location map has high data redundancy (e.g., more than 90% of elements of the location map are 0 in Fig. 3). After being compressed using the extended run-length coding in [32], a compressed location map ***LM*** is generated as a part of the side information. Since there are three types of predictable pixels, namely, the l-MSB prediction error Lpe is -1, 0, or 1, we encode these three types of prediction errors using three codes, namely, 10, 0, and 11, respectively. Then, the prediction error codes ***Lpes*** can be generated by

$$\boldsymbol{Lpes} = \begin{cases} \boldsymbol{Lpes} \parallel 0 & \text{for } Lpe = 0; \\ \boldsymbol{Lpes} \parallel 10 & \text{for } Lpe = -1; \\ \boldsymbol{Lpes} \parallel 11 & \text{for } Lpe = 1. \end{cases} \tag{10}$$

where '$\parallel$' stands for concatenation symbol.

The l MSBs of all the unpredictable pixels should be stored. According to the median edge detector used in Eq. (4), the first pixel cannot be processed, and its l MSBs should also be stored. The l MSBs of the first pixel and all the unpredictable pixels are combined to obtain a binary sequence ***B***, which is another part of the side information.

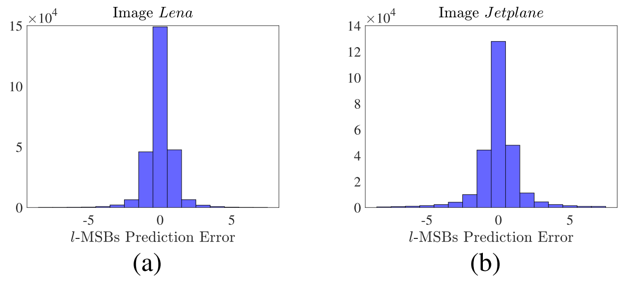


Fig. 3. Histograms of the l-MSB prediction errors in different test images. (a) Image *Lena* with $l = 5$; (b) image *Jetplane* with $l = 6$.

TABLE I

Pixel Concatenation Strategies in the $(8 - l)$-LSBs Image for Different Optimal Level l

Optimal level l	Concatenated pixel number	Largest prime F
4	2	251
5	2	61
6	4	251

After processing all the l MSBs of the image pixels, the $(8-l)$ LSBs of the image are encrypted using the proposed CFSS method. To achieve a high efficiency, several adjacent pixels are concatenated to form a larger number, and the modular coefficient F in Eq. (3) is set as the largest prime integer that is smaller than or equal to the largest concatenated element. Table I shows the pixel concatenation strategies. For example, if $l = 4$, two adjacent pixels are concatenated, and the largest concatenated value is $2^8 - 1$, namely, 255. Then, set $F = 251$.

Then, for these concatenated elements $I^c_{i,j}$ whose values are larger than or equal to F, we change their values to $F-1$. The modified information should be stored to recover the original pixels. The reference information $\boldsymbol{T}$ encodes the difference between $F-1$ and the concatenated elements $I^c_{i,j}$ that is larger than or equal to F. For example, if $F = 251$, the reference information $\boldsymbol{T}$ can be generated as

$$\boldsymbol{T} = \begin{cases} \boldsymbol{T} \parallel 101 & \text{for } I^c_{i,j} = 255; \\ \boldsymbol{T} \parallel 100 & \text{for } I^c_{i,j} = 254; \\ \boldsymbol{T} \parallel 011 & \text{for } I^c_{i,j} = 253; \\ \boldsymbol{T} \parallel 010 & \text{for } I^c_{i,j} = 252; \\ \boldsymbol{T} \parallel 001 & \text{for } I^c_{i,j} = 251; \\ \boldsymbol{T} \parallel 000 & \text{for } I^c_{i,j} = 250; \\ \boldsymbol{T} & \text{otherwise.} \end{cases} \tag{11}$$

Note that only two bits are required to encode the difference between $F - 1$ and $I^c_{i,j}$ when $F = 61$.

Then, the side information $\boldsymbol{SI}$ consists of the compressed location map $\boldsymbol{LM}$, l-MSB prediction error codes $\boldsymbol{Lpes}$, l MSBs of the first and unpredictable pixels $\boldsymbol{B}$, and the reference information $\boldsymbol{T}$, namely, $\boldsymbol{SI} = \boldsymbol{LM} \parallel \boldsymbol{Lpes} \parallel \boldsymbol{B} \parallel \boldsymbol{T}$. For an image of size $M \times N$, the maximum lengths of $\boldsymbol{LM}$, $\boldsymbol{Lpes}$ and $\boldsymbol{B}$ are MN, $2MN$ and lMN, respectively. For $\boldsymbol{T}$, when $l = 4$ with $F = 251$, it has the theoretical maximum length $3MN/2$. Thus, we use $L_{\boldsymbol{LM}} = \lceil \log_2(MN) \rceil$ bits, $L_{\boldsymbol{Lpes}} = \lceil \log_2(2MN) \rceil$ bits, $L_{\boldsymbol{B}} = \lceil \log_2(lMN) \rceil$ bits and $L_{\boldsymbol{T}} = \lceil \log_2(3MN/2) \rceil$ bits to record their lengths. In addition, since $\boldsymbol{LM}$ is obtained by compressing the location map using the method in [32] and this method contains four possible rearrangement types, another 2 bits are used to record the rearrangement type. All these additional bits are placed in front of $\boldsymbol{SI}$.

The final side information $\overline{\boldsymbol{SI}}$ consists of the $\boldsymbol{SI}$ and the bit sequence $\boldsymbol{Tsi}$ for preprocessing $\boldsymbol{SI}$, which is discussed in Section III-B.2. Table II shows the components of the final side information. Since the total length of $\overline{\boldsymbol{SI}}$ can be obtained in the recovery stage, the length of $\boldsymbol{Tsi}$ does not need to be recorded. After being encrypted using the CFSS, each share of $\overline{\boldsymbol{SI}}$ is embedded into the vacated room of each encrypted image.

B. (r, n)-Threshold Sharing and Encryption

On the content owner side, the $(8 - l)$ LSBs of the original image are encrypted into n encrypted images using CFSS with an encryption key. To losslessly recover the whole image I, the final side information $\overline{\boldsymbol{SI}}$ should be used in the recovery process and embedded into the encrypted images. We also encrypt $\overline{\boldsymbol{SI}}$ into n shares and then embed each share into one encrypted image. After being embedded with several parameters and the $\overline{\boldsymbol{SI}}$ share, the n encrypted images are separately sent to n data hiders.

1) (r, n)-Threshold Sharing: Suppose that the size of the image I is $M \times N$. For the $(8 - l)$ LSBs of the image, we concatenate two or more pixels to obtain larger elements according to Table I. Suppose that the concatenated image I^c has the size $M' \times N'$. A permutation is performed to it. To ensure security, the permutation sequence is generated by the encryption key and the sum of the pixel values in image I^c. Then, the permuted image can be divided into $\left\lceil \frac{M' \times N'}{r-1} \right\rceil$ sections, and each section has $r - 1$ pixels. If the last section has fewer than $r-1$ elements, random integers are generated for padding. Using the $r - 1$ pixels $\{x_0, \cdots, x_{r-2}\}$ in the j-th section and one previous sharing result, one can construct n polynomials of $r - 1$ degree for section j as

$$\begin{cases} f_j(q_j) = (\sum_{i=0}^{r-2} x_i q_j^i + f_{j-1}(q_{j-1} + p) q_j^{r-1}) \bmod F; \\ f_j(q_j + 1) = (\sum_{i=0}^{r-2} x_i (q_j + 1)^i \\ \qquad + f_{j-1}(q_{j-1} + p)(q_j + 1)^{r-1}) \bmod F; \\ \quad \vdots \\ f_j(q_j + n - 1) = (\sum_{i=0}^{r-2} x_i (q_j + n - 1)^i \\ \qquad + f_{j-1}(q_{j-1} + p)(q_j + n - 1)^{r-1}) \bmod F; \end{cases} \tag{12}$$

where q_j is the j-th element of $\mathbf{Q}$, which is a pseudorandom integer sequence generated by the encryption key and $q_j \le F - n$, and $f_{j-1}(q_{j-1} + p)$ is a randomly selected previous sharing result. $f_{j-1}(q_{j-1} + p)$ can be any random integer within range $[0, F)$ when $j = 1$.

The n outputs of Eq. (12), $f_j(q_j), f_j(q_j+1), \cdots, f_j(q_j + n - 1)$, are the j-th pixel of the n encrypted images. Note that each output should be separated into multiple $(8-l)$ bits of pixels, which is opposite to the concatenation strategies in Table I. After all the sections are processed, the image constructed by the $(8 - l)$ LSBs of the original image is encrypted into n encrypted images. Finally, the l MSBs of

TABLE II

COMPONENTS OF THE FINAL SIDE INFORMATION $\overline{\boldsymbol{SI}}$

Symbols	Information	Bit lengths
-	Rearrangement type for generating $\boldsymbol{LM}$	2
$L_{\boldsymbol{LM}}$	Length of $\boldsymbol{LM}$	$\lceil \log_2(MN) \rceil$
$L_{\boldsymbol{Lpes}}$	Length of $\boldsymbol{Lpes}$	$\lceil \log_2(2MN) \rceil$
$L_{\boldsymbol{B}}$	Length of $\boldsymbol{B}$	$\lceil \log_2(lMN) \rceil$
$L_{\boldsymbol{T}}$	Length of $\boldsymbol{T}$	$\lceil \log_2(3MN/2) \rceil$
$\boldsymbol{LM}$	Compressed location map	-
$\boldsymbol{Lpes}$	Prediction error codes	-
$\boldsymbol{B}$	l MSBs of the first and unpredictable pixels	-
$\boldsymbol{T}$	Reference information	-
$\boldsymbol{Tsi}$	Bits for preprocessing $\boldsymbol{SI}$	-

the encrypted images are filled using random bits to obtain n completely encrypted images.

