# A High Capacity Test Disguise Method Combined With Interpolation Backup and Double Authentications

Hai Lu, Shaanxi Normal University, China
Liping Shao, Shaanxi Normal University, China*
Qinglong Wang, Shaanxi Normal University, China

## ABSTRACT

To improve the hidden capacity of a single question, further avoid the absence of authentication, and provide self-repair ability, this paper proposes a high capacity test disguise method combined with interpolation backup and double authentications. Firstly, secret byte sequence is backed up and further encoded to a backup index sequence by secret information backup and encoding strategy. Secondly, a test question database divided into eight sets is created. Finally, the backup index sequence is disguised as a stego test paper using 24 different candidate answer orders and 4-bit hash values. In restoration, double authentications are applied to authenticate candidate restored value, and the most reliable candidate restored values are obtained by the reliable calculation to reconstruct secret information. The experimental results and analysis show that the proposed method can distinguish error candidate restored values and calculate the reliability of each restored byte. Moreover, it has excellent self-repair ability with a higher hidden capacity of a single question.

## KEYWORDS

Double Authentication, High Capacity, Interpolation Backup, Lagrange Interpolation Over GF(2^8), Reliable Calculation, Self-Repair Ability, Stem Hash Value, Test Disguise

## INTRODUCTION

Information hiding is a technique to hide secret information in other irrelevant carriers to make embedded information invisible. Traditional information hiding usually modifies carrier redundancy for embedding. However, it is challenging to escape detections from steganalysis for it always leaves modification traces. To address it, CIH (coverless information hiding) is proposed. Unlike modification-based information hiding, CIH generates or searches stego carriers according to secret information directly. In CIH, SCIH (search-based CIH) is a typical class of strategies. The main idea of SCIH is finding appropriate carriers from a big database and using keywords following location tags to express secret information.

According to different properties of text and image, SCIH has two categories: one is SCIHT (SCIH for text), the other is SCIHI(SCIH for image).

SCIHT usually regarded text features as location tags to mark positions of secrets such as radicals of Chinese characters, hash values about series of Chinese character, ranks of English words, and so

DOI: 10.4018/ijdcf.295815

*Corresponding Author

on. For example, Chen et al. (2015) regarded Chinese character radicals appeared in the top 50 as location tags and used Chinese characters or words following location tags as keywords. To avoid the extraction confusion in the work of Chen et al. (2015), Zhou et al. (2016b) restricted the range of location tags to the first appearing top 50 radicals in texts, and the hidden information following location tags must be a single character. Chen et al. (2017) further transformed a series of $n$ Chinese characters into $n$ -bit hash value as location tags to extract secret Chinese words following them. To enhance the hidden capacity of the work of Chen et al. (2017), Chen et al. (2018) regarded the high-frequency combination of Chinese word following location tags as keywords. To reduce the number of texts in database and improve the success rate of secret matching, Xia et al. (2017) used 12 predefined positions as location tags in one text, regarded LSBs of located Chinese characters as keywords. Based on English linguistic characteristics, Zhang et al. (2017a) and Zhang et al. (2017b) employed both the word rank map and the frequent word distance to express secret information.

SCIHI usually, according to certain mapping rules, mapped entire images or image blocks to bitstreams or Chinese words, which served as keywords. In other words, SCIHI divided an image database into various sets and used the set index to express secret information. For example, Zhou et al. (2015) and Yuan et al. (2017) mapped images into 8-bit hash values as keywords without location tags. Among them, Zhou et al. (2015) divided every image into nine blocks to form 8-bit hash by comparing block mean values, while Yuan et al. (2017) generated hash values by visual words based on SIFT and K-means. However, the security of these works is low because of the absence of key-related location tags. To improve security, some methods regarded image block positions as location tags which are generated by user keys (Cao et al., 2018; Zhang et al., 2018; Zhou et al., 2016a, 2017). Among them, Zhou et al. (2017) transformed the gradient magnitude and direction of each image block into 20-bit keywords. Zhou et al. (2016a) created the mapping rule between visual words and Chinese words and further used the maximum frequency of visual words to determine decryption order. Cao et al. (2018) restricted the type of images to molecular structure images of material for flexible expressing, and further used set index of block mean values as keywords. Zhang et al. (2018) mapped DC coefficients of each block to $M$ -bit hash value as keywords and employed average values of block DC coefficients in a single image to determine the decryption order. To reduce the size of the image database, Zheng et al. (2017) divided each image into nine blocks as nine leaf nodes, and every block was mapped into 2-bit hash by extreme points using SIFT. Therefore, one image can represent at most 256 different keywords by changing the order of nine leaf nodes. To improve the hidden capacity and decrease the search cost, Wu et al. (2018) mapped every image into a 7-bit hash and combined four images to express a 32-bit keyword. Among them, 32-bit keyword consisting of four image hash values(28 bits) and 16 different combination order(four bits). Zou et al. (2019) mapped images into 80-bit hash values by block mean comparing, and used every 20-bit hash in an image to express Chinese sentence elements. Based on the work of Zhou et al. (2015), Zhou et al. (2019) transformed image blocks to hash values as location tags, employed located blocks to reconstruct secret image approximately.

However, there are some problems in the above works.

1. A small range of selected location tags cannot provide enough security. For example, Chen et al. (2015) and Zhou et al. (2016b) only regarded the top 50 frequent radicals as location tags. Zhou et al. (2017) only used 16 block positions as location tags. Some methods hid secret information in images without encryption or key-related location tags (Zhou et al., 2015; Yuan et al., 2017)
2. Unmodified images and texts have a poor ability to express unrelated secret information. For example, the hidden capacities are one Chinese character (Chen et al., 2015), 2.41 Chinese characters (Chen et al., 2018), 8 bits (Zhou et al., 2015), 32 bits (Wu et al., 2018), and so on. Moreover, a large database is necessary to improve the expressive ability of SCIH. For example, the numbers of texts in databases are approximately 11 million (Zhou et al., 2016b) and 5 million (Chen et al. 2017), respectively. The numbers of images in databases are 20 million (Zhou et al.,

2017) and 50968 (Zou et al., 2019), respectively. Although the number of images in the work of Zheng et al. (2017) is only 2000, it goes against the primary intention of SCIH because it embeds extra information by changing LSBs of images.

3. Low expressive ability results in a huge database, which poses a heavy burden when searching for appropriate carriers from the database exhaustively (Xia et al., 2017; Zhang, 2017a; Zhang, 2017b). To address this problem, SCIH usually built a database with MIIS (multi-level inverted index structure). The cost of database creation, search, and maintenance is high even with MIIS. For example, the sizes of index tables are 8.4 - 9.5 MB (Chen et al., 2017) and 60.7 - 125.4 MB (Chen et al., 2018), respectively, when mapping parameter $n = 4, 5, \cdots, 10$. And index tables must be rebuilt when $n$ is changed. In the work of Zhou et al. (2017), the image database was divided into $2^{20} \times 16 = 16777216$ sets to express secret information. Zhou et al. (2016a) and Zhang et al. (2018) further employed the maximum frequency of visual words and average values of block DC coefficients to determine the decryption order based on the work of Zhou et al. (2017). Therefore the costs of these works are higher.
4. Due to low hidden capacity, the dense transmission of massive carriers is necessary. Although every single carrier is unmodified, which can resist steganalysis, the dense transmission will also attract attackers, and the quality of restored information can't be guaranteed under attacks. However, the authentication strategy is not considered in SCIH.

In Lu et al. (2018), they regarded arithmetic multiple-choice and blank-filling randomly generated by user keys as hiding unit to express a 5-bit secret segment and combined the hidden question into a stego test paper for a confidential transmission. Although this method can avoid low hidden capacity, high cost of database creation, search, maintenance and dense transmission of massive carriers, and only a part of secret information is necessary for transmission because of the random code, and some problems need to be solved:

1. Without self-repair ability, the quality of restored information under attacks cannot be guaranteed.
2. Without authentications, it cannot distinguish error extracted information, and cannot calculate the reliability of the restored information.
3. The hidden capacity is only 2.5 bits/single question because of low utilization of question redundancy.

To avoid problems in Lu et al. (2018), this paper proposes a high capacity test paper disguise method combined with backup interpolation and double authentications. The main contributions are followings:

1. Use Lagrange $(2, 4)$ polynomial interpolation over $\mathrm{GF}(2^8)$ to back up a secret byte. It can obtain six candidate restored values for each secret byte in restoration. And the quality of the restored information is good even if some candidate restored values are damaged.
2. Introduce codebook extension authentication and sharing coefficient authentication as double authentications. Among them, the codebook extension authentication is used to authenticate bytes extracted from the stego test paper, while the sharing coefficient authentication is employed to mark error candidate restored values to avoid unreliable computing. Moreover, frequencies of candidate restored values can determine the reliability of restored information.
3. Create a question stem database, and further map all stems into 4-bit hash values. In disguise process, employ 24 different candidate answer orders and 4-bit hash values of a single multiple-choice to express a number in [0,383]. So the hidden capacity of a single question in the proposed method is higher than Lu et al. (2018).

The rest of this paper is organized as follows. Sec. 2 describes key strategies. Sec. 3 depicts the complete steps of the proposed method. Sec. 4 shows the experimental data. Finally, Sec 5 gives the conclusion.

## THE PROPOSED METHOD

Figure 1 shows the flowchart of the proposed method, where the input and output sets are presented as the elliptic boxes, and the proposed strategies are represented as the solid boxes. As shown in Figure 1, the secret byte sequence $\boldsymbol{P}_S = (p_i)_l$ is transformed into the backup index sequence $\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$ by the secret information backup and encoding process. Furthermore, $\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$ is disguised as the hidden question sequence $\boldsymbol{L}_1 = (st_i)_{l_1}$ by the test disguise strategy combined with candidate answer order and stem hash value. In addition, some non-hidden questions are also selected from the test database to form the non-hidden question sequence $\boldsymbol{L}_2 = (st_i)_{l_2}$ . Finally, $\boldsymbol{L}_1$ and $\boldsymbol{L}_2$ are combined to form the stego test paper $\boldsymbol{L} = (st_i)_{l_3}$ for the secret transmission. Suppose $\boldsymbol{L}'$ is the received version of $\boldsymbol{L}$ . The information reconstruction strategy combined with double authentications and reliable calculation is employed to restore the reconstructed byte sequence $\boldsymbol{P}_S' = (p_i)_l$ . Moreover, the reliability sequence $\boldsymbol{A} = (a_i)_l$ is generated to mark the reliability of reconstructed bytes.