2) Side Information Embedding: As discussed in Section III-A, the final side information $\overline{\boldsymbol{SI}}$ should be embedded into the n encrypted images before being sent to the data hider. To equally separate the final side information into the n encrypted images, we also encrypt it into n shares using the CFSS. Only by owning r shares can one recover the final side information.

The final side information $\overline{\boldsymbol{SI}}$ consists of the side information $\boldsymbol{SI}$ and the bit sequence $\boldsymbol{Tsi}$ for preprocessing $\boldsymbol{SI}$. First, divide the side information $\boldsymbol{SI}$ into 7-bit streams and transform each 7-bit stream, $b_0b_1b_2b_3b_4b_5b_6$, to decimal value. Thus, the side information $\boldsymbol{SI}$ contains $\lceil \frac{SI}{7} \rceil$ decimal values. The F in sharing the side information is set as 127. Then, the elements in $\boldsymbol{SI}$ with a value of 127 should be set as 126, $\boldsymbol{Tsi}$ is used to record the location of elements with a value of 127, and it can be generated as

$$\boldsymbol{Tsi} = \begin{cases} \boldsymbol{Tsi} \parallel 1 & \text{for } \boldsymbol{SI}_k = 127; \\ \boldsymbol{Tsi} \parallel 0 & \text{for } \boldsymbol{SI}_k = 126; \\ \boldsymbol{Tsi} & \text{otherwise.} \end{cases} \tag{13}$$

Since the elements in the original $\boldsymbol{SI}$ with values 127 and 126 are only a small probability, $\boldsymbol{Tsi}$ is very short. To avoid the appearance of 127 in $\boldsymbol{Tsi}$, we add a '0' after every six bits into $\boldsymbol{Tsi}$. Then, the maximum value in $\boldsymbol{Tsi}$ is 126. Finally, the $\boldsymbol{Tsi}$ is appended to the end of the $\boldsymbol{SI}$ to obtain the final $\overline{\boldsymbol{SI}}$.

Now, the total number of decimal values in the final side information $\overline{\boldsymbol{SI}}$ is $\lceil \frac{\overline{SI}}{7} \rceil$, and all the values are smaller than 127. Then, these decimal values are divided into $\lceil \frac{\overline{SI}}{7(r-1)} \rceil$ sections, and each section has $r-1$ elements. Another pseudorandom integer sequence $\tilde{\mathbf{Q}}$ of length $\lceil \frac{\overline{SI}}{7(r-1)} \rceil$ is generated using the encryption key. All the elements in $\tilde{\mathbf{Q}}$ are not larger than $127-n$, as the modular coefficient is set as $F = 127$. Using the pseudorandom integer sequence $\tilde{\mathbf{Q}}$, one can encrypt the final side information $\overline{\boldsymbol{SI}}$ to be n shares, and the sharing process is the same as that in Eq. (12). Then, n $\overline{\boldsymbol{SI}}$ shares are generated.

Since the l MSBs of the encrypted images are reserved for data embedding, we embed each $\overline{\boldsymbol{SI}}$ share and some additional bits into each encrypted image by substituting its l MSBs. These additional bits include the codes of the optimal level l, parameter r, identity of the encrypted image, the original image height M and width N, and the length of the $\overline{\boldsymbol{SI}}$ share. We use 3 bits, 8 bits, 8 bits and 40 bits to present the optimal level l, parameter r, identity of the encrypted image, and the original image height M and width N, respectively. Each $\overline{\boldsymbol{SI}}$ share contains $\lceil \frac{\overline{SI}}{7(r-1)} \rceil$ decimal values. Its length is $\lceil \frac{\overline{SI}}{7(r-1)} \rceil \times 7$ bits. Considering the theoretical maximum lengths of $\overline{\boldsymbol{SI}}$ and smallest r, it is sufficient to use $\lceil \log_2 MN \rceil + 4$ bits to present the length of each $\overline{\boldsymbol{SI}}$ share.

We first successively embed the above 3 bits, 8 bits, 8 bits, 40 bits and $\lceil \log_2 MN \rceil + 4$ bits to present the optimal level l, parameter r, identity of the encrypted image, original image height M and width N, and length of each $\overline{\boldsymbol{SI}}$ share, respectively, and then embed the $\overline{\boldsymbol{SI}}$ share. Note that the 3 bits for encoding the optimal level l should be embedded into the fixed positions, and we embed them into the most significant bit of the first 3 pixels. All these additional bits and $\overline{\boldsymbol{SI}}$ share the form of the overhead $\boldsymbol{OH}$ of the encrypted image.

C. Data Hiding

For one data hider who has the i-th encrypted image $\mathbf{E}_i$, he/she can directly embed secret data to the encrypted domain without knowing the encryption key or the contents of the original image. The secret data are first encrypted by an existing cryptographic algorithm (e.g., DES or AES) using a data hiding key K_d. The optimal level l is first extracted from the most significant bit of the first 8 pixels. Then, the available embedding space can be found by checking the length of overhead $\boldsymbol{OH}$. The data hider can embed the encrypted secret data by substituting the l MSBs of the pixels.

D. Data Extraction and Image Reconstruction

Since the data embedding and image encryption are independent, the data extraction and image reconstruction are separable. Using the related data hiding key, a receiver can extract the embedded data from one marked encrypted image. Using the encryption key, a receiver can reconstruct the original image from r marked encrypted images.

1) Data Extraction With Data Hiding Key: For each marked encrypted image, a receiver can extract the embedded data using the related data hiding key from the following steps.

- ***Step 1***: Extract the most significant bit of the first 8 pixels to obtain the optimal level l.
- ***Step 2***: According to the embedding strategy, the data are embedded in the l MSBs of the marked encrypted image. Therefore, the l MSBs of all the pixels are extracted.
- ***Step 3***: From these extracted data, the receiver first obtains the overhead $\boldsymbol{OH}$ of the encrypted image.
- ***Step 4***: After excluding the overhead $\boldsymbol{OH}$, the remaining embedded data are the encrypted secret data. The secret data can be obtained by decrypting the encrypted secret data using the data hiding key.

2) Image Reconstruction With Encryption Key: With r marked encrypted images, a receiver can reconstruct the original image using the encryption key from the following steps.

- ***Step 1***: First, the receiver extracts the optimal level l, parameter r, identity of each encrypted image, original image height M and width N, and r shares of $\overline{\boldsymbol{SI}}$ from these marked encrypted images. Generate the same pseudorandom integer sequence $\tilde{\mathbf{Q}}$ using the encryption key and then perform the reverse CFSS to recover the final side information $\overline{\boldsymbol{SI}}$. Then, the side information $\boldsymbol{SI}$ can be obtained. Finally, the compressed location map $\boldsymbol{LM}$ and its compression type, l-MSB prediction error codes $\boldsymbol{Lpes}$, binary sequence $\boldsymbol{B}$, and reference information $\boldsymbol{T}$ can be recovered.
- ***Step 2***: Generate the same pseudorandom integer sequence $\mathbf{Q}$ using the encryption key. Combine the $(8-l)$ LSBs of the encrypted images according to Table I and perform the reverse CFSS to recover the modified concatenated image $\tilde{I^c}$.
- ***Step 3***: Recover the original concatenated image I^c using sequence $\boldsymbol{T}$. If a pixel value in $\tilde{I^c}$ is $F-1$, three bits are extracted from $\boldsymbol{T}$ when $F = 251$ (two bits are extracted when $F = 61$) and added to the pixel. After all pixels have been processed, a reverse permutation is followed to obtain the original concatenated image I^c. Then, after pixel separation according to Table I, the $(8-l)$ LSBs of the original image I are reconstructed.
- ***Step 4***: Finally, MSB prediction is performed to recover the l MSBs of the original image I using $\boldsymbol{LM}$, $\boldsymbol{Lpes}$, and $\boldsymbol{B}$. The uncompressed location map is obtained from $\boldsymbol{LM}$. The pixels are scanned from left to right and from top to bottom. First, l bits are extracted from $\boldsymbol{B}$ to recover the first pixel to initialize the prediction. Then, for each pixel, if its marker in the location map is '1', it is an unpredictable pixel, and its l MSBs are directly extracted from $\boldsymbol{B}$. If its marker in the location map is '0', it is a predicable pixel. We first calculate its prediction value $\tilde{x}_{i,j}^{lMSB}$ using the MED predictor as mentioned in Section III-A and obtain a prediction error code from $\boldsymbol{Lpes}$. Then, using the $\tilde{x}_{i,j}^{lMSB}$ and error code $\boldsymbol{Lpes}_k$, the l MSBs of the original pixel can be recovered as

$$x_{i,j}^{lMSB} = \begin{cases} \tilde{x}_{i,j}^{lMSB} & \text{for } \boldsymbol{Lpes}_k = 0; \\ \tilde{x}_{i,j}^{lMSB} - 1 & \text{for } \boldsymbol{Lpes}_k = 10; \\ \tilde{x}_{i,j}^{lMSB} + 1 & \text{for } \boldsymbol{Lpes}_k = 11. \end{cases} \quad (14)$$

  The original pixel can be reconstructed by combining the recovered l MSBs and $(8-l)$ LSBs. After reconstructing all the pixels, the original image I can be obtained.