**Figure 1. The Flowchart of the proposed method**

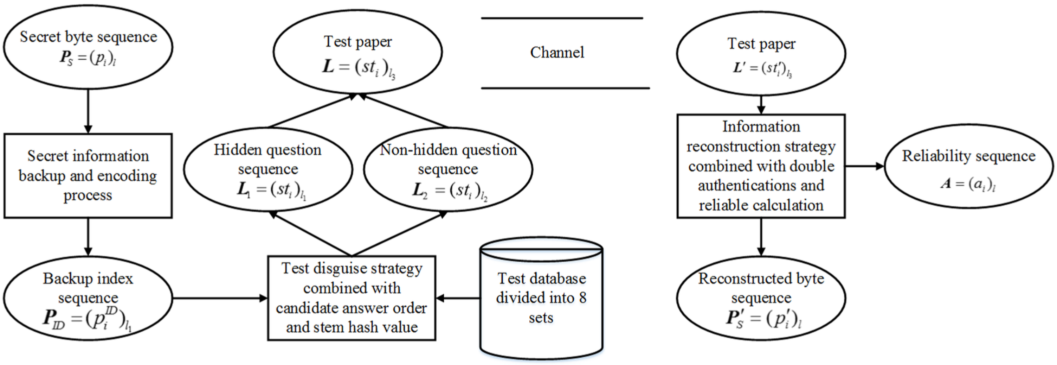

Secret byte sequence
$\boldsymbol{P}_S = (p_i)_l$
Test paper
$\boldsymbol{L} = (st_i)_{l_3}$
Channel
Test paper
$\boldsymbol{L}' = (st_i')_{l_3}$
Secret information backup and encoding process
Hidden question sequence
$\boldsymbol{L}_1 = (st_i)_{l_1}$
Non-hidden question sequence
$\boldsymbol{L}_2 = (st_i)_{l_2}$
Information reconstruction strategy combined with double authentications and reliable calculation
Reliability sequence
$\boldsymbol{A} = (a_i)_l$
Backup index sequence
$\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$
Test disguise strategy combined with candidate answer order and stem hash value
Test database divided into 8 sets
Reconstructed byte sequence
$\boldsymbol{P}_S' = (p_i')_l$

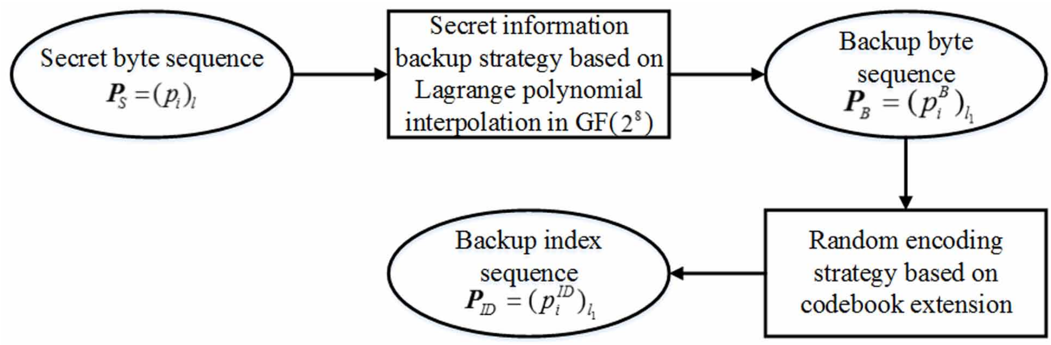
Figure 1 presents three important strategies: (1) the secret information backup and encoding process; (2) the test disguise strategy combined with candidate answer order and stem hash value and (3) the information reconstruction strategy combined with double authentications and reliable calculation. Among them, (1) is used to back up secret information $\boldsymbol{P}_S$ and further encode it to the backup index sequence $\boldsymbol{P}_{ID}$ . Then, (2) is employed to disguise $\boldsymbol{P}_{ID}$ as a stego test paper $\boldsymbol{L} = (st_i)_{l_3}$ for secret transmission. In restoration, (3) is applied to restore the reconstructed byte sequence $\boldsymbol{P}_S' = (p_i)_l$ and then generate the reliability sequence $\boldsymbol{A} = (a_i)_l$ to mark the reliability of $\boldsymbol{P}_S' = (p_i)_l$ .

### Secret Information Backup and Encoding Strategy

SCIH and Lu et al. (2018) do not consider self-repair ability. Therefore, the quality of restored information is low under attacks. To address this, Lagrange $(2,4)$ polynomial interpolation over $\mathrm{GF}(2^8)$ is constructed to share every secret byte $\forall p_i \in \boldsymbol{P}_S$ , and four backup bytes $p_{4i}^B, p_{4i+1}^B, p_{4i+2}^B, p_{4i+3}^B \in \boldsymbol{P}_B$ are obtained. Moreover, $\boldsymbol{P}_B$ is further encoded to the backup index sequence $\boldsymbol{P}_{ID}$ by the random code strategy. In restoration, six candidate restored values can be calculated by

$p_{4i}^{B}, p_{4i+1}^{B}, p_{4i+2}^{B}, p_{4i+3}^{B} \in \boldsymbol{P}_B$ through the Lagrange (2,4) polynomial interpolation over $\mathrm{GF}(2^8)$. Even if some candidate restored values are attacked, other unattacked restored values could also reconstruct secret information. Figure 2 shows the flowchart of secret information backup and encoding strategy.

Figure 2. The flowchart of Secret information backup and encoding strategy


Secret byte sequence
$\boldsymbol{P}_S = (p_i)_l$
Secret information backup strategy based on Lagrange polynomial interpolation in GF($2^8$)
Backup byte sequence
$\boldsymbol{P}_B = (p_i^B)_{l_1}$
Random encoding strategy based on codebook extension
Backup index sequence
$\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$

In Figure 2, $\boldsymbol{P}_S = (p_i)_l$ is backed up as the backup byte sequence $\boldsymbol{P}_B = (p_i^B)_{l_1}$ by the Lagrange (2,4) polynomial interpolation over $\mathrm{GF}(2^8)$, where $l_1 = 4l$. Then, $\boldsymbol{P}_B$ is encoded to $\boldsymbol{P}_{ID}$ using the random code strategy based on codebook extension to avoid direct transmission of $\boldsymbol{P}_B$ and the interval extension is used to authenticate $\forall p_i^B \in \boldsymbol{P}_B$.

In $\mathrm{GF}(2^8)$, $(k,n)$ threshold secret sharing scheme (Ou-yang et al., 2017; Le et al. 2018; Shao et al., 2019) is defined as Equation(1).

$$f(\dot{o}) = (\dot{r}_0 + \dot{r}_1\dot{o} + \dot{r}_2\dot{o}^2 + \cdots + \dot{r}_{k-1}\dot{o}^{k-1}) \bmod \dot{p} \tag{1}$$

where $\dot{p}$ is a primitive integer polynomial over $\mathrm{GF}(2^8)$. $o$ is a participant number, $r_0, r_1, \cdots, r_{k-1} \in \{0, 1, \cdots, p-1\}$ are polynomial coefficients and $\dot{o}, \dot{r}_0, \dot{r}_1, \cdots, \dot{r}_{k-1}$ are the integer polynomials of $o, r_0, r_1, \cdots, r_{k-1}$ in $\mathrm{GF}(2^8)$ where $r_0, r_1, \cdots, r_{k-1}$ are usually expressed as secrets or authentication information (Ou-yang et al., 2017; Le et al. 2018; Shao et al., 2019). The unique integers $o_i, i = 0, 1, \cdots, n-1$ can be generated as participant numbers and $o_i, i = 0, 1, \cdots, n-1$ can be substituted into Equation(1) to obtain $n$ shares $o_i', i = 0, 1, \cdots, n-1$. In restoration, if $l (l \geq k)$ shares $\dot{o}_i' = f(\dot{o}_i)$ s are obtained arbitrarily, Equation(2) can be employed to reconstruct the interpolation polynomial $f(\dot{o})$ and get $r_0, r_1, \cdots, r_{k-1}$.

$$f(\dot{o}) = \left( \sum_{i=0}^{l} \left( f(\dot{o}_i) \prod_{j=1, j \neq i}^{l} (\dot{x} - \dot{o}_j)(\dot{o}_i - \dot{o}_j)_{\dot{p}}^{-1} \right) \right) \bmod \dot{p} \tag{2}$$

where $(\dot{o}_i - \dot{o}_j)_{\dot{p}}^{-1}$ is the multiplicative inverse of $\dot{o}_i - \dot{o}_j$ over $\mathrm{GF}(2^8)$. If $l < k$, $f(\dot{o})$ cannot be determined by Equation(2). Therefore, $r_0, r_1, \cdots, r_{k-1}$ cannot be obtained.

Based on the above method, $r_0$ can be replaced with $\forall p_i \in \boldsymbol{P}_S$, and $r_1, r_2, \cdots, r_{k-1}$ can be used to authenticate $p_i$. According to Equation(1), $n$ shares are regarded as backup bytes for each $p_i$. In restoration, $\mathrm{C}_n^k$ candidate restored values are obtained by substituting $k$ shares arbitrarily into

Equation(2) to find the most reliable $p_i$. Therefore, it enables this method to be self-repairing that there are still several correct candidate restored values even if some candidate restored values are attacked.

However, $\boldsymbol{P}_S=(p_i)_l$ will transform into $\boldsymbol{P}_B=(p_i^B)_{l_1}$ where $l_1=n\cdot l$. The higher $n$ is, the lengthier $\boldsymbol{P}_B=(p_i^B)_{l_1}$ is. It will enlarge the generated test paper and further reduce hiding efficiency of the proposed method. Therefore, $n$ should be as small as possible. However, if $n=3$, $k$ can only be set to 2, the number of candidate restored values is only $\mathrm{C}_3^2=3$. It will reduce the ability of anti-attack and self-repair. To avoid this, this paper restricts $(k,n)$ to $(2,4)$. Thus, the expansion rate is only 4, and there are $\mathrm{C}_4^2=6$ candidate restored values for $\forall p_i\in\boldsymbol{P}_S$. Algorithm 1 shows the proposed secret information backup strategy.

Algorithm 1: Secret Information Backup Strategy

**Input:** User keys $key$, secret byte sequence $\boldsymbol{P}_S=(p_i)_l$.

**Output:** Backup byte sequence $\boldsymbol{P}_B=(p_i^B)_{l_1}$.

**Step 1:** Initialize $\boldsymbol{P}_B=\phi$ and $i=0$.

**Step 2:** Use $key$ to generate a random pair $(x,y),x,y\in(0,1)$ iteratively and employ Equation(3) to generate two sharing coefficients $r_0,r_1$ where $r_1$ is regarded as the authentication variable of $r_0$.

$$\begin{aligned}&r_0=p_i\\&r_1=(p_i+\lfloor(x\times y)\times10^{10}\rfloor)\bmod 256\end{aligned}\tag{3}$$

**Step 3:** Use $key$ to generate four random pairs $(x,y)$s iteratively and transform them into four different participant numbers $o_0,o_1,o_2,o_3\in\{1,2,\cdots,255\}$. Then construct polynomial interpolation shown as Equation(4).

$$f(\dot{o})=(\dot{r}_0+\dot{r}_1\dot{o})\bmod\dot{p}\tag{4}$$

**Step 4:** Substitute $\dot{o}_0,\dot{o}_1,\dot{o}_2,\dot{o}_3$ into Equation(4) successively to generate four shares $\dot{o}_0',\dot{o}_1',\dot{o}_2',\dot{o}_3'$. Insert $o_0',o_1',o_2',o_3'$ into $\boldsymbol{P}_B$.

**Step 5:** Repeat Step 2 - Step 4 until $i=l$.

To avoid the direct transmission of secret information and improve the authentication accuracy of extracted data, the random codebook strategy (Lu et al., 2018) is employed to encode $\boldsymbol{P}_B$, and further extend $\boldsymbol{P}_B=(p_i^B)_{l_1}$ to a larger interval. The random codebook encoding strategy is given as Algorithm 2.