Traditional RDH-EI schemes focus on one data hider. The original image cannot be recovered if only one encrypted image is lost or damaged. The proposed CFSS-RDHEI scheme can solve this problem because it encrypts the original image into n shares, and one can completely recover the original image from any r $(r \leq n)$ shares. Thus, even when some shares are lost or damaged, the original image can still be completely recovered as long as r shares are undamaged.

E. A Numeral Example

To better show the main procedures of the proposed CFSS-RDHEI scheme, we provide a numeral example with a $(3, 4)$-threshold, which indicates that an image is encrypted into four encrypted images and that one can reconstruct the original image using three of them. Fig. 4 demonstrates all operations of image sharing, data hiding and image reconstruction. Suppose that the optimal level l is 4; then, two pixels in the $(8-l)$ LSBs of the image are concatenated according to Table I. Thus, four pixels of the original image are required in one sharing operation, and they are supposed to be $(143, 151, 147, 145)$ in the j-th section. In addition, the j-th element in the pseudorandom integer sequence $\mathbf{Q}$ is $q_j = 68$.

For the content owner, side information is first generated. Assume that the prediction values of these four pixels using the MED predictor are $(143, 151, 143, 127)$. Then, their l-MSB prediction errors Lpe are $(0, 0, 1, 2)$. Since a pixel is predictable if its $Lpe \in \{-1, 0, 1\}$, the first three pixels are predictable pixels, while the last pixel is an unpredictable pixel. Then, the original location map $\boldsymbol{OLM}$ = '0001' and the encoded location map $\boldsymbol{LM}$ = '00001'. After encoding the Lpes of the first three pixels using Eq. (10), we can obtain that the prediction error codes $\boldsymbol{Lpes}$ = '0011'. The l MSBs of the last pixel are recorded in $\boldsymbol{B}$, and thus, we can obtain $\boldsymbol{B}$ = '1001'.

Then, we obtain the $(8-l)$-LSB values of these four pixels as $(15, 7, 3, 1)$ and concatenate them to two elements $\hat{x}_0 = 247$ and $\hat{x}_1 = 49$. Since $\hat{x}_0 < 250$ and $\hat{x}_1 < 250$, we can obtain $\boldsymbol{T}$ = '' according to Eq. (11). Using $\hat{x}_0$, $\hat{x}_1$, q_j, and one previous sharing result $f_{j-1}(q_{j-1} + p) = 91$, one can obtain the secret sharing polynomials:

$$f_j(q_j + k) = (\hat{x}_0 + \hat{x}_1(q_j + k) + f_{j-1}(q_{j-1} + p)(q_j + k)^2) \times \text{mod } F, \quad (15)$$

where $k \in \{0, 1, 2, 3\}$. For the modular coefficient $F = 251$, the four outputs $f_j(q_j), f_j(q_j + 1), f_j(q_j + 2)$ and $f_j(q_j + 3)$ can be obtained and separated to be the $(8-l)$-LSBs of two pixels in each encrypted image, which are $(10, 13)$, $(8, 11)$, $(2, 4)$ and $(7, 3)$, respectively. The l MSBs of these pixels are filled using random bits. Then, four shares with two pixels can be obtained. The $\boldsymbol{SI} = \boldsymbol{LM} \parallel \boldsymbol{Lpes} \parallel \boldsymbol{B} \parallel \boldsymbol{T}$ = '0000100111001'. According to Eq. (13), $\boldsymbol{Tsi}$ = ''. Then, each component in Table II is generated, and the final side information can be obtained.

For each data hider, he/she can embed l bits into each pixel by substituting its l MSBs. As shown in Fig. 4, the four images become $(106, 237)$, $(200, 91)$, $(114, 68)$ and $(167, 35)$ after embedding secret data into their l MSBs.

For the receiver, suppose that the three marked encrypted images with pixels $(106, 237)$, $(200, 91)$, and $(114, 68)$ are collected. First, calculate the $(8-l)$-LSBs of the three shares as $(10, 13)$, $(8, 11)$ and $(2, 4)$. Then, they are concatenated to three elements as $(173, 139, 36)$. Perform the reverse CFSS using the same integer $q_j = 68$ to recover the $(8-l)$-LSBs of the four pixels as $(15, 7, 3, 1)$. Perform the MED predictor to generate the prediction values $(143, 151, 143, 127)$ and obtain the l MSBs of these prediction values as $(8, 9, 8, 7)$, respectively. Using the side information $\boldsymbol{OLM}$, $\boldsymbol{Lpes}$ and $\boldsymbol{B}$, the l MSBs of four pixels can be recovered as $(8, 9, 9, 9)$. The

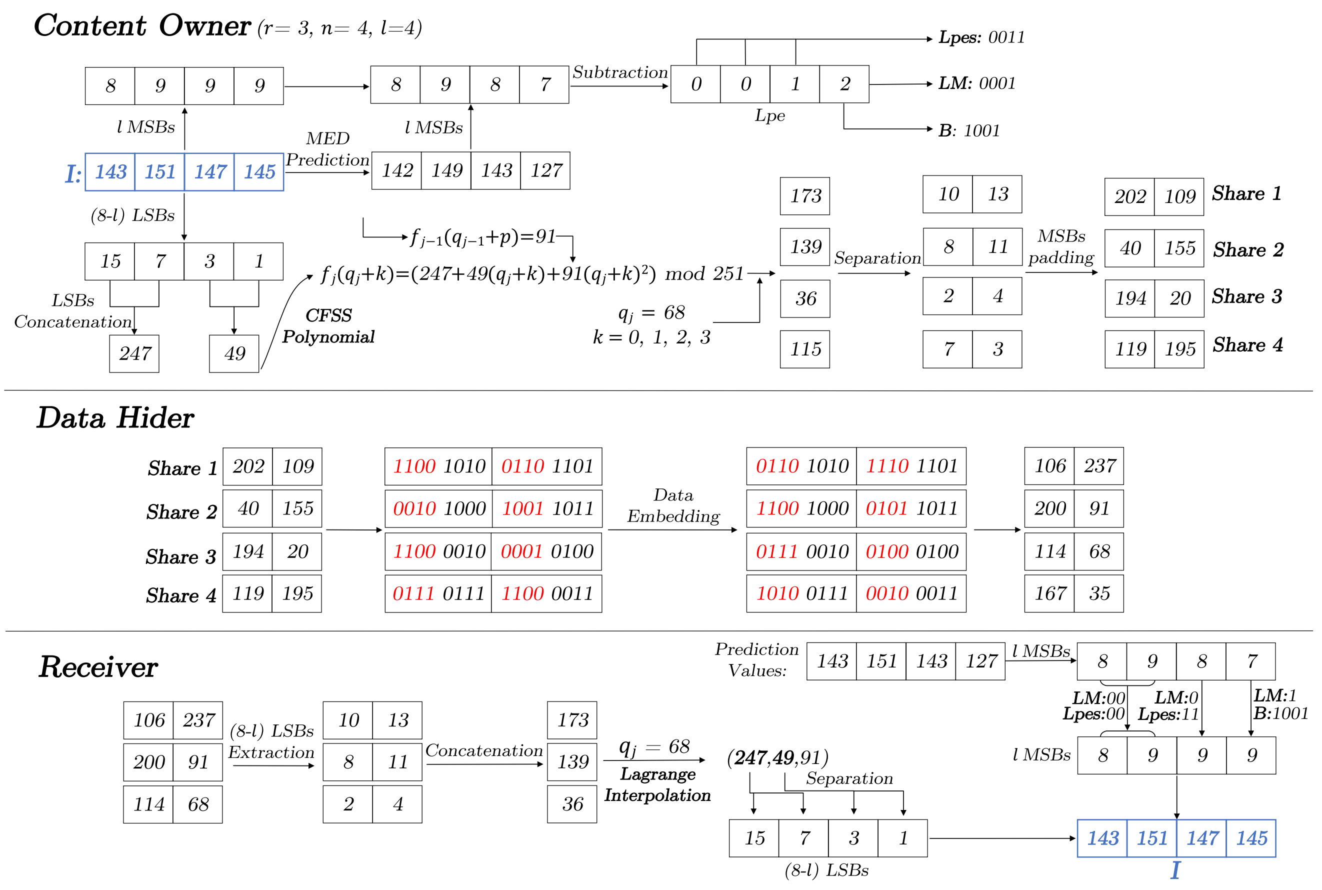


Fig. 4. A numeral example of the CFSS-RDHEI scheme with a (3, 4)-threshold.

original four pixels (143, 151, 147, 145) can be recovered by combining the l MSBs and $(8 - l)$ LSBs.

F. Discussion

The proposed CFSS-RDHEI scheme can well consider the adjacent pixel correlation and encrypt image pixels with high security. It can achieve the following advantages.

1) CFSS-RDHEI can encrypt an original image to be n encrypted images following the cipher-feedback strategy of AES.
2) CFSS-RDHEI has much lower data expansion than other similar methods, and each encrypted image has only the $1/(r - 1)$ size of the original image.
3) A multi-MSB prediction method is used to embed secret data such that CFSS-RDHEI can obtain a large embedding rate for many natural images.
4) Since random integers are used in the sharing process, the CFSS-RDHEI is a nondeterministic system. This provides a strong ability to resist many potential attacks.
5) Since each step of the encryption and embedding processes is reversible, our scheme can completely recover the embedded data and original image without data distortion.
6) Since the data embedding and image encryption are independent, the data extraction and image reconstruction are separable.

IV. Simulation Results and Performance Analysis

In this section, we simulate the CFSS-RHDEI scheme and compare its performance with existing secret sharing-based RDH-EI schemes. Eight commonly used grayscale images with different features are selected as the test images and are shown in Fig. 5. The embedding rate is highly determined by the smoothness of the image and has little relation to the image size. To provide an objective experimental result, we directly use the original image size 512×512 in our experiments. The secret data to be embedded are randomly generated. They are encrypted using encryption standards such as AES before being embedded.

A. Simulation Results

Fig. 6 shows the simulation results of the (3, 4)-threshold image sharing for the test image *Lena*. According to Eq. (7), the optimal level for the image *Lena* is $l = 5$. Fig. 6(a) is the original *Lena* image, and Figs. 6(b)-(e) are four encrypted images. All the encrypted images are noise-like and do not contain any information of the original information from the statistical characteristics. Figs. 6(f)-(i) are four marked encrypted images obtained by embedding secret data into Figs. 6(b)-(e), respectively. It is necessary to evaluate the visual imperceptibility of the images containing secret data for reversible data hiding (RDH) technologies, since the data are embedded into the plain-image and different embedding

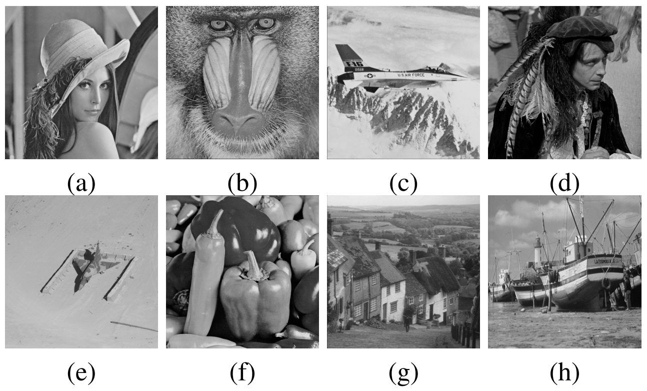


Fig. 5. The eight test images of size 512 × 512. (a) *Lena*; (b) *Baboon*; (c) *Jetplane*; (d) *Man*; (e) *Airplane*; (f) *Peppers*; (g) *Goldhill*; (h) *Boat*.

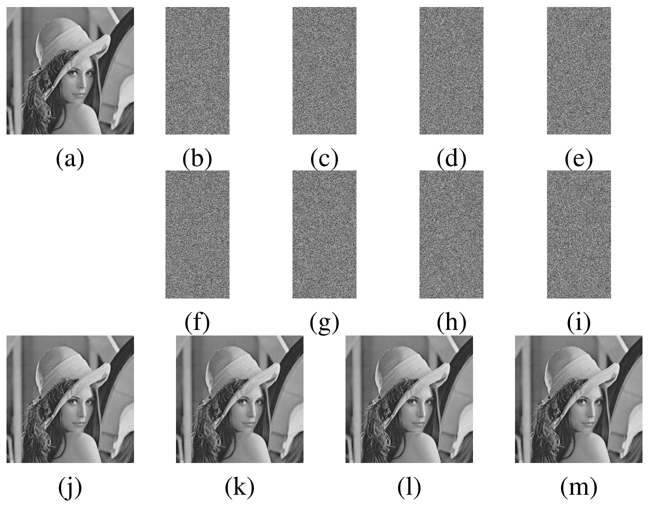


Fig. 6. Simulation results of the CFSS-RDHEI with $(3, 4)$-threshold image sharing. (a) Original Image *Lena* of size 512 × 512; (b)-(e) four encrypted images; (f)-(i) four marked encrypted images; (j) reconstructed image with PSNR $= +\infty$ dB from (f), (g), and (h); (k) reconstructed image with PSNR $= +\infty$ dB from (f), (g), and (i); (l) reconstructed image with PSNR $= +\infty$ dB from (f), (h), and (i); (m) reconstructed image with PSNR $= +\infty$ dB from (g), (h), and (i).

strategies may cause different visual effects to the plain-image. However, in our RDH-EI scheme, the image owner encrypts original image into encrypted image, while the data hider can directly embed some secret data into the encrypted image. The receiver can separately extract the secret data and recover the original image. Since the encrypted data are directly embedded into the encrypted image, both the encrypted image and marked encrypted image are always random-like. Thus, the embedded data are always visually imperceptible in the marked encrypted image and the marked encrypted image will not reveal any information of the original image and the secret data. The contents of the original image and embedded data are imperceptible by attackers. Figs. 6(j)-(m) are the four reconstructed images from three of the four marked encrypted images using the encryption key. Their PSNR (peak signal to noise ratio) values are $+\infty$ dB, which means that they are exactly the same as the original image in Fig. 6(a). Therefore, our CFSS-RDHEI is a lossless scheme. It shows that each marked encrypted image is $1/(r-1)$ of the original image. Thus, when $r > 2$, the marked encrypted image has a smaller size than the original image.

B. Data Expansion

According to the discussions in [23], data expansion for the whole scheme is the ratio between the bits of all the marked encrypted images and the bits of the original image, while the data expansion for each data hider is the ratio between the bits of each marked encrypted image and the bits of the original image. The data expansion for each data hider in traditional RDH-EI schemes is 1. Since the encrypted results of homomorphic encryption are larger than the original data and multiple images are generated in a secret share scheme, the total size of the marked encrypted images is often larger than the original image in these homomorphic encryption-based RDH-EI schemes and secret sharing-based RDH-EI schemes.

In our CFSS-RDHEI with an (r, n)-threshold scheme, a secret image is encrypted into n encrypted images, and each encrypted image has the $1/(r-1)$ size of the original image. Since the data embedding operation cannot cause data expansion, the total size of the n marked encrypted images is the $n/(r-1)$ size of the original image. Since each data hider keeps one marked encrypted image, the expansion rate for each data hider is $1/(r-1)$. Table III lists the data expansion comparison of different secret sharing-based RDH-EI schemes. The data expansion for the whole scheme is n in Wu *et al.* [23], Chen *et al.* [25] and Qin *et al.* [26] methods and is $n/(r-1)$ in our CFSS-RDHEI. Note that the data expansion for the whole scheme is always 1 in Chen *et al.* [24] method because it has only one encrypted image. In addition, the data expansion for each data hider is only $1/(r-1)$ in CFSS-RDHEI, which is much smaller than that of the other four methods. In general, our proposed scheme causes much less data expansion than other secret sharing-based RDH-EI schemes.

TABLE III
DATA EXPANSION COMPARISON OF THE SECRET SHARING-BASED RDH-EI SCHEMES

Methods	Expansion rate for the whole scheme	Expansion rate for each data hider
Chen *et al.* [24]	1	1
Wu *et al.* [23]	n	1
Chen *et al.* [25]	n	1
Qin *et al.* [26]	n	1
CFSS-RDHEI	$n/(r-1)$	$1/(r-1)$

C. Embedding Rate

The embedding performance of concern to a data hider is how many data points can be embedded into the received encrypted image. The effective embedding capacity indicates the maximum number of secret data that can be embedded into the encrypted image by the data hider. It is usually evaluated by the embedding rate ER. The ER is calculated as,

$$ER = \frac{\text{Effective embedding capacity}}{\text{Pixel number of each encrypted image}}, \tag{16}$$