Algorithm 2: The Random Encoding Strategy

**Input:** User keys $key$, the initialized codebook sequence $\boldsymbol{M}$ which is a permutation of $(0,1,\cdots,383)$ and the backup byte sequence $\boldsymbol{P}_B=(p_i^B)_{l_1}$.

**Output:** The coded index sequence $\boldsymbol{P}_{ID}=(p_i^{ID})_{l_1}$.

**Step 1:** Initialize $\boldsymbol{P}_{ID}=\phi$ and $i=0$.

**Step 2:** Use $key$ to generate a random number sequence $\boldsymbol{X}=(x_k)_{384}$, then arrange $\boldsymbol{X}$ to $\boldsymbol{X}'$ in descending order, and scramble $\boldsymbol{M}$ to $\boldsymbol{M}'=(m_i')_{384}$ by Equation(5) according to the relationship between $\boldsymbol{X}$ and $\boldsymbol{X}'$.

$$\boldsymbol{M}\xrightarrow{X\to X'}\boldsymbol{M}'\tag{5}$$

**Step 3:** Transform $p_i^B$ into $p_i^{ID}$ by Equation(6), and let $\boldsymbol{M} = \boldsymbol{M}'$, $i = i + 1$.

$$p_i^{ID} = \text{index}(\boldsymbol{M}', p_i^B) \tag{6}$$

where $\text{index}(\boldsymbol{M}', p_i^B)$ is used to obtain the index of $p_i^B$ in $\boldsymbol{M}'$.

**Step 4:** Repeat Step 2 - Step 3 until $i = l_1$.

Based on Algorithm 2, $\forall p_i^B \in [0,255]$ is transformed into $p_i^{ID} \in [0,383]$. If Equation(6) is replaced with Equation(7), $p_i^B$ is obtained where $\text{get}(\boldsymbol{M}', p_i^{ID})$ is employed to obtain the element $p_i^B$ whose index is $p_i^{ID}$ in $\boldsymbol{M}'$.

$$p_i^B = \text{get}(\boldsymbol{M}', p_i^{ID}) \tag{7}$$

In this paper, one multiple-choice is used to hide one byte $\forall p_i^B \in [0,255]$. Because the maximum hidden capacity of a multiple-choice is one integer in [0,383], $p_i^B$ can be extended into [0,383] at most. Therefore $\boldsymbol{M}$ should be a permutation of $(0,1,\cdots,383)$. Moreover, the maximum hidden capacity will be discussed later. In restoration, if $p_i^B$ recovered by $p_i^{ID}$ is not in [0,255], $p_i^B$ has been attacked. In other words, the extended interval [256,383] is used to identify attacks. Denote this strategy as the codebook extension authentication. For simplification, Algorithm 2 and its decoding version are denoted as Equation(8) and Equation(9), respectively.

$$\boldsymbol{P}_{ID} = \text{Rcode}(\boldsymbol{P}_B, key, \boldsymbol{M}) \tag{8}$$

$$\boldsymbol{P}_B = \text{Rcode}^{-1}(\boldsymbol{P}_{ID}, key, \boldsymbol{M}) \tag{9}$$

### The Proposed Test Disguise Strategy

In Lu et al. (2018), the hidden capacity of a single test question is only 2.5 bits. To improve the hidden capacity, the authors limit question type to arithmetic multiple-choice in [0,200] and employ 24 different candidate answer orders and the 4-bit question stem hash value together to express one number which is in [0,383]. For the mapping method of stem hash values, the authors create a test database divided into eight sets and generate the hash values of stems by their set indexes. The strategy of test database creation and stem hash mapping is given as follows.

Suppose $\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$ is the backup index sequence to be hidden, where $\forall p_i^{ID} \in [0,383]$. $\mathbf{F} = (\boldsymbol{F}_k)_8$ is the question stem database divided into 8 sets, $\boldsymbol{F}_k = (f_i^k)_{n_k}$ is the $k$th set of $\mathbf{F}$, where $f_i^k$ is the $i$th question stem in $\boldsymbol{F}_k$, and $n_k$ is the number of question stems in $\boldsymbol{F}_k$. $\mathbf{E} = (\boldsymbol{E}_k)_8$ is the stem hash matrix to mark hash values of $\mathbf{F}$, $\boldsymbol{E}_k = (e_i^k)_{n_k}$ is the $k$th stem hash sequence, $e_i^k$ is the corresponding hash value of $f_i^k$. $\mathbf{G} = (\boldsymbol{G}_k)_8$ is the stem mark matrix, and $\boldsymbol{G}_k = (g_i^k)_{n_k}$ is employed to identify the selected stems where if $g_i^k = 1$, is selected; otherwise, is not. In this paper, the combination of $\mathbf{F}$, $\mathbf{E}$ and $\mathbf{G}$ is the corresponding test database.

For $\forall i, j \in [0,99]$, the set index $k$ of question stem $f = "i + j = "$ can be calculated according to Equation(10). Then, $f = "i + j = "$ is added into $\boldsymbol{F}_k$. Moreover, the hash value $e$ of $f$ is calculated by Equation(11). Then, $e$ is added into $\boldsymbol{E}_k$ . $\begin{aligned} answer &= i+j \\ k &= answer \bmod 8 \end{aligned}$ (10) $e = \mathrm{GetValue}(\mathrm{MD5}(f), k)$ (11)

where $\mathrm{MD5}(f)$ is used to get MD5 value of question stem and $\mathrm{GetValue}(\mathrm{MD5}(f), k)$ is used to get the 16-decimal value in $f$ 's MD5.

Equation(10) and Equation(11) are the corresponding strategy of test database creation and stem hash mapping. For an HM (hidden multiple-choice), in disguise process, one 16-decimal hash value and one 24-decimal number generated by its candidate answer order are employed to express $u, v$ which are two parts of a backup index $p_i^{ID} \in \boldsymbol{P}_{ID}$, respectively. Moreover, the sparse distribution of HMs in the stego test paper is important to escape attacks. Therefore, when HMS (hidden multiple-choice sequence) is obtained, a certain number of NMs (non-hidden multiple-choices) are generated, and they are combined with HMS as the stego test paper. Finally, all multiple-choices in the test paper are scrambled by $key$ to destroy the position relationship between HMs and NMs. The corresponding test disguise strategy can be described as following.

Let $\boldsymbol{L}_1 = (st_i = (T_i, a_i, b_i, c_i, d_i))_{l_1}$ and $\boldsymbol{L}_2 = (st_i = (T_i, a_i, b_i, c_i, d_i))_{l_1}$ be HMS and NMS (non-hidden multiple-choice sequence) respectively, where $T_i$ is a question stem, $a_i, b_i, c_i, d_i$ are four candidate answers of $st_i$ , $l_1, l_2$ are the length of $\boldsymbol{L}_1, \boldsymbol{L}_2$ ,respectively. Suppose $rate$ is the ratio of NMs to HMs for controlling the number of NMs. And $rate$ can be designated by both communication sides.

Initialize HMS $\boldsymbol{L}_1 = \phi$ , $i = 0$ . Traverse all elements $p_i^{ID}$ in $\boldsymbol{P}_{ID}$ . For $\forall p_i^{ID} \in \boldsymbol{P}_{ID}$ , $p_i^{ID}$ is transformed to a 16-decimal number $u$ and a 24-decimal number $v$ using Equation(12). Then stem hash values and candidate answer orders can be employed to express them, respectively.

$$\begin{aligned} u &= p_i^{ID} \bmod 16 \\ v &= \left\lfloor p_i^{ID} / 16 \right\rfloor \end{aligned} \qquad (12)$$

For expressing $u$ , a stem whose hash value equals $u$ in the appropriate set of test database can be found to express $u$ . In this paper, a random pair $(x, y)$ is iteratively generated by $key$ . Then, $k, j$ are calculated by Equation(13).

$$\begin{aligned} k &= \left\lfloor (x+y) \times 10^{10} \right\rfloor \bmod 8 \\ j &= \left\lfloor x \times 10^{10} \right\rfloor \bmod n_k \end{aligned} \qquad (13)$$

For elements of $\boldsymbol{E}_k = (e_j^k)_{n_k}$ , $\boldsymbol{G}_k = (g_j^k)_{n_k}$ and $\boldsymbol{F}_k = (f_i^k)_{n_k}$ , if $e_j^k \neq u$ or $g_j^k = 1$ , make $j = (j+1) \bmod n_k$ , and retrieve the test database again until $e_j^k = u, g_j^k = 0$ . Then, calculate the answer $ans_i$ of $f_i^k$ and make $T_i = f_i^k, g_j^k = 1$ .

When the stem $T_i$ whose hash value equals $u$ is selected, a candidate answer list can be generated to express $v$ . Firstly, a random pair $(x, y)$ is iteratively generated by $key$ . Then, $id$ is calculated by Equation(14). Moreover, $ans_i - 1, ans_i, ans_i + 1, ans_i + 2$ are substituted into Equation(15) to generate candidate answers $a_i, b_i, c_i, d_i$ .

$$id = \left( v + \left\lfloor \sqrt{xy} \times 10^{10} \right\rfloor \right) \bmod 24 \qquad (14)$$

where the reason for the combination of $v$ and $(x, y)$ is to destroy the mapping relationship between $v$ and candidate answer orders.

$$(a_i, b_i, c_i, d_i) = \text{Dic}(ans_i - 1, ans_i, ans_i + 1, ans_i + 2, id) \tag{15}$$

where Dic() is used to generate a candidate answer list with an appropriate order. Suppose $ans_i = 2, id = 1$ and four candidate answers are 1,2,3,4. There will be 24 different arrangements $((1,2,3,4),(1,2,4,3),\cdots,(4,3,2,1))$, then choose (1,2,4,3) whose index is $id = 1$ as its candidate answer order.

Add HM $st_i = (T_i, a_i, b_i, c_i, d_i)$ into $\boldsymbol{L}_1$. Repeat these steps until $\boldsymbol{P}_{ID}$ is traversed completely, and HMS $\boldsymbol{L}_1 = (st_i)_{l_1}$ is obtained.

Furthermore, $l_2$ NMs is generated by Equation(16) according to the ratio variable $rate$. Then, add $l_2$ NMs to the end of $\boldsymbol{L}_1$.

$$l_2 = \lfloor l_1 \times rate \rfloor \tag{16}$$

Finally, the authors use $key$ to scramble $\boldsymbol{L}_1 \| \boldsymbol{L}_2$ into $\boldsymbol{L}''$ and regard $\boldsymbol{L}''$ as the final stego test paper $\boldsymbol{L} = (st_i)_{l_3}$, where $l_3 = l_1 + l_2$. For simplification, denote the above-mentioned test disguise method as Equation(17).

$$\boldsymbol{L} = \text{Tdis}(key, \boldsymbol{P}_{ID}, \mathbf{F}, \mathbf{G}, \mathbf{E}, rate) \tag{17}$$

Algorithm 3 describes the corresponding test extraction process of Equation(17). For simplification, denote the test extraction process as Equation(18).

$$\boldsymbol{P}_{ID} = \text{Tdis}^{-1}(key, \boldsymbol{L}, rate) \tag{18}$$

Algorithm 3: Test Extraction Strategy Combined with Stem Hash and Candidate Answer Orders

**Input:** User keys $key$, the ratio variable $rate$, and received the test paper $\boldsymbol{L}' = (st_i')_{l_3}$.

**Output:** Backup index sequence $\boldsymbol{P}'_{ID} = (p_i'^{ID})_{l_1}$.

**Step 1:** Use Equation(19) to get the number $l_2$ of NMs.

$$l_2 = \lceil l_3 / (1 + rate) \rceil \tag{19}$$

**Step 2:** Employ $key$ to descramble $\boldsymbol{L}'$ to $\boldsymbol{L}'''$. Then, remove $l_2$ NMs from the end of $\boldsymbol{L}'''$ to obtain HMS $\boldsymbol{L}_1$.