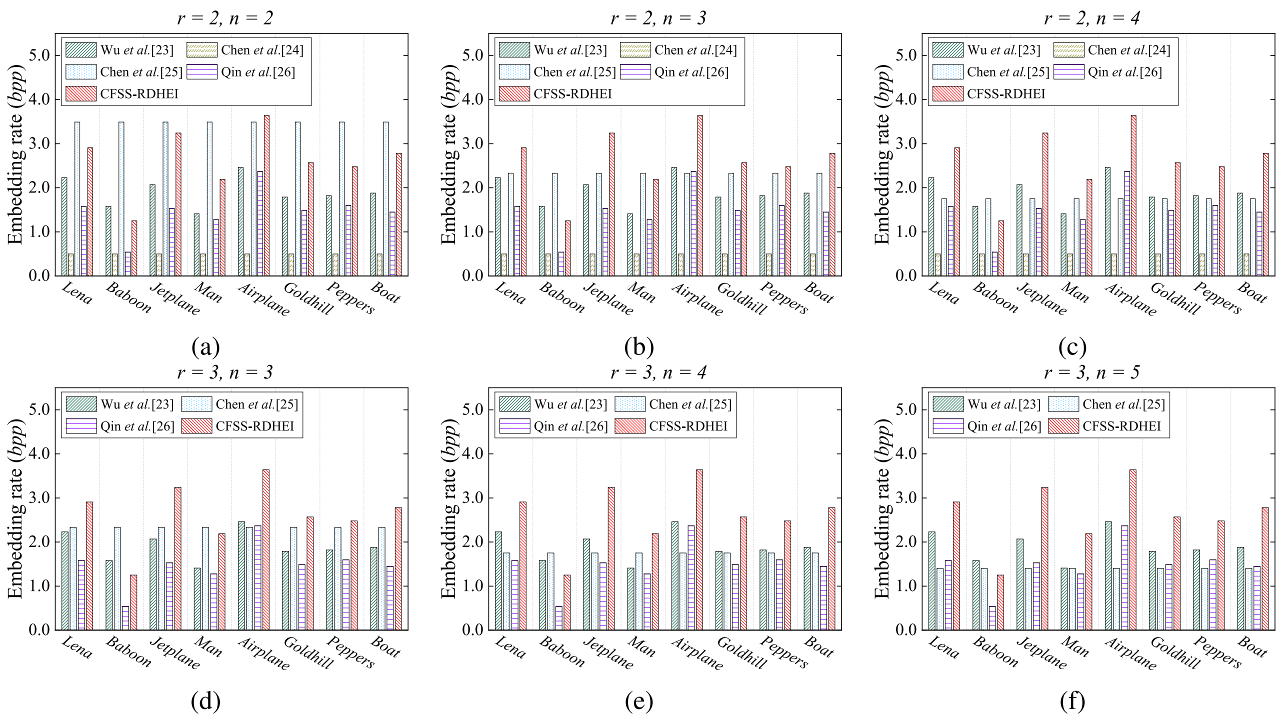


Fig. 7. Embedding rates of different secret sharing-based RDH-EI schemes at different (r, n)-thresholds.

where the effective embedding capacity indicates that the embeddable capacity subtracts the overhead $\boldsymbol{OH}$.

The embedding rate is expected to be large to embed more secret data. In our CFSS-RDHEI scheme, the embeddable capacity for an encrypted image of size $\tilde{M} \times \tilde{N}$ is $l\tilde{M}\tilde{N}$, where l is the optimal level of the original image, and the overhead $\boldsymbol{OH}$ consists of the share of final side information and some additional bits, discussed in Section III-B.2. In an RDH-EI scheme, the overhead usually indicates the extra information that is required to recover the embedded data and original image. However, to make the whole structure simpler and easier to understand, the final side information in our structure contains not only this extra information but also some image information (i.e. the $\boldsymbol{B}$ in Table II). Thus, the embedding rate of our CFSS-RDHEI scheme is $ER = (l\tilde{M}\tilde{N} - \boldsymbol{OH})/\tilde{M}\tilde{N}$.

Table IV lists the sizes of the embeddable capacity, overhead and effective embedding capacity and the embedding rates of our CFSS-RDHEI for eight test images with a $(2, n)$-threshold scheme. The optimal level l in different images may be different and result in different embeddable capacities and embedding rates. For example, the embedding rate of image *Lena* is 2.91 *bpp* with $l = 5$. The embedding rate of an image is mainly determined by the prediction errors of the image. The images *Baboon* and *Man* have relatively low embedding rates because they have small l and many unpredictable pixels, which lead to a small embeddable capacity and a large overhead. In contrast, the images *Airplane* and *Jetplane* can achieve much larger embedding rates.

For an (r, n)-threshold secret sharing-based RDH-EI, its embedding rate may be determined by the parameters r or n. Our experiment compares the embedding rates of the proposed CFSS-RDHEI with Wu *et al.* [23], Chen *et al.* [24], Chen *et al.* [25] and Qin *et al.* [26] methods, and Fig. 7 lists the experimental results. Chen *et al.* [24] method is not applicable when r is odd and the embedding rate of Chen *et al.* [25] method is $7/n$. The embedding rates of Wu *et al.* [23], Chen *et al.* [24] and Qin *et al.* [26] methods remain the same with different r or n values. For our CFSS-RDHEI, the embedding rates of an image are slightly different with different r values because the sizes of overhead may be different for different r values. The results show that Chen *et al.* [25] method can achieve the best embedding rate when $n = 2$. It is obvious that with the increase of r or n, our CFSS-RDHEI can achieve the best embedding rates for most test images.

TABLE IV
EMBEDDING RATES OF EIGHT TEST IMAGES WITH A $(2, n)$-THRESHOLD SCHEME

Test images	Optimal level l	Embeddable capacity	Overhead $\boldsymbol{OH}$	Effective embedding capacity	ER (*bpp*)
Lena	5	1,310,720	547,116	763,604	2.91
Baboon	4	1,048,576	720,307	328,269	1.25
Jetplane	6	1,572,864	723,111	849,753	3.24
Man	5	1,310,720	737,586	573,134	2.19
Airplane	6	1,572,864	619,034	953,830	3.64
Peppers	5	1,310,720	637,113	673,607	2.57
Goldhill	5	1,310,720	660,377	650,343	2.48
Boat	5	1,310,720	582,273	728,447	2.78

D. Time Complexity

Efficiency is one of the most important indicators for a secure system. We first compare the time complexity of our CFSS with several state-of-the-art secret sharing techniques

TABLE V
THEORETICAL TIME COMPLEXITY OF DIFFERENT SECRET SHARING TECHNIQUES

(r,n)	Methods	One sharing			Sharing for image of size $M \times N$		
		Addition	Multiplication	Total	Addition	Multiplication	Total
$(2,n)$	Harn $et\ al.$ [33]	3	4	7	$\frac{3MN}{2}$	$2MN$	$\frac{7MN}{2}$
	Liu $et\ al.$ [34] Meng $et\ al.$ [35]	3	4	7	$3MN$	$4MN$	$7MN$
	Wu $et\ al.$ [23] Chen $et\ al.$ [25] Qin $et\ al.$ [26]	1	1	2	MN	MN	$2MN$
	Chen $et\ al.$ [24]	3	6	9	$\frac{3MN}{2}$	$3MN$	$\frac{9MN}{2}$
	CFSS	1	1	2	MN	MN	$2MN$
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
(r,n)	Harn $et\ al.$ [33]	r^2-1	$r^2(r-1)$	$(r^2+r+1)(r-1)$	$\frac{(r^2-1)MN}{r}$	$r(r-1)MN$	$\frac{(r^2+r+1)(r-1)MN}{r}$
	Liu $et\ al.$ [34] Meng $et\ al.$ [35]	r^2-1	$r^2(r-1)$	$(r^2+r+1)(r-1)$	$(r^2-1)MN$	$r^2(r-1)MN$	$(r^2+r+1)(r-1)MN$
	Wu $et\ al.$ [23] Chen $et\ al.$ [25] Qin $et\ al.$ [26]	$r-1$	$\frac{r(r-1)}{2}$	$\frac{(r+2)(r-1)}{2}$	$(r-1)MN$	$\frac{r(r-1)MN}{2}$	$\frac{(r+2)(r-1)MN}{2}$
	Chen $et\ al.$ [24]	$2r-1$	$r(2r-1)$	$(r+1)(2r-1)$	$\frac{(2r-1)MN}{r}$	$\frac{r(2r-1)MN}{r}$	$\frac{(r+1)(2r-1)MN}{r}$
	CFSS	$r-1$	$\frac{r(r-1)}{2}$	$\frac{(r+2)(r-1)}{2}$	MN	$\frac{rMN}{2}$	$\frac{(r+2)MN}{2}$

in [33]–[35] and the secret sharing techniques used in different RDH-EI schemes [23]–[26]. Table V shows the comparison results. Our CFSS and the secret sharing techniques used in [23], [25], [26] have the same number of additions and multiplications in one sharing. The secret sharing technique used in [24] has the most time complexity for one sharing. However, it can process several pixels at a time, which makes its whole time complexity smaller than that of the secret sharing techniques in [34], [35]. Since our CFSS in the CFSS-RDHEI can share $r-1$ pixels in one sharing, the CFSS can achieve the least time complexity among all the secret sharing techniques when processing the same image.

In addition, we also calculate the real running time of different secret sharing-based RDH-EI schemes for different parties. All the programs were developed by Visual Studio 2019 with the C++ programming language and run on 64-bit Windows 10 with an Intel Core i5-9500 CPU @3.00 GHz (6 CPUs) and 8 GB RAM. For the sake of fairness, $(2,2)$, $(3,3)$ and $(4,4)$ thresholds are tested, and the embedding rate is set as 0.4 bpp. Table VI shows the time consumption of different RDH-EI schemes for different parties. It can be seen that for the $(2,2)$-threshold scheme, our CFSS-RDHEI scheme requires the least time in the data hider and requires the second least time in the content owner compared with other schemes. With the increase in the parameters r and n, our CFSS-RDHEI needs the least running time in all the content owners, data hiders and receivers. Note that the scheme in [24] only works when the parameter r is even.