**Step 3:** Initialize $\boldsymbol{P}'_{ID} = \phi, i = 0$.

**Step 4:** Get the question stem $T_i$ and the answer $ans_i$ of $st_i$. Use Equation(20) to calculate the set index $k$ of $st_i$, and substitute $\text{MD5}(T_i), k$ into Equation(11) to obtain the hash value $e$ of $st_i$. Make $u = e$.

$$k = ans_i \bmod 8 \tag{20}$$

**Step 5:** Use $key$ to iteratively generate a random pair , extract the candidate answers $a_i, b_i, c_i, d_i$ of $st_i$. Then employ Equation(21) to get $id$. Finally, $v$ is calculated by Equation(22).

$$id = \mathrm{Dic}^{-1}(a_i, b_i, c_i, d_i) \tag{21}$$

where $\mathrm{Dicr}^{-1}()$, the inverse function of $\mathrm{Dicr}()$, is employed to get the index of candidate answer order.

$$v = (id - \lfloor x_i y_i \times 10^{10} \rfloor) \bmod 24 \tag{22}$$

**Step 6:** Use Equation(23) to get $p_i'^{ID}$ and add it into $\boldsymbol{P}_{ID}'$. Make $i = i + 1$.

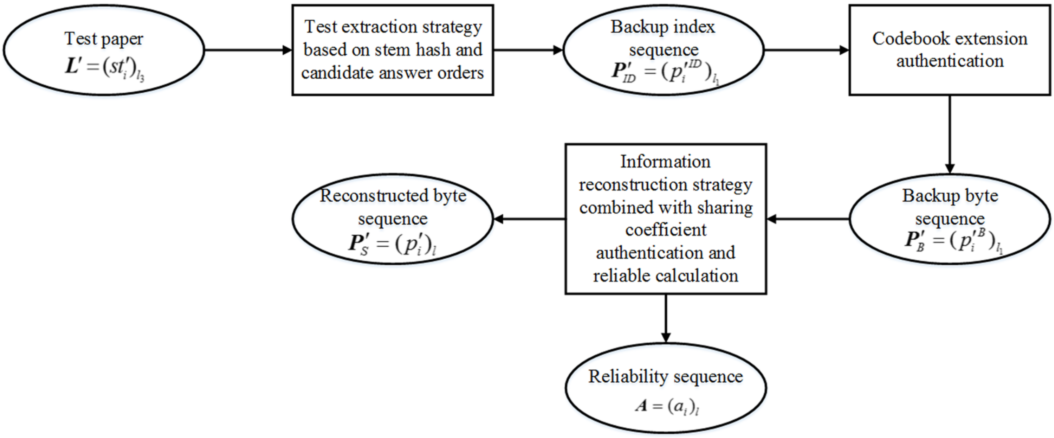
$$p_i'^{ID} = u + 16 \times v \tag{23}$$

**Step 7:** Repeat Step 4 - Step6 until $i = l_1$.

## THE PROPOSED INFORMATION RECONSTRUCTION STRATEGY

To avoid the absence of authentication strategy and reliable calculation, the authors employ double authentications to distinguish extracted information and obtain six candidate restored values of $\forall p_i \in \boldsymbol{P}_S$. Then, the authors choose the most frequent value as the final restored value to reconstruct secret information, and calculate its reliability according to its frequency.

Figure 3. The flowchart of information reconstruction combined with double authentications and reliable calculation


Test paper
$\boldsymbol{L}' = (st_i')_{l_3}$
Test extraction strategy based on stem hash and candidate answer orders
Backup index sequence
$\boldsymbol{P}_{ID}' = (p_i'^{ID})_{l_1}$
Codebook extension authentication
Backup byte sequence
$\boldsymbol{P}_B' = (p_i'^{B})_{l_1}$
Information reconstruction strategy combined with sharing coefficient authentication and reliable calculation
Reconstructed byte sequence
$\boldsymbol{P}_S' = (p_i')_l$
Reliability sequence
$\boldsymbol{A} = (a_i)_l$

In Figure 3, the test extraction strategy based on stem hash and candidate answer orders (Equation(18)) is used to extract $\boldsymbol{P}_{ID}' = (p_i'^{ID})_{l_1}$ from $\boldsymbol{L}' = (st_i')_{l_3}$. Then, the codebook extension authentication (Equation(9)) is employed to decode $\boldsymbol{P}_{ID}' = (p_i'^{ID})_{l_1}$ to $\boldsymbol{P}_B' = (p_i'^{B})_{l_1}$ and set $p_i'^{B} \in [256, 383]$ to -1 to mark it as an error backup byte. Finally, IRSCSCARC (the information reconstruction strategy combined with sharing coefficient authentication and reliable calculation) is applied to reconstruct $\boldsymbol{P}_S' = (p_i')_l$ by $\boldsymbol{P}_B'$, and generate $\boldsymbol{A} = (a_i)_l$ to mark the reliability of elements in $\boldsymbol{P}_S'$.

For IRSCSCARC, the main idea is to reconstruct the interpolation polynomial $f(\dot{o}) = (\dot{r}_0 + \dot{r}_1\dot{o}) \bmod \dot{p}$ to obtain candidate restored values which are authenticated by the sharing coefficient authentication. Then, choose the most reliable candidate restored values to reconstruct secret information and calculate the reliability of secret information. There are two key strategies in IRSCSCARC: (1) the sharing coefficient authentication and (2) the reliable calculation.

For sharing coefficient authentication, the main idea is that two elements of $\boldsymbol{P}_B'$ generated by Equation(9) are combined to reconstruct $f(\dot{o}) = (\dot{r}_0 + \dot{r}_1\dot{o}) \bmod \dot{p}$ . Then, $r_1$ is further used to authenticate candidate restored value $r_0$ .

Suppose two backup bytes involved in calculation are $o_m', o_n' \in \boldsymbol{P}_B'$, respectively. The corresponding participant numbers are $o_m, o_n$ , respectively. The following strategies can be employed to obtain a candidate restored value and further authenticate it by these two backup vectors $(o_m, o_m'), (o_n, o_n')$ .

If $o_m' = -1 \vee o_n' = -1$ , it is determined that at least one of $o_m', o_n'$ fails to pass the codebook extension authentication. Therefore the obtained candidate restored value $b$ is most likely to be wrong. In this paper, set $b = -1$ directly.

If $o_m' \neq -1 \wedge o_n' \neq -1$ , it is determined that $o_m', o_n'$ have passed the codebook extension authentication. Thus, $f(\dot{o}) = (\dot{r}_0 + \dot{r}_1\dot{o}) \bmod \dot{p}$ is reconstructed by $(\dot{o}_m, \dot{o}_m'), (\dot{o}_n, \dot{o}_n')$ using Equation(2).

In the construction of $f(\dot{o}) = (\dot{r}_0 + \dot{r}_1\dot{o}) \bmod \dot{p}$ , $r_1$ is set to the combination of $r_0$ and a random number generated by $key$. Denote the random number as $disturb$. Therefore, if $r_0, r_1$ satisfy Equation(24), $r_0$ is regarded as a correct value, and set $b = r_0$; otherwise, $r_0$ has been attacked, and set $b = -1$ to mark it as an unreliable candidate restored value.

$$r_0 = (r_1 - disturb) \bmod 256 \tag{24}$$

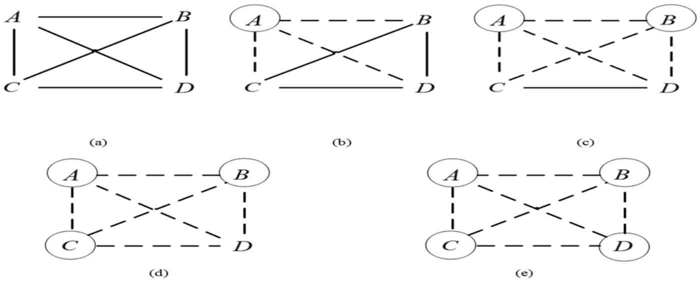
For conciseness, call these strategies the sharing coefficient authentication and denote it as Equation(25).

$$b = \mathrm{Sca}((o_m, o_m'), (o_n, o_n'), disturb) \tag{25}$$

By this way, four bytes $p_{4i}'^B, p_{4i+1}'^B, p_{4i+2}'^B, p_{4i+3}'^B$ are read from $\boldsymbol{P}_B'$ successively. Then, use $key$ to generate $disturb$ and four participant numbers $o_0, o_1, o_2, o_3$. Finally, combine two backup vectors in $\{(o_0, p_{4i}'^B), (o_1, p_{4i+1}'^B), (o_2, p_{4i+2}'^B), (o_3, p_{4i+3}'^B)\}$ arbitrarily to get $\mathrm{C}_4^2$=6 candidate restored values $b_0, b_1, b_2, b_3, b_4, b_5$ by Equation(25).

For the reliable calculation, the most frequent value in $\{b_i \mid i = 0, 1, \cdots, 5, b_i \neq -1\}$ are chosen as $b_{\text{freq}}$ .Figure 4 shows five statuses of final value choice.

**Figure 4. Five statuses of the most reliable value choice, where, Figure 4a – Figure 4e are status 1 - status 5, respectively**



In Figure 4, $A, B, C, D$ are four backup vectors respectively, the connection between two backup vectors is the candidate restored value generated by them. The circle represents that the backup vector in it has been attacked, and the dotted line represents the corresponding value which is very likely to be wrong. Let $appearence(b_{\text{freq}})$ denote the appearance times of $b_{\text{freq}}$.

1. If $appearence(b_{\text{freq}}) = 6$, it is Status 1 because $b_i, i = 0,1,\cdots,5$ are all authenticated as correct values. Set the final value $p_i' = b_{\text{freq}}$ and its reliability $a_i = 3$ to mark $p_i'$ as a very reliable value.
2. If $3 \le appearence(b_{\text{freq}}) < 6$, it is Status 2. Set $p_i' = b_{\text{freq}}$ and its reliability $a_i = 2$ to mark as a reliable value.
3. If $1 \le appearence(b_{\text{freq}}) < 3$, It is only regarded as one of Status 3, Status 4 and Status 5, because it is possible that some error candidate restored values which are mistakenly authenticated as correct values are involved in this calculation. If $b_{\text{freq}}$ is unique, set $p_i' = b_{\text{freq}}$ and its reliability $a_i = 1$ to mark as an unreliable value; if is not unique, this method cannot distinguish which value is the correct one. Therefore, randomly choose one candidate restored value that is not -1 and make $p_i'$ equal to it. Then set $a_i = 1$.
4. If doesn't exist, it is Status 5 because $b_i = -1, i = 0,1,\cdots,5$. Then, assign $p_i'$ randomly and set $key$ $a_i = 0$ to mark it as a very unreliable value.