V. SECURITY ANALYSIS

An RDH-EI scheme is expected to have high security to protect the embedded data and the original images. Many existing RDH-EI schemes can achieve high security only for embedded data. However, they cannot protect the original images well. In our CFSS-RDHEI, the CFSS follows the cryptography standards, and the embedded secret data can be encrypted by any encryption standard such as the well-known AES. Thus, it is able to protect the embedded secret data and original images. This section evaluates the security of our CFSS-RDHEI from the aspects of key sensitivity, Shannon entropy, differential attack and missing shares. These analyses can reflect the statistical characteristics of our CFSS-RDHEI.

TABLE VI
TIME CONSUMPTION (IN SECONDS) OF DIFFERENT SECRET SHARING-BASED RDH-EI SCHEMES FOR DIFFERENT PARTIES. THE TEST IMAGE IS *Lena* WITH A SIZE OF 512 × 512

(r,n)	Methods	Content Owner	Data Hider (0.4 *bpp*)	Receiver
$(2,2)$	Wu $et\ al.$ [23]	0.4843	0.0171	0.3385
	Chen $et\ al.$ [24]	0.3468	0.0053	0.7226
	Chen $et\ al.$ [25]	0.4847	0.0023	2.1751
	Qin $et\ al.$ [26]	0.4964	0.0066	0.3587
	CFSS-RDHEI	0.4028	0.0022	0.5572
$(3,3)$	Wu $et\ al.$ [23]	0.5290	0.0171	0.4819
	Chen $et\ al.$ [25]	0.5514	0.0023	2.7235
	Qin $et\ al.$ [26]	0.5483	0.0066	0.5176
	CFSS-RDHEI	0.2814	0.0015	0.4133
$(4,4)$	Wu $et\ al.$ [23]	0.6239	0.0171	0.5312
	Chen $et\ al.$ [24]	0.2166	0.0056	0.4981
	Chen $et\ al.$ [25]	0.6379	0.0023	3.5254
	Qin $et\ al.$ [26]	0.6452	0.0066	0.5847
	CFSS-RDHEI	0.2113	0.0011	0.3267

A. Key Sensitivity Analysis

In CFSS-RDHEI, an encryption key K_e is used to generate the parameters for the sharing process, namely, the two pseudorandom integer sequences $\mathbf{Q}$ and $\tilde{\mathbf{Q}}$ for image sharing and final side information sharing, respectively. Then, the security of the original image depends on both the number of shares and the encryption key. Because of the initial state sensitivity, unpredictability and easy implementation, a chaotic

Algorithm 1 Initial State Generation From the Encryption Key

```
Input: $F$, $K_e = [k_1, k_2, \cdots, k_{256}] (k_i \in \{0, 1\})$
  for $j = 1$ to 4 do
    $v_j \leftarrow \sum_{i=48j-47}^{48j} k_i \times 2^{-i+48j-48}$
  end for
  for $k = 1$ to 4 do
    $u_k \leftarrow \sum_{i=16k-15+192}^{16k+192} k_i \times 2^{i-16k+15-192}$
  end for
  $\hat{a} \leftarrow ((v_1 \times u_1) \mod 96) + 5$
  $\hat{b} \leftarrow ((v_2 \times u_2) \mod 96) + 5$
  $x_0 \leftarrow (v_3 \times u_3) \mod F$
  $y_0 \leftarrow (v_4 \times u_4) \mod F$
Output: Initial state $(\hat{a}, \hat{b}, x_0, y_0)$
```

system is suitable for generating pseudorandom numbers. Here, an *improved Hènon map* [36] with good performance is used and defined as

$$\begin{cases} x_{n+1} = (1 - \hat{a}x_n^2 + y_n) & \mod F; \\ y_{n+1} = \hat{b}x_n & \mod F, \end{cases} \tag{17}$$

where x_0 and y_0 are two initial values, and $\hat{a}$ and $\hat{b}$ are two control parameters. The encryption key K_e initializes $(\hat{a}, \hat{b}, x_0, y_0)$, and the procedure is shown in Algorithm 1. Then, the elements of $\mathbf{Q}$ and $\tilde{\mathbf{Q}}$ can be obtained as

$$\begin{cases} q_i = (\lfloor x_{i+1} \times 2^{21} \rfloor \mod F - n) + 1; \\ \tilde{q}_i = (\lfloor y_{i+1} \times 2^{21} \rfloor \mod 127 - n) + 1. \end{cases} \tag{18}$$

To demonstrate the key sensitivity of the CFSS-RDHEI, an encryption key K_e^0 is randomly generated. Then, change each bit of the K_e^0 to obtain 256 different keys, in which each one has only one-bit difference with K_e^0. The number of bit change rates (NBCRs) [37] is used to calculate the difference between two images. For two encrypted images E_1 and E_2 encrypted by two keys with one-bit difference, their NBCR is defined as

$$NBCR = \frac{Ham(E_1, E_2)}{Len}, \tag{19}$$

where $Ham(E_1, E_2)$ means the Hamming distance of E_1 and E_2, and Len is the bit length of an image. For two random and independent images C_1 and C_2, the probabilities of bit pair $(C_1(i), C_2(j))$ being 00, 01, 10, 11 are 1/4, 1/4, 1/4, and 1/4, respectively. Thus, $Ham(C_1(i), C_2(i)) = 1/4 + 1/4 = 50\%$. Since this is workable for each bit, the NBCR of two random and independent images is 50%.

Fig. 8 shows the experimental results of the key sensitivity analysis in a $(2, 2)$-threshold sharing scheme. Each encryption generates two encrypted images, and we calculate the NBCRs of two encrypted images from two encryption processes with one-bit different keys. This means that E_1 and E_2 are two encrypted images encrypted using one-bit different keys. The results indicate that a one-bit change in the encryption key causes different encrypted images. Therefore, the CFSS-RDHEI is sensitive to its encryption key. To provide a fair comparison, all the comparison methods use this key generator function in our experiments.

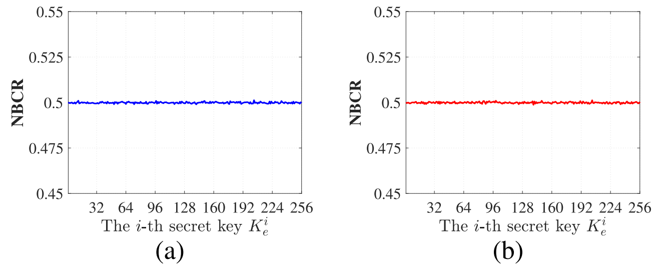


Fig. 8. Key sensitivity analysis of the CFSS-RHDEI in each bit of the encryption key. (a) NBCR of the first encrypted image; (b) NBCR of the second encrypted image.

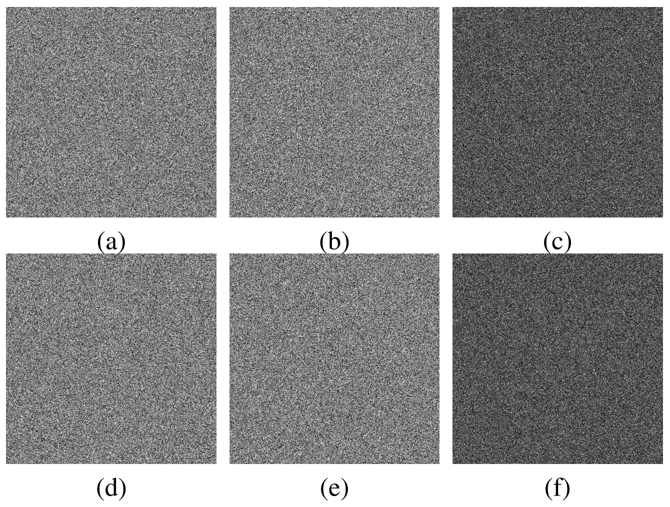


Fig. 9. Encrypted results of image *Lena* by $(2, 2)$-threshold CFSS-RDHEI using the same encryption key in two executions: (a) the first encrypted image in the first execution; (b) the first encrypted image in the second execution; (c) difference of (a) and (b); (d) the second encrypted image in the first execution; (e) the second encrypted image in the second execution; (f) difference of (d) and (e).

B. Nondeterminacy

The CFSS scheme used in the CFSS-RDHEI is a randomized and nondeterministic encryption system. When using the CFSS scheme to encrypt an image twice with the same encryption key, the generated encrypted images are completely different. This is because some random values in Eq. (3) are used, and these random values are different in each encryption. Fig. 9 shows the encrypted results of image *Lena* by a $(2, 2)$-threshold scheme using the same encryption key in two executions. The encrypted images generated in the two executions are completely different (see Figs. 9 (c) and (f)). A nondeterministic and randomized encryption system has the ability to resist many potential attacks [38].