For simplification, denote the reliable calculation strategy as Equation(26).

$$(p_i', a_i) = \text{Choice}(b_0, b_1, b_2, b_3, b_4, b_5) \tag{26}$$

The combination of Equation(25) and Equation(26) is IRSCSCARC shown as Algorithm 4.
Algorithm 4: IRSCSCARC

**Input:** Backup byte sequence $\boldsymbol{P}_B' = (p_i'^B)_{l_1}$ and user keys $key$.

**Output:** Restored byte sequence $\boldsymbol{P}_S' = (p_i')_l$ and $\boldsymbol{A} = (a_i)_l$.

**Step 1:** Initialize $\boldsymbol{P}_S' = \phi, i = 0$.

**Step 2:** Use $key$ to generate $(x, y)$ iteratively and employ Equation(27) to generate $disturb$.

$$disturb = \left\lfloor (x_k \times y_k) \times 10^{10} \right\rfloor \bmod 256 \tag{27}$$

**Step 3:** Use $key$ to generate four random pairs $(x, y)$ s and transform them into four different participant numbers $o_0, o_1, o_2, o_3$. Then read $p_{4i}'^B, p_{4i+1}'^B, p_{4i+2}'^B, p_{4i+3}'^B$ from $\boldsymbol{P}_B'$.

**Step 4:** Obtain six candidate restored values $b_i, i = 0,1,\cdots,5$ by the combination of two elements in $\{(o_0, p_{4i}'^B), (o_1, p_{4i+1}'^B), (o_2, p_{4i+2}'^B), (o_3, p_{4i+3}'^B)\}$ using Equation(25).

**Step 5:** Substitute $b_i, i = 0,1,\cdots,5$ into Equation(26) to generate final value $p_i'$ and its reliability $a_i$. Add $p_i'$ into $\boldsymbol{P}_S'$ and $a_i$ into $\boldsymbol{A}$.

**Step 6:** Repeat Step2 - Step 5 until $i = l$.

Note that double authentication strategies are necessary for this information reconstruction process. Error candidate restored values will be involved in this process with the absence of the codebook extension authentication and the sharing coefficient authentication. It may decrease the quality of restored information and the accuracy of reliable calculation.

## THE PROPOSED DISGUISE AND RESTORING ALGORITHM

Algorithm 5: The Entire Test Paper Disguise Strategy

**Input:** Secret byte sequence $\boldsymbol{P}_S = (p_i)_l$ , user keys $key$ , the ratio variable $rate$ , and the initialized codebook sequence $\boldsymbol{M}$ .

**Output:** The stego paper $\boldsymbol{L} = (st_i)_{l_3}$ .

**Step 1:** $\boldsymbol{P}_S = (p_i)_l$ is backed up as the backup byte sequence $\boldsymbol{P}_B = (p_i^B)_{l_1}$ by $key$ using Algorithm 1.

**Step 2:** Transform $\boldsymbol{P}_B = (p_i^B)_{l_1}$ into the backup index sequence $\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$ by $key$ using Algorithm 2.

**Step 3:** Generate the test database consisting of $\mathbf{F}$ , $\mathbf{E}$ and $\mathbf{G}$ .

**Step 4:** Use Equation(17) to disguise $\boldsymbol{P}_{ID} = (p_i^{ID})_{l_1}$ as the stego test paper $\boldsymbol{L} = (st_i)_{l_3}$ by $key$ .

Algorithm 6: The Entire Secret Information Restoring Strategy

**Input:** The received stego test paper $\boldsymbol{L}' = (st_i)_{l_3}$ , user keys $key$ , the ratio variable $rate$ , and the initialized codebook sequence $\boldsymbol{M}$ .

**Output:** The restored byte sequence $\boldsymbol{P}_S' = (p_i')_l$ and the reliability sequence $\boldsymbol{A} = (a_i)_l$ .

**Step 1:** Use Equation(18) to extract the backup index sequence $\boldsymbol{P}_{ID}' = (p_i'^{ID})_{l_1}$ from $\boldsymbol{L}' = (st_i)_{l_3}$ by $key$ .

**Step 2:** Employ Equation(9) to decode $\boldsymbol{P}_{ID}' = (p_i'^{ID})_{l_1}$ as the backup byte sequence $\boldsymbol{P}_B' = (p_i'^B)_{l_1}$ .

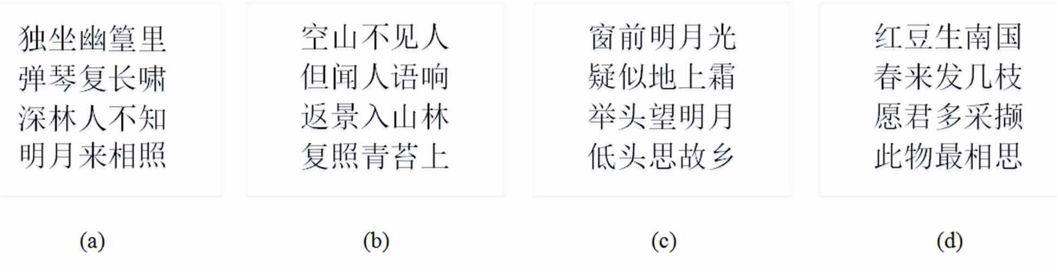
**Step 3:** Use Algorithm 4 to reconstruct the restored byte sequence $\boldsymbol{P}_S' = (p_i')_l$ and generates its reliability value sequence $\boldsymbol{A} = (a_i)_l$ by $key$ and $\boldsymbol{P}_B'$ .

## EXPERIMENT

The experimental test environment is Windows 10. The CPU is an Intel(R) Core (TM) i5-6600. The CPU main frequency is 3.31 GHz. The memory size is 8.00 GB. The programming language implemented is JAVA.

The performance of the proposed method is measured using several factors such as practicability, authentication accuracy, and self-repair ability. Moreover, a comprehensive comparison of SCIH and Lu et al. (2018) is made based on hidden capacity and database design. Fig.5 shows the secret information used for the experiment.

Figure 5. Secret information



独坐幽篁里
弹琴复长啸
深林人不知
明月来相照

(a)

空山不见人
但闻人语响
返景入山林
复照青苔上

(b)

窗前明月光
疑似地上霜
举头望明月
低头思故乡

(c)

红豆生南国
春来发几枝
愿君多采撷
此物最相思

(d)

Figure 5a – Figure 5b are four ancient Chinese poems: zhuliguan, luzhai, jingyesi and xiangsi. In this experiment, we transform every poem into 40 secret bytes by mapping a Chinese character to 2-byte ASCII.

In this paper, error rate ( ER ) is used to evaluate the quality of reconstructed information. ER is shown as Equation(28).

$$\text{ER} = \frac{l_{error}}{l_{total}} \times 100\% \tag{28}$$

where $l_{total}$ is the number of reconstructed bytes, $l_{error}$ is the number of error reconstructed bytes.

The hidden rate ( HR ) is employed to evaluate the hidden capacity. HR is shown as Equation(29).

$$\text{HR} = \frac{l_S}{l_C} \tag{29}$$

where $l_S$ is the secret information size in bytes (B), $l_C$ is the carrier size in kilobytes(KB).

The authentication success rate ( ASR ) is used to evaluate the authentication accuracy. ASR is shown as Equation(30).

$$\text{ASR} = \frac{l_{success}}{l_{total}} \tag{30}$$

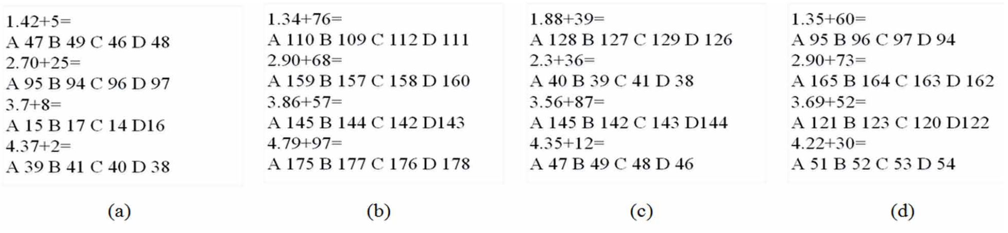
where $l_{success}$ is the number of restored bytes authenticated successfully, $l_{total}$ is the number of total restored bytes.

In this experiment, the authors set the ratio variable $rate = 0.2$ , the initialized codebook sequence, $\boldsymbol{M} = (0,1,\cdots,383)$ , and the selected primitive polynomial over $\text{GF}(2^8)$ is $t^8 + t^6 + t^5 + t^4 + 1$ , where the primitive integer polynomial $\dot{p} = 3\dot{6}9$ . Meanwhile it is a small probability event that error candidate restored values are authenticated as correct values and further involved in the reliable calculation. Therefore, the authors only regard the restored bytes whose reliabilities are 0 as error restored bytes. Moreover, the authors employ the mutual disturbance of double logistic mapping strategy (Lu et al., 2018) to generate random pairs $(x, y)$ s for the randomness of random numbers. Therefore, the user keys $key$ are $x_{init}, \mu_{init}, t, IT$ , where $x_{init}, \mu_{init}$ are the initialized iterative parameters, $t$ is the control threshold parameter and $IT$ is the parameter to eliminate transient effect.

## Practicability Verification Experiment

To examine the practicability of the proposed method, Algorithm 5 is used to disguise Figure 5a – Figure 5d as the stego test papers by different user keys. Figure 6 shows the parts of these stego test papers. Then the restored information is shown as Figure 7a – Figure 7d can be extracted from test papers successfully. The experimental data and the details of stego test papers are given in Table 1 and Table 2 respectively.

Figure 6. The stego test paper parts generated in practicability verification experiment, where Figure 6a – Figure 6d are the stego test paper parts of Group 1 - Group 4 in Table 1, respectively



1.42+5=
A 47 B 49 C 46 D 48
2.70+25=
A 95 B 94 C 96 D 97
3.7+8=
A 15 B 17 C 14 D16
4.37+2=
A 39 B 41 C 40 D 38

(a)

1.34+76=
A 110 B 109 C 112 D 111
2.90+68=
A 159 B 157 C 158 D 160
3.86+57=
A 145 B 144 C 142 D143
4.79+97=
A 175 B 177 C 176 D 178

(b)

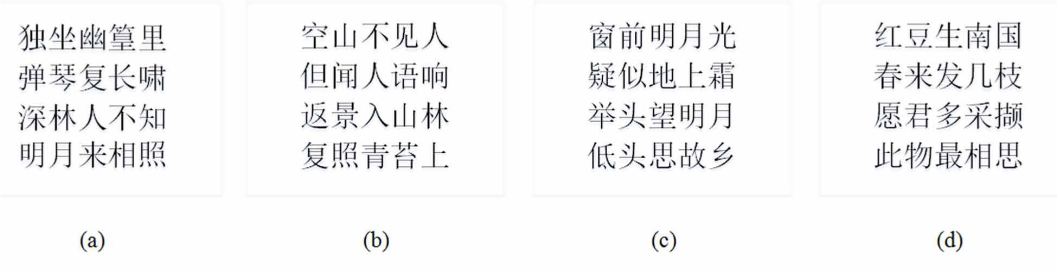
1.88+39=
A 128 B 127 C 129 D 126
2.3+36=
A 40 B 39 C 41 D 38
3.56+87=
A 145 B 142 C 143 D144
4.35+12=
A 47 B 49 C 48 D 46

(c)

1.35+60=
A 95 B 96 C 97 D 94
2.90+73=
A 165 B 164 C 163 D 162
3.69+52=
A 121 B 123 C 120 D122
4.22+30=
A 51 B 52 C 53 D 54

(d)

Figure 7. The restored information in practicability verification experiment, where Figure 7a – Figure 7d are the restored information of Group 1 - Group 4 in Table 1, respectively



独坐幽篁里
弹琴复长啸
深林人不知
明月来相照

空山不见人
但闻人语响
返景入山林
复照青苔上

窗前明月光
疑似地上霜
举头望明月
低头思故乡

红豆生南国
春来发几枝
愿君多采撷
此物最相思

Table 1. Experimental data of practicability verification experiment

| NO. | Secret information | $x_{init}$ | $\mu_{init}$ | $t$ | $IT$ | ER | Executive time(ms) |
|---|---|---|---|---|---|---|---|
| 1 | Figure 5a | 0.758221532132447 | 3.854632158141234 | 0.1 | 20 | 0 | 338 |
| 2 | Figure 5b | 0.569872465168463 | 3.732189613168495 | 0.4 | 60 | 0 | 320 |
| 3 | Figure 5c | 0.246358792515649 | 3.964532456489618 | 0.3 | 40 | 0 | 344 |
| 4 | Figure 5d | 0.975632145201538 | 3.631754563047836 | 0.05 | 30 | 0 | 322 |

Table 2. Details of the stego test papers in Figure 6 and the corresponding HR

| NO. | Test paper | Number of questions | Length of test paper(KB) | Secret information | HR |
|---|---|---|---|---|---|
| 1 | Figure 6a | 192 | 6.45 | Figure 5a | 6.20 |
| 2 | Figure 6b | 192 | 6.48 | Figure 5b | 6.18 |
| 3 | Figure 6c | 192 | 6.42 | Figure 5c | 6.23 |
| 4 | Figure 6d | 192 | 6.44 | Figure 5d | 6.21 |

As shown above, secret information can be disguised as the corresponding stego test papers by different user keys. The secret information hidden in test papers can be restored by correct user keys because of ER . Moreover, the average HR of our method is 6.20.

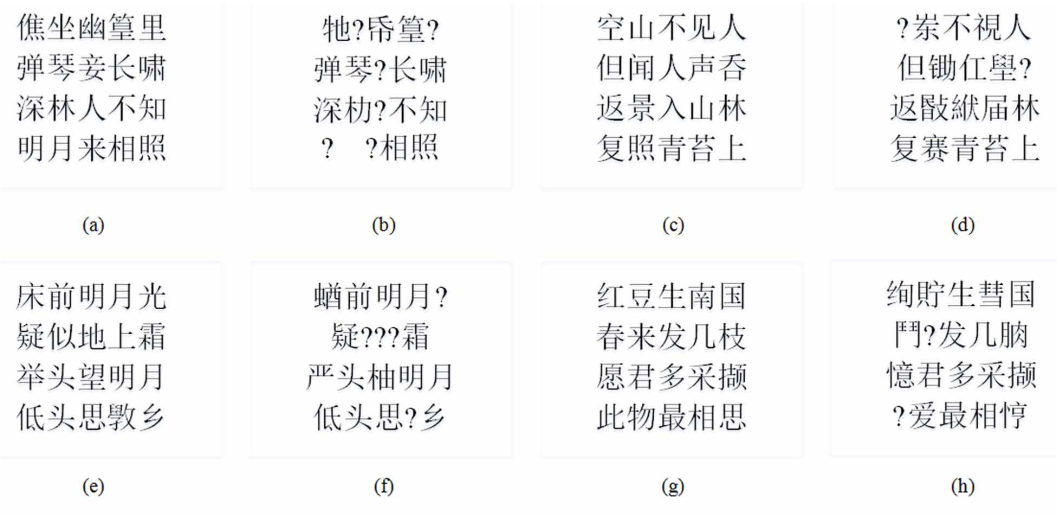
## The Self-Repair and Authentication Ability Experiment

To examine the self-repair ability of the proposed method under attacks and further verify its authentication accuracy, the stego test papers shown in Figure 6 are attacked by different ways given as follows.

1. Attack 1: Change candidate answer orders of multiple-choices.
2. Attack 2: Modify stems of multiple-choices to destroy hash values of them.
3. Attack 3: Delete a certain number of multiple-choices from stego test paper, and add the same amount of NMs at the attack positions.

To examine the self-repair ability and authentication accuracy of the proposed method under Attack 1, the authors attack Figure 6a – Figure 6d by Attack 1 in different scales. Table 3 shows the experimental data. The reconstructed information and authentication image are given as Figure 8 and Figure 9, respectively.

Figure 8. The restored information in the experiment of self-repair ability and authentication accuracy under Attack 1, where Figure 8a – Figure 8h are the restored information of Group 1 - Group 8 in Table 3, respectively



焦坐幽篁里
弹琴妾长啸
深林人不知
明月来相照

(a)

牠?皊篁?
弹琴?长啸
深枋?不知
?　?相照

(b)

空山不见人
但闻人声呑
返景入山林
复照青苔上

(c)

?崈不視人
但锄仜墍?
返敯絥届林
复赛青苔上

(d)

床前明月光
疑似地上霜
举头望明月
低头思敦乡

(e)

蜡前明月?
疑???霜
严头柚明月
低头思?乡

(f)

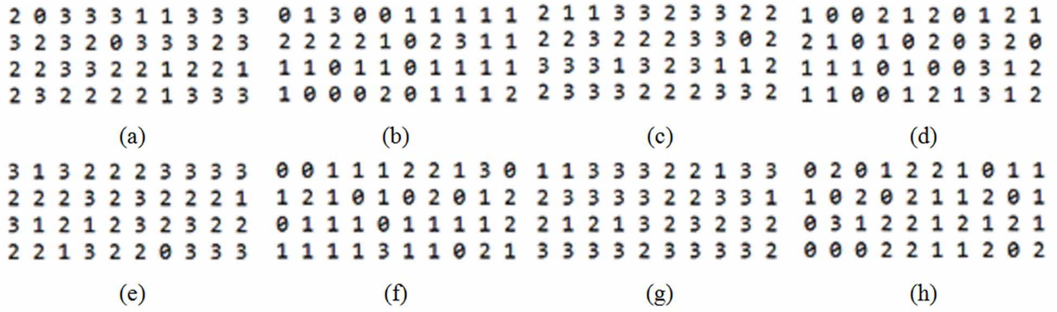
红豆生南国
春来发几枝
愿君多采撷
此物最相思

(g)

绚貯生彗国
閂?发几朒
憶君多采撷
?爱最相悙

(h)

Figure 9 The authentication images in the experiment of self-repair ability and authentication accuracy under Attack 1, where Figure 9a – Figure 9h are the authentication images of Group 1 - Group 4 in Table 3, respectively



```
2 0 3 3 3 1 1 3 3 3
3 2 3 2 0 3 3 3 2 3
2 2 3 3 2 2 1 2 2 1
2 3 2 2 2 2 1 3 3 3
```

(a)

```
0 1 3 0 0 1 1 1 1 1
2 2 2 2 1 0 2 3 1 1
1 1 0 1 1 0 1 1 1 1
1 0 0 0 2 0 1 1 1 2
```

(b)

```
2 1 1 3 3 2 3 3 2 2
2 2 3 2 2 2 3 3 0 2
3 3 3 1 3 2 3 1 1 2
2 3 3 3 2 2 2 3 3 2
```

(c)

```
1 0 0 2 1 2 0 1 2 1
2 1 0 1 0 2 0 3 2 0
1 1 1 0 1 0 0 3 1 2
1 1 0 0 1 2 1 3 1 2
```

(d)

```
3 1 3 2 2 2 3 3 3 3
2 2 2 3 2 3 2 2 2 1
3 1 2 1 2 3 2 3 2 2
2 2 1 3 2 2 0 3 3 3
```

(e)

```
0 0 1 1 1 2 2 1 3 0
1 2 1 0 1 0 2 0 1 2
0 1 1 1 0 1 1 1 1 2
1 1 1 1 3 1 1 0 2 1
```

(f)

```
1 1 3 3 3 2 2 1 3 3
2 3 3 3 3 2 2 3 3 1
2 1 2 1 3 2 3 2 3 2
3 3 3 3 2 3 3 3 3 2
```

(g)

```
0 2 0 1 2 2 1 0 1 1
1 0 2 0 2 1 1 2 0 1
0 3 1 2 2 1 2 1 2 1
0 0 0 2 2 1 1 2 0 2
```

(h)

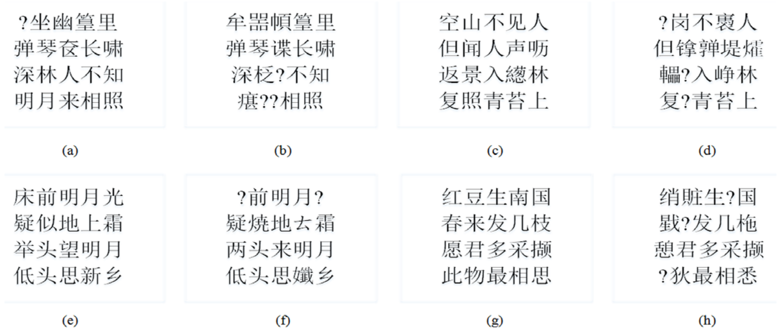
Table 3. Details in the experiment of self-repair ability and authentication accuracy under Attack 1

| NO. | Stego test paper | Attack rate | Number of attacked questions | Number of restored bytes authenticated successfully | ER | ASR |
|---|---|---|---|---|---|---|
| 1 | Figure 6a | 0.2 | 38 | 40 | 5% | 100% |
| 2 | Figure 6a | 0.5 | 96 | 40 | 30 | 100% |
| 3 | Figure 6b | 0.2 | 38 | 40 | 2.5% | 100% |
| 4 | Figure 6b | 0.5 | 96 | 40 | 30% | 100% |
| 5 | Figure 6c | 0.2 | 38 | 40 | 2.5% | 100% |
| 6 | Figure 6c | 0.5 | 96 | 40 | 22.5% | 100% |
| 7 | Figure 6d | 0.2 | 38 | 40 | 0 | 100% |
| 8 | Figure 6d | 0.5 | 96 | 40 | 27.5% | 100% |

In Figure 9, each number represents the reliability of the corresponding restored byte. As shown in Figure 8, Figure 9 and Table 3, even though the stego test papers are attacked by Attack 1 with 0.2 attack rate, ER is only about 0% - 5%. Among them, the restored information in the 7th group of Table 3 is reconstructed without error. Meanwhile, ER is only about 22.5% - 30% when the attack rate is increased to 0.5. It is verified that the proposed method can resist a certain degree of Attack 1 with good self-repair ability. Moreover, ASR=100% in all groups of Table 3. It is verified that the proposed method can eliminate the influence of error candidate restored values on reconstructed information to ensure the quality of it with good authentication accuracy.