C. Shannon Entropy

An encrypted image is expected to have uniformly distributed pixels to defend against statistics-based security attacks. The Shannon entropy can be used to measure the distribution of image pixels and is mathematically defined as

$$H = -\sum_{i=1}^{N_a} Pr(x_i) \log Pr(x_i), \tag{20}$$

TABLE VII
SHANNON ENTROPIES OF ENCRYPTED IMAGES ENCRYPTED BY DIFFERENT SECRET SHARING-BASED METHODS FOR THE IMAGE *Lena*

Methods	(3,3)-threshold scheme			(5,6)-threshold scheme					
	Share 1	Share 2	Share 3	Share 1	Share 2	Share 3	Share 4	Share 5	Share 6
Chen $et\ al.$ [24]		7.9708				7.9708			
Wu $et\ al.$ [23]	7.9707	7.9707	7.9709	7.9708	7.9708	7.9708	7.9708	7.9707	7.9707
Chen $et\ al.$ [25]	7.9708	7.9709	7.9709	7.9708	7.9708	7.9708	7.9709	7.9708	7.9708
Qin $et\ al.$ [26]	7.9707	7.9706	7.9707	7.9708	7.9707	7.9707	7.9707	7.9706	7.9707
CFSS-RDHEI ($l = 4$)	7.9972	7.9967	7.9967	7.9955	7.9956	7.9950	7.9956	7.9951	7.9958
CFSS-RDHEI ($l = 5$)	7.9929	7.9930	7.9961	7.9916	7.9921	7.9912	7.9908	7.9918	7.9917
CFSS-RDHEI ($l = 6$)	7.9982	7.9981	7.9982	7.9965	7.9968	7.9965	7.9966	7.9965	7.9966

TABLE VIII
NPCR, UACI, SRCC, KRCC RESULTS OF CFSS-RDHEI AND RELATED METHODS IN A (5,6)-THRESHOLD SCHEME

	NPCR (%)						UACI (%)					
Chen $et\ al.$ [24]			0.0022						0.0004			
Wu $et\ al.$ [23]	99.6017	99.6002	99.6208	99.6155	99.6056	99.6170	32.8110	32.7243	32.8842	32.7243	32.7288	32.8197
Chen $et\ al.$ [25]	99.6033	99.6037	99.5995	99.6067	99.6056	99.5956	32.7525	32.8473	32.8096	32.8448	32.7970	32.9304
Qin $et\ al.$ [26]	99.5850	99.6323	99.6429	99.6017	99.5819	99.6033	32.7984	32.7492	32.8662	32.8237	32.9214	32.6916
CFSS-RDHEI ($l = 4$)	99.6274	99.6123	99.6785	99.6239	99.6343	99.5985	33.4264	33.5621	33.6142	33.6058	33.6721	33.5948
CFSS-RDHEI ($l = 5$)	99.5873	99.6306	99.5839	99.5663	99.6274	99.5957	33.4778	33.3290	33.3626	33.5272	33.4318	33.3787
CFSS-RDHEI ($l = 6$)	99.6341	99.5869	99.6637	99.6594	99.6334	99.5749	33.4483	33.4045	33.3606	33.3328	33.3973	33.4784
	SRCC (%)						KRCC (%)					
Chen $et\ al.$ [24]			99.9991						99.9999			
Wu $et\ al.$ [23]	0.3067	0.3292	0.2526	0.6265	-0.1574	0.3723	0.2056	0.2503	0.1148	0.4723	-0.0771	0.1858
Chen $et\ al.$ [25]	0.3334	-0.1615	0.1474	-0.2523	0.2585	0.1691	0.2196	-0.0463	0.0372	-0.1574	0.1230	0.0053
Qin $et\ al.$ [26]	-0.0024	0.0007	0.0001	0.0013	-0.0042	0.0011	-0.0016	0.0005	0.0001	0.0001	-0.0028	0.0007
CFSS-RDHEI ($l = 4$)	-0.0892	0.4466	0.7054	0.7499	0.6294	0.0326	-0.0583	0.2991	0.4726	0.5019	0.4215	0.0224
CFSS-RDHEI ($l = 5$)	0.3127	0.5512	0.3707	-0.2935	0.3309	0.2614	0.0205	0.3618	0.2462	-0.1964	0.2276	0.1798
CFSS-RDHEI ($l = 6$)	-0.1301	0.1624	0.3375	0.6348	0.5170	0.0039	-0.0884	0.1093	0.2258	0.4251	0.3460	0.0025

where N_a is the number of possible pixel values and $Pr(x_i)$ is the probability of the i-th possible value. For an 8-bit grayscale image, $N_a = 256$, and the theoretical maximum Shannon entropy is obtained when each possible value has the same probability, namely, $Pr(x_i) = 1/256$ for $i \in [1, N_a]$. Thus, the theoretically maximum Shannon entropy $H_{\max} = -\sum_{i=1}^{256} 1/256 \times \log(1/256) = 8$. A larger Shannon entropy indicates a more uniform distribution of the image pixels.

Table VII lists the Shannon entropies of encrypted images encrypted by different secret sharing-based RDH-EI methods. Chen *et al.* [24] method can generate only one encrypted image for different threshold schemes. For the proposed CFSS-RDHEI, we calculate the Shannon entropies of encrypted images when the optimal level $l \in \{4, 5, 6\}$. The CFSS-RDHEI can generate encrypted images that have larger Shannon entropies than the other three methods and are very close to the theoretical maximum value of 8. This indicates that its generated encrypted images have high pixel randomness.

D. Differential Attack

The differential attack is an effective cryptanalysis method. It studies how the difference in the plaintext can affect the difference in the ciphertext. By choosing the plaintext to encrypt, the attackers can build the connections between the plaintext and ciphertext and then use these built connections to reconstruct the ciphertext without a secret key.

To resist the differential attack, the slight change in the plaintext should cause a large difference in the ciphertext. The number of pixel change rates (NPCRs) and uniform average change intensity (UACI) [39] are two measurements to test the ability of a cryptosystem to resist differential attacks. The NPCR is the total number of different pixels in two images, and the UACI means the average difference between pixels in two images. According to the theoretical analysis in [39], the ideal NPCR and UACI values for 8-bit grayscale images are 99.6094% and 33.4635%, respectively.

To investigate the correlation of the encrypted images, we use Spearman's rank correlation coefficient (SRCC) and Kendall rank correlation coefficient (KRCC) [40] to evaluate their monotonicity. The obtained SRCC and KRCC are within the range $[-1, 1]$. A value close to -1 or 1 means a high correlation between two images, and a value close to 0 indicates a low correlation.

Table VIII lists the results of the $(5, 6)$-threshold scheme. The NPCR and UACI values of Chen *et al.* [24] method are close to 0, and its SRCC and KRCC values are close to 1. This is because its secret sharing technique does not have random integers and diffusion properties, and a one-bit change in the original images causes a change in only the current section in the encrypted images. Then, this secret sharing-based encryption strategy is weak to resist the differential attack. On the other hand, the proposed CFSS-RDHEI and other schemes can obtain NPCRs and UACIs that are close to the theoretical values, namely, 99.6094% and 33.4635%, and the SRCC and KRCC values are all close to 0. This is because the schemes in [23], [25], [26] and our CFSS contain random numbers in the sharing process. However, when using many random numbers to achieve a high security level, each encrypted image in these schemes has the same size as the original image, and all of these schemes cause serious data expansion, which can be seen in Table III. Our CFSS-RDHEI scheme uses a cipher-feedback strategy to replace these random numbers, and each encrypted image has only $1/(r-1)$ size of the original image while being well protected in this attack.

E. Missing Share

Our CFSS-RHDEI uses the CFSS to encrypt the original image to be n encrypted images. Only by using r $(r \le n)$ encrypted images can one completely recover the original image. This property is guaranteed by Eq. (3), which indicates that at least r equations are required to solve the r coefficients $a_0, \ldots, a_{r-1}$ of the polynomial. Assuming that an attacker has collected $r-1$ encrypted images, $r-1$ equations can be constructed. The probability of solving r variables using $r-1$ equations is $1/F$. For an image of size $M \times N$, there are $(M \times N)/(r-1)$ sections. Then, the probability of correctly recovering the original image using $r-1$ encrypted images is $(1/F)^{M\times N/(r-1)}$. Thus, the attackers have difficulty correctly recovering the original image from $r-1$ encrypted images. All secret sharing techniques have the ability to resist this collusion analysis, since this ability is a basic property for a secret sharing technique.

VI. Conclusion

In this paper, we first developed a new polynomial-based secret sharing scheme called CFSS. It adopts the cipher-feedback strategy to process image pixels. Using the CFSS, we further proposed an RDH-EI scheme with multiple data hiders called CFSS-RDHEI. A multi-MSB prediction method is used to embed secret data. First, an optimal level is set according to its prediction precision. Then, the side information is generated to store the required information of recovering the multi MSBs of the image. Next, the final side information and multi-LSBs of the image are encrypted to several shares by CFSS, and each share of the final side information is embedded into one encrypted image. Finally, the content owner sends each obtained encrypted image to one data hider, which can embed secret data into the encrypted image. A receiver can extract the embedded data or recover the original image from one or a predefined number of marked encrypted images. Performance evaluations show that our CFSS-RDHEI has a high capacity rate and that its generated encrypted images are much smaller, while still being well protected, compared to other secret sharing-based RDH-EI schemes. Since the proposed CFSS processes image pixels in a prime number field, preprocessing is required to process these pixels. Our future work will explore the image secret sharing technique without preprocessing and use this technique in reversible data hiding in encrypted images.

Acknowledgment

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions that greatly contributed to improving the quality of the manuscript.

References

[1] X. Zhang, Y. Ren, L. Shen, Z. Qian, and G. Feng, "Compressing encrypted images with auxiliary information," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1327–1336, Aug. 2014.

[2] Y. Ke, M.-Q. Zhang, J. Liu, T.-T. Su, and X.-Y. Yang, "Fully homomorphic encryption encapsulated difference expansion for reversible data hiding in encrypted domain," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 8, pp. 2353–2365, Aug. 2020.