To examine the self-repair ability and authentication accuracy of the proposed method under Attack 2, the authors attack Figure 6a – Figure 6d by Attack 2 in different scales. Table 4 shows the experimental data. The reconstructed information and authentication image are given as Figure 10 and Figure 11, respectively.

Figure 10. The restored information in the experiment of self-repair ability and authentication accuracy under Attack 2, where Figure 10a – Figure 10h are the restored information of Group 1 - Group 8 in Table 4, respectively



?坐幽篁里
弹琴夜长啸
深林人不知
明月来相照

(a)

牟嚣幀篁里
弹琴谍长啸
深杈?不知
瘗??相照

(b)

空山不见人
但闻人声呖
返景入繐林
复照青苔上

(c)

?岗不褭人
但镎弹堤熣
轜?入峥林
复?青苔上

(d)

床前明月光
疑似地上霜
举头望明月
低头思新乡

(e)

?前明月?
疑烧地去霜
两头来明月
低头思孅乡

(f)

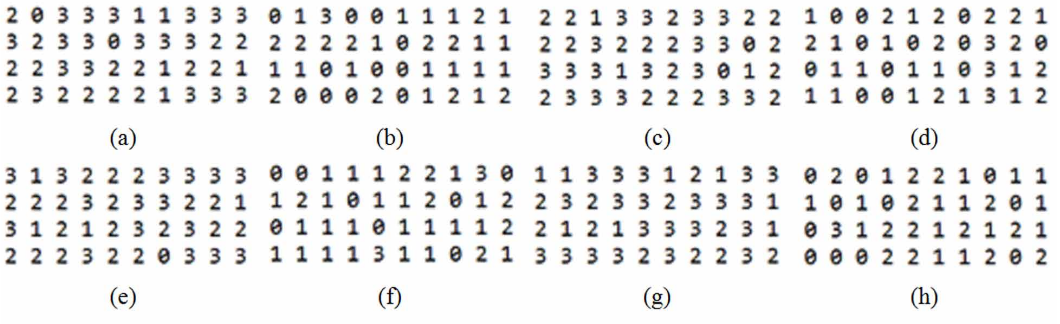
红豆生南国
春来发几枝
愿君多采撷
此物最相思

(g)

绡赃生?国
戢?发几栬
憩君多采撷
?狄最相悉

(h)

**Figure 11. The authentication images in the experiment of self-repair ability and authentication accuracy under Attack 2, where Figure 11a – Figure 11h are the authentication images of Group 1 - Group 8 in Table 4, respectively**



```
2 0 3 3 3 1 1 3 3 3
3 2 3 3 0 3 3 3 2 2
2 2 3 3 2 2 1 2 2 1
2 3 2 2 2 2 1 3 3 3
```

(a)

```
0 1 3 0 0 1 1 1 2 1
2 2 2 2 1 0 2 2 1 1
1 1 0 1 0 0 1 1 1 1
2 0 0 0 2 0 1 2 1 2
```

(b)

```
2 2 1 3 3 2 3 3 2 2
2 2 3 2 2 2 3 3 0 2
3 3 3 1 3 2 3 0 1 2
2 3 3 3 2 2 2 3 3 2
```

(c)

```
1 0 0 2 1 2 0 2 2 1
2 1 0 1 0 2 0 3 2 0
0 1 1 0 1 1 0 3 1 2
1 1 0 0 1 2 1 3 1 2
```

(d)

```
3 1 3 2 2 2 3 3 3 3
2 2 2 3 2 3 3 2 2 1
3 1 2 1 2 3 2 3 2 2
2 2 2 3 2 2 0 3 3 3
```

(e)

```
0 0 1 1 1 2 2 1 3 0
1 2 1 0 1 1 2 0 1 2
0 1 1 1 0 1 1 1 1 2
1 1 1 1 3 1 1 0 2 1
```

(f)

```
1 1 3 3 3 1 2 1 3 3
2 3 2 3 3 2 3 3 3 1
2 1 2 1 3 3 3 2 3 1
3 3 3 3 2 3 2 2 3 2
```

(g)

```
0 2 0 1 2 2 1 0 1 1
1 0 1 0 2 1 1 2 0 1
0 3 1 2 2 1 2 1 2 1
0 0 0 2 2 1 1 2 0 2
```

(h)

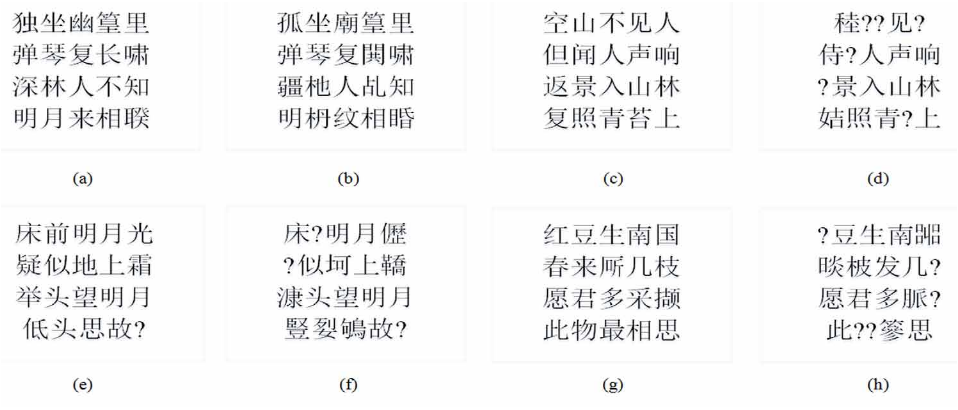
**Table 4. Details in the experiment of self-repair ability and authentication accuracy under Attack 2**

| NO. | Stego test paper | Attack rate | Number of attacked questions | Number of restored bytes authenticated successfully | ER | ASR |
|---|---|---|---|---|---|---|
| 1 | Figure 6a | 0.2 | 38 | 40 | 5% | 100% |
| 2 | Figure 6a | 0.5 | 96 | 40 | 27.5% | 100% |
| 3 | Figure 6b | 0.2 | 38 | 40 | 5% | 100% |
| 4 | Figure 6b | 0.5 | 96 | 40 | 30% | 100% |
| 5 | Figure 6c | 0.2 | 38 | 40 | 2.5% | 100% |
| 6 | Figure 6c | 0.5 | 96 | 40 | 20% | 100% |
| 7 | Figure 6d | 0.2 | 38 | 40 | 0 | 100% |
| 8 | Figure 6d | 0.5 | 96 | 40 | 27.5% | 100% |

As shown in Figure 10, Figure 11 and Table 4, even though the stego test papers are attacked by Attack 2 with 0.2 attack rate, ER is only about 0% - 5%. Among them, the restored information in the 7th group of Table 4 is reconstructed without error. Meanwhile, ER is only about 20% - 30% when the attack rate is increased to 0.5. It is verified that the proposed method can resist a certain degree of Attack 2 with good self-repair ability. Moreover, ASR=100% in all groups of Table 4. It is verified that the proposed method can eliminate the influence of error candidate restored values on reconstructed information to ensure the quality of it with good authentication accuracy.

To examine the self-repair ability and authentication accuracy of the proposed method under Attack 3, the authors attack Figure 6a - Figure 6d by Attack 3 in different scales. Table 5 shows the experimental data. The reconstructed information and authentication image are given as Figure 12 and Figure 13, respectively.

**Figure 12. The restored information in the experiment of self-repair ability and authentication accuracy under Attack 3, where Figure 12a – Figure 12h are the restored information of Group 1 - Group 8 in Table 5, respectively**



独坐幽篁里
弹琴复长啸
深林人不知
明月来相瞬

(a)

孤坐廟篁里
弹琴复閺啸
疆杝人乩知
明�纹相暋

(b)

空山不见人
但闻人声响
返景入山林
复照青苔上

(c)

稑??见?
侍?人声响
?景入山林
姞照青?上

(d)

床前明月光
疑似地上霜
举头望明月
低头思故?

(e)

床?明月偃
?似坷上鞽
漉头望明月
豎裂鴘故?

(f)

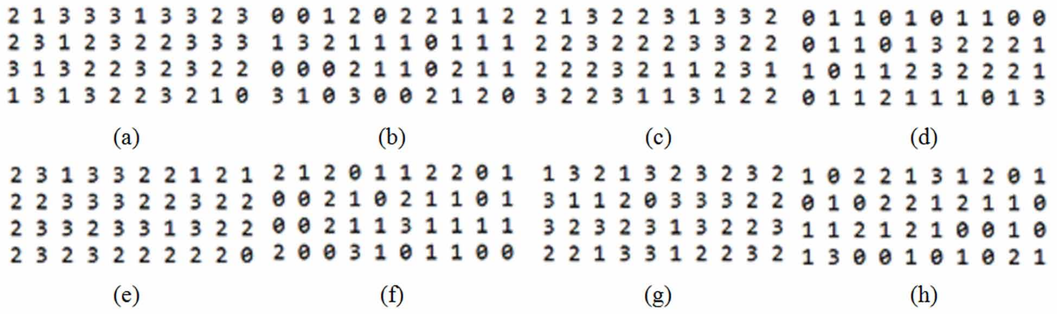
红豆生南国
春来厮几枝
愿君多采撷
此物最相思

(g)

?豆生南鼦
睒柀发几?
愿君多脈?
此??篸思

(h)

**Figure 13. The authentication images in the experiment of self-repair ability and authentication accuracy under Attack 3, where Figure 13a – Figure 13h are the authentication images of Group 1 - Group 8 in Table 5, respectively**



```
2 1 3 3 3 1 3 3 2 3
2 3 1 2 3 2 2 3 3 3
3 1 3 2 2 3 2 3 2 2
1 3 1 3 2 2 3 2 1 0
```

(a)

```
0 0 1 2 0 2 2 1 1 2
1 3 2 1 1 1 0 1 1 1
0 0 0 2 1 1 0 2 1 1
3 1 0 3 0 0 2 1 2 0
```

(b)

```
2 1 3 2 2 3 1 3 3 2
2 2 3 2 2 2 3 3 2 2
2 2 2 3 2 1 1 2 3 1
3 2 2 3 1 1 3 1 2 2
```

(c)

```
0 1 1 0 1 0 1 1 0 0
0 1 1 0 1 3 2 2 2 1
1 0 1 1 2 3 2 2 2 1
0 1 1 2 1 1 1 0 1 3
```

(d)

```
2 3 1 3 3 2 2 1 2 1
2 2 3 3 3 2 2 3 2 2
2 3 3 2 3 3 1 3 2 2
2 3 2 3 2 2 2 2 2 0
```

(e)

```
2 1 2 0 1 1 2 2 0 1
0 0 2 1 0 2 1 1 0 1
0 0 2 1 1 3 1 1 1 1
2 0 0 3 1 0 1 1 0 0
```

(f)

```
1 3 2 1 3 2 3 2 3 2
3 1 1 2 0 3 3 3 2 2
3 2 3 2 3 1 3 2 2 3
2 2 1 3 3 1 2 2 3 2
```

(g)

```
1 0 2 2 1 3 1 2 0 1
0 1 0 2 2 1 2 1 1 0
1 1 2 1 2 1 0 0 1 0
1 3 0 0 1 0 1 0 2 1
```

(h)

**Table 5. Details in the experiment of self-repair ability and authentication accuracy under Attack 3**

| NO. | Stego test paper | Attack rate | Number of attacked questions | Number of restored bytes authenticated successfully | ER | ASR |
|---|---|---|---|---|---|---|
| 1 | Figure 6a | 0.2 | 38 | 40 | 2.5% | 100% |
| 2 | Figure 6a | 0.5 | 96 | 40 | 30% | 100% |
| 3 | Figure 6b | 0.2 | 38 | 40 | 0% | 100% |
| 4 | Figure 6b | 0.5 | 96 | 40 | 25% | 100% |
| 5 | Figure 6c | 0.2 | 38 | 40 | 2.5% | 100% |
| 6 | Figure 6c | 0.5 | 96 | 40 | 32.5% | 100% |
| 7 | Figure 6d | 0.2 | 38 | 40 | 2.5% | 100% |
| 8 | Figure 6d | 0.5 | 96 | 40 | 32.5% | 97.5% |

As shown in Figure 12, Figure 13 and Table 5, even though the stego test papers are attacked by Attack 3 with 0.2 attack rate, is only about 0% - 2.5%. Among them, the restored information in the 3rd group of Table 5 is reconstructed without error. Meanwhile, is only about 30% when the attack rate is increased to 0.5. It is verified that the proposed method can resist a certain degree of Attack 3 with good self-repair ability. Moreover, in all groups of Table 5 is 100% except the 8th Group. It is verified that the proposed method can eliminate the influence of error candidate restored values on reconstructed information to ensure the quality of it with good authentication accuracy.