[3] 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.*, early access, Mar. 1, 2021, doi: 10.1109/TCSVT.2021.3062947.

[4] Y. Du, Z. Yin, and X. Zhang, "High capacity lossless data hiding in JPEG bitstream based on general VLC mapping," *IEEE Trans. Depend. Sec. Comput.*, early access, Jan. 31, 2021, doi: 10.1109/TDSC.2020.3013326.

[5] W. Puech, M. Chaumont, and O. Strauss, "A reversible data hiding method for encrypted images," *Proc. SPIE*, vol. 6819, Feb. 2008, Art. no. 68191E.

[6] X. Zhang, "Reversible data hiding in encrypted image," *IEEE Signal Process. Lett.*, vol. 18, no. 4, pp. 255–258, Apr. 2011.

[7] X. Liao and C. Shu, "Reversible data hiding in encrypted images based on absolute mean difference of multiple neighboring pixels," *J. Vis. Commun. Image Represent.*, vol. 28, pp. 21–27, Apr. 2015.

[8] J. Zhou, W. Sun, L. Dong, X. Liu, O. C. Au, and Y. Y. Tang, "Secure reversible image data hiding over encrypted domain via key modulation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 3, pp. 441–452, Mar. 2016.

[9] X. Zhang, "Separable reversible data hiding in encrypted image," *IEEE Trans. Inf. Forensics Security*, vol. 7, no. 2, pp. 826–832, Apr. 2012.

[10] K. Ma, W. Zhang, X. Zhao, N. Yu, and F. Li, "Reversible data hiding in encrypted images by reserving room before encryption," *IEEE Trans. Inf. Forensics Security*, vol. 8, no. 3, pp. 553–562, Mar. 2013.

[11] W. Zhang, K. Ma, and N. Yu, "Reversibility improved data hiding in encrypted images," *Signal Process.*, vol. 94, pp. 118–127, Jan. 2014.

[12] X. Zhang, J. Long, Z. Wang, and H. Cheng, "Lossless and reversible data hiding in encrypted images with public-key cryptography," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 26, no. 9, pp. 1622–1631, Sep. 2016.

[13] S. Yi and Y. Zhou, "Binary-block embedding for reversible data hiding in encrypted images," *Signal Process.*, vol. 133, pp. 40–51, Apr. 2017.

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

[15] 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, Apr. 2020.

[16] 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, Aug. 2020.

[17] C. Qin, X. Qian, W. Hong, and X. Zhang, "An efficient coding scheme for reversible data hiding in encrypted image with redundancy transfer," *Inf. Sci.*, vol. 487, pp. 176–192, Jun. 2019.

[18] 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, Aug. 2019.

[19] L. Y. Zhang *et al.*, "On the security of a class of diffusion mechanisms for image encryption," *IEEE Trans. Cybern.*, vol. 48, no. 4, pp. 1163–1175, Apr. 2018.

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

[21] H.-T. Wu, Y.-M. Cheung, Z. Yang, and S. Tang, "A high-capacity reversible data hiding method for homomorphic encrypted images," *J. Vis. Commun. Image Represent.*, vol. 62, pp. 87–96, Jul. 2019.

[22] C. Jiang and Y. Pang, "Encrypted images-based reversible data hiding in Paillier cryptosystem," *Multimedia Tools Appl.*, vol. 79, nos. 1–2, pp. 693–711, Jan. 2020.

[23] X. Wu, J. Weng, and W. Yan, "Adopting secret sharing for reversible data hiding in encrypted images," *Signal Process.*, vol. 143, pp. 269–281, Feb. 2018.

[24] 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 Security*, vol. 14, no. 12, pp. 3332–3343, Dec. 2019.

[25] 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. Depend. Sec. Comput.*, early access, 2020, doi: 10.1109/TDSC.2020.3011923.

[26] 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.*, early access, Jun. 21, 2021, doi: 10.1109/TCSVT.2021.3091319.

[27] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.

[28] C.-C. Thien and J.-C. Lin, "Secret image sharing," *Comput. Graph.*, vol. 26, no. 5, pp. 765–770, Oct. 2002.

[29] M. J. Weinberger, G. Seroussi, and G. Sapiro, "The LOCO-I lossless image compression algorithm: Principles and standardization into JPEG-LS," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1309–1324, Aug. 2000.

[30] B. Patrick and F. Teddy. (2007). *Image Database of BOWS-2*. [Online]. Available: http://bows2.ec-lille.fr/

[31] X. Li, B. Yang, and T. Zeng, "Efficient reversible watermarking based on adaptive prediction-error expansion and pixel selection," *IEEE Trans. Image Process.*, vol. 20, no. 12, pp. 3524–3533, Dec. 2011.

[32] 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, Jan. 2019.

[33] L. Harn and C.-F. Hsu, "(t, n) multi-secret sharing scheme based on bivariate polynomial," *Wireless Pers. Commun.*, vol. 95, no. 2, pp. 1495–1504, Jul. 2017.

[34] Y. Liu, C. Yang, Y. Wang, L. Zhu, and W. Ji, "Cheating identifiable secret sharing scheme using symmetric bivariate polynomial," *Inf. Sci.*, vol. 453, pp. 21–29, Jul. 2018.

[35] K. Meng, F. Miao, W. Huang, and Y. Xiong, "Threshold changeable secret sharing with secure secret reconstruction," *Inf. Process. Lett.*, vol. 157, May 2020, Art. no. 105928.

[36] Z. Hua, Y. Zhang, and Y. Zhou, "Two-dimensional modular chaotification system for improving chaos complexity," *IEEE Trans. Signal Process.*, vol. 68, pp. 1937–1949, 2020.

[37] J. C. H. Castro, J. M. Sierra, A. Seznec, A. Izquierdo, and A. Ribagorda, "The strict avalanche criterion randomness test," *Math. Comput. Simul.*, vol. 68, no. 1, pp. 1–7, Feb. 2005.

[38] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Boca Raton, FL, USA: CRC Press, 2020.

[39] Y. Wu, J. P. Noonan, and S. Agaian, "NPCR and UACI randomness tests for image encryption," *Cyber J., Multidisciplinary J. Sci. Technol., J. Sel. Areas Telecommun.*, vol. 1, no. 2, pp. 31–38, 2011.

[40] Z. Wang and Q. Li, "Information content weighting for perceptual image quality assessment," *IEEE Trans. Image Process.*, vol. 20, no. 5, pp. 1185–1198, May 2011.

Zhongyun Hua (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, Shenzhen, China. He has published more than 50 papers on the subject and receiving more than 2700 citations. His research interests include chaotic systems, image processing, and information security. He serves as an Associate Editor for International Journal of Bifurcation and Chaos.

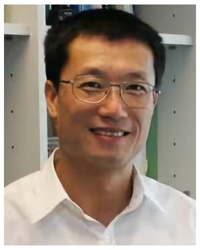
Yanxiang Wang received the B.S. degree in software engineering from Tongji University, Shanghai, China, in 2015. He is currently pursuing the M.S. degree in computer science and technology with the Harbin Institute of Technology, Shenzhen, China. His research interests include reverible data hiding and secret sharing.

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.

Yicong Zhou (Senior Member, IEEE) received the B.S. degree in electrical engineering from Hunan University, Changsha, China, and the M.S. and Ph.D. degrees in electrical engineering from Tufts University, Medford, MA, USA.

He is currently a Professor with the Department of Computer and Information Science, University of Macau, Macau, China. His research interests include image processing, computer vision, machine learning, and multimedia security. He is a fellow of the Society of Photo-Optical Instrumentation Engineers (SPIE) and was recognized as one of "World's Top 2% Scientists" and one of "Highly Cited Researchers" in 2020 and 2021. He received the Third Price of Macao Natural Science Award as a Sole Winner in 2020 and a co-recipient in 2014. He has been a leading the Co-Chair of Technical Committee on Cognitive Computing in the IEEE Systems, Man, and Cybernetics Society since 2015. He serves as an Associate Editor for IEEE TRANSACTIONS ON NEUTRAL NETWORKS AND LEARNING SYSTEMS, IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS FOR VIDEO TECHNOLOGY, IEEE TRANSACTIONS ON GEOSCIENCE AND REMOTE SENSING, and four other journals.



Xiaohua Jia (Fellow, IEEE) received the B.Sc. and M.Eng. degrees from the University of Science and Technology of China in 1984 and 1987, respectively, and the D.Sc. degree in information science from the University of Tokyo in 1991.

He is currently an Adjunct with the Harbin Institute of Technology, Shenzhen, while performing this work. He is also the Chair Professor with the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, wireless sensor networks, and mobile wireless networks. He is a fellow of the IEEE Computer Society. He is the General Chair of ACM MobiHoc 2008, the TPC Co-Chair of IEEE MASS 2009, the Area-Chair of IEEE INFOCOM 2010, the TPC Co-Chair of IEEE GlobeCom 2010-Ad Hoc and Sensor Networking Symposium, and the Panel Co-Chair of IEEE INFOCOM 2011. He is an editor of the IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS from 2006 to 2009, *Wireless Networks*, *Journal of World Wide Web*, and *Journal of Combinatorial Optimization*.