## The Comparative Experiment

To compare the hidden capacity of the proposed method with traditional SCIH, Table 6 shows the public experimental data of SCIH and the proposed method. In Table 6, because Zhou et al. (2019) employed marked image blocks to reconstruct secret image and the size of blocks is unknown, the authors do not list the data of the work of Zhou et al. (2019) in Table 6. Meanwhile, because the hidden capacities of a single text in Zhang et al. (2017a, 2017b) are 1 English word and the sizes of carrier images in SCIHI are unknown, the authors can't calculate their HR . Therefore, the authors represent the corresponding data as "-" to mark it as non-existence. To facilitate the calculation of HR , the authors regard one Chinese character as a 2-byte ASCII in these methods.

Table 6. Hidden capacity of the proposed method and traditional SCIH

| Method | Size of a single carrier(KB) | Hidden capacity of a single carrier | HR |
|---|---|---|---|
| Chen et al (2015) | 1 | 2B | 2.00 |
| Zhou et al (2016b) | 1 | 3.14 B | 3.14 |
| Chen et al (2017) | 2 | 4.14 B | 2.07 |
| Chen et al (2018) | 2 | 4.82 B | 2.41 |
| Xia et al. (2017) | - | 1.5 B | - |
| Zhang et al. (2017a, 2017b) | - | 1 English word | - |
| Zhou et al. (2015) and Yuan et al. (2017) | - | 1 B | - |
| Zhou et al. (2017) | - | 2.5 B | - |
| Zhou et al (2016a) | - | 3.72 B | - |
| Cao et al. (2018) | - | 4.5 B | - |
| Zhang et al. (2018) | - | $M/8$ B | - |
| Zheng et al. (2017) | - | 2.25 B | - |
| Wu et al. (2018) | - | 4 B | - |
| Zou et al. (2019) | - | 10 B | - |
| The proposed method | 6.45 | 40 B | 6.20 |

As shown in Table 6, the hidden capacity of the proposed method is higher than SCIH. Among SCIHT, the highest hidden capacity is 4.82 bytes (Chen et al, 2018) which is only the hidden capacity of 5 HMs in the proposed method.

To compare the hidden capacity of a single question in the proposed method, and Lu et al. (2018), Table 7 shows the public experimental data. Moreover, Lu et al. (2018) employs one multiple-choice and one blank-filling to express 5-bit secret together, the authors regard the hidden capacity of a single question as 2.5 bits.

Table 7. Hidden capacity of the proposed method and Lu et al. (2018)

| Method | Size of stego test paper(KB) | Hidden capacity of a single question (B) | HR |
|---|---|---|---|
| Lu et al (2018) | 2.88 | 2.5 | 13.89 |
| The proposed method | 6.45 | 8 | 6.20 |

As shown in Table 7, the hidden capacity of a single question in the proposed method is higher than Lu et al. (2018), despite $HR$ is lower. The reason is that the authors back up secret information for self-repair ability and introduce NMs in stego test paper. It results in the lengthier stego test paper and lower $HR$ . However, the proposed method can calculate reliability of reconstructed information with good self-repair ability and effective authentication ability. Therefore, the proposed method has better practicability.

For the cost of database creation, search, and maintenance, there are only 10000 question stems in the test database. Moreover, the proposed method divide these stems into eight sets and map stems into 16-bit hash values by their set indexes. Moreover, the corresponding executive time is only about 331 ms. In other words, the question stems are only divided into $8\times16=128$ categories to express secret information. Compared with SCIH, the proposed method has a smaller database and a simpler database structure with lower executive cost.

## CONCLUSION

To improve the hidden capacity of a single question, further avoid the absence of authentication and provide self-repair ability, this paper proposes a high capacity test paper disguise method combined with interpolation backup and double authentications. Firstly, a secret byte sequence is backed up as a backup byte sequence by Lagrange (2,4) interpolation polynomials over $GF(2^8)$, and then the backup byte sequence is further transformed into a backup index sequence. Secondly, a test question database divided into eight sets is created and then the question stem MD5 values are mapped to stem hash values by stem set indexes. Finally, the backup index sequence is disguised as a stego test paper by candidate answer orders and stem hash values together. In restoration, codebook extension authentication and sharing coefficient authentication are applied to authenticate candidate restored value, and the most reliable candidate restored value is selected by the frequency of candidate restored values to reconstruct secret information. The experimental results and analysis show that the proposed method can eliminate the influence of error candidate restored values on restored information using double authentications and further calculate its reliability by reliable calculation. Due to the secret information backup strategy, it has an excellent self-repair ability. Moreover, it improves the hidden capacity of a single question by using stem hash and candidate answer orders through the simpler test database.

## REFERENCES

Cao, Y., Zhou, Z., Sun, X., & Gao, C. (2018). Coverless information hiding based on the molecular structure images of material. *Computers Material and Continua*, *54*(2), 197–207.

Chen, X., & Chen, S. (2018). Text coverless information hiding based on compound and selection of words. *Soft Computing*, *23*(15), 6323–6330. doi:10.1007/s00500-018-3286-7

Chen, X., Chen, S., & Wu, Y. (2017). Coverless information hiding based on the Chinese character encoding. *Journal of Internet Technology*, *18*(2), 313–320.

Chen, X., Sun, H., Tobe, Y., Zhou, Z., & Sun, X. (2015). Coverless information hiding method based on the Chinese mathematical expression. In *The 1st International Conference on Cloud Computing and Security* (pp. 133-143). Springer International Publishing. doi:10.1007/978-3-319-27051-7_12

Shao, L., & Le, Z., (2018). Multiple thresholds progressive secret image sharing scheme based on DCT. *Netinfo Security*, *18*(3), 390–403.

Lu, H., & Shao, L. (2018). Coverless test paper disguise combined with non-direct transmission and random codebook. *Journal of Applied Sciences*, *36*(2), 331–346.

Ou-yang, X., Shao, L., & Le, Z. (2017). Gloise field self-recovery image sharing scheme with non-equivalent backup and double authentications. *Journal of Software*, *28*(12), 3306–3346.

Shao, L., & Le, Z. (2019). (*t,s,k,n*) image sharing scheme with multi-version backups and restricted double authentications. *Acta Electronic Sinca*, *47*(2), 390–403.

Wu, J., Jia, Y., & Liu, Y. (2018). Coverless information hiding algorithm based on image coding and stitching. *Journal of South China University of Technology*, *46*(5), 32–38.

Xia, Z., & Li, X. (2017). Coverless information hiding method based on LSB of the character's unicode. *Journal of Internet Technology*, *18*(6), 1353–1360.

Yuan, C., Xia, Z., & Sun, X. (2017). Coverless image steganography based on SIFT and BOF. *Journal of Internet Technology*, *18*(2), 435–442.

Zhang, J., Shen, J., Wang, L., & Lin, H. (2017). Coverless text information hiding method based on the word rank map. *Journal of Internet Technology*, *18*(2), 427–434.

Zhang, J., Xie, Y., Wang, L., & Lin, H. (2017). Coverless text information hiding method using the frequent words distance. In *The 3rd International Conference on Cloud Computing and Security*(pp.133-143). Nanjing, China: Springer International Publishing. doi:10.1007/978-3-319-68505-2_11

Zhang, X., Peng, F., & Long, M. (2018). Robust coverless image steganography based on DCT and LDA topic classification. *IEEE Transactions on Multimedia*, *20*(12), 3223–3238. doi:10.1109/TMM.2018.2838334

Zheng, S., Wang, L., Ling, B., & Hu, D. (2017). Coverless information hiding based on robust image hashing. In *The 13th International Conference on Intelligent Computing* (pp. 536-547). Liverpool, UK: Springer International Publishing. doi:10.1007/978-3-319-63315-2_47

Zhou, Z., Cao, Y., & Sun, X. (2016). Coverless information hiding based on bag-of-words model of image. *Journal of Applied Sciences*, *34*(5), 527–536.

Zhou, Z., Mu, Y., & Wu, Q. M. J. (2019). Coverless image steganography using partial-duplicate image retrieval. *Soft Computing*, *23*(13), 4927–4938. doi:10.1007/s00500-018-3151-8

Zhou, Z., Mu, Y., Zhao, N., Wu, Q. M. J., & Yang, C. (2016). Coverless information hiding method based on multi-keywords. *International Journal of Security and Its Applications*, *10*(9), 309–320. doi:10.14257/ijsia.2016.10.9.30

Zhou, Z., Sun, H., Harit, R., Chen, X., & Sun, X. (2015). Coverless image steganography without embedding. In *1st International Conference on Cloud Computing and Security*(pp.123-132). Nanjing, China: Springer International Publishing. doi:10.1007/978-3-319-27051-7_11

Zhou, Z., Wu, Q., Yang, C., Sun, X., & Pan, Z. (2017). Coverless image steganography using histograms of oriented grandients-based hashing algorithm. *Journal of Internet Technology*, *18*(5), 1177–1184.

Zou, L., Sun, J., Gao, M., Wan, W., & Gupta, B. B. (2019). A novel coverless information hiding method based on the average pixel value of the sub-images. *Multimedia Tools and Applications*, *78*(7), 7965–7980. doi:10.1007/s11042-018-6444-0

*Hai Lu received a master's degree from Shaanxi Normal University. His research interests include multimedia security and coverless information hiding.*

*Liping Shao received the Ph.D. degree from Xi'an Jiaotong University in 2010. Now, he is an associate professor of School of Computer Science, Shaanxi Normal University, Xi'an, Shaanxi, China. His current research interests include information hiding, watermarking, steganography, disguising, secret sharing, scrambling and encryption.*

*Qinglong Wang is a master student of Shaanxi Normal University.*