Computationally Secure Secret Image Sharing

Amir M. Ahmadian, Maryam Amirmazlaghani
Department of Computer Engineering and Information Technology,
Amirkabir University of Technology, Tehran, Iran
Email: {amir.ahmadian, mazlaghani}@aut.ac.ir

***Abstract*—A secret image sharing scheme is used to protect the confidentiality of important images and also to safeguard them against single point of failure. the first proposed secret image sharing scheme used a technique to reduce the size of shares to $\frac{1}{t}$ of the secret, the benefits of small share size drawn the attention of many researchers to that technique and it was used in many of the later studies. In this paper we first show that this technique has some serious security weaknesses, then in order to overcome these drawbacks and achieve computational security, we use All-or-Nothing transformation before share generation process. Also, we use an Information Dispersal Algorithm instead of a secret sharing algorithm which results in better performance in share generation and secret reconstruction. In this way, our proposed secret image sharing scheme enjoys better performance, computational security and small share size. Experimental results confirm the efficiency of the proposed scheme.**

I. Introduction

A (t,n) threshold secret image sharing is a method of distributing some secret image among n participants in such a way that only if t or more *different* participants pool their shares together they can reconstruct the original secret image, while providing any less than t shares reveals no *useful* information about it.

Alongside *Image Encryption* and *Steganography*, *Secret Image Sharing* is another way of providing confidentiality for secret images. When using image encryption if something happens to the encrypted image or its encryption key (for example it becomes corrupted or destroyed during storage or transmission) the whole secret is lost. The same problem is applicable to steganography, if the stego-image or its embedding key becomes corrupted or gets destroyed *in most cases* the embedded secret is lost, but in secret image sharing the secret is distributed among some participants, so as long as t participants (shares) remain available it is possible to reconstruct the original secret image.

The fact that having less than t shares should provide no *useful* information about the secret is called *Perfectness*, in a threshold secret sharing scheme perfectness should be provided by one of the following security guarantees:

- Information Theoretic Security: which means that even if the attacker has unlimited computational power, it is still impossible for him to learn any information about the secret.
- Computational Security : the security is guaranteed based on some assumptions about attacker's computational capabilities and some computational hardness assumptions (e.g. Discrete Logarithm, Factorization, etc.). Almost all of the well-known encryption schemes (i.g. AES, RSA, etc.) fall into this category.

Several threshold secret image sharing schemes have been proposed in the last decade. Although some of them [1] use Blakley's secret sharing method [2], Cellular Automata [3], [4] or Chinese Reminder Theorem (CRT) [5] as the basis for their schemes, most of them [6–14] used Shamir's secret sharing scheme [15] because of its popularity and information theoretic security.

The Blakley's and CRT based approaches have not been very popular because of their computational complexity and the fact that in those methods the size of each share is as large as the secret image and these problems makes them impractical in case of very large secret images. On the other hand the Cellular Automata based schemes have linear computational complexity that is ideal for secret image sharing, but they also suffer from a major drawback which makes them unuseful for real world applications, only t *consecutive* shares are able to reconstruct the secret.

Therefore, most of the proposed secret image sharing schemes used Shamir's scheme as the base of their methods, but as we will see later in this paper, most of them suffer from a security issue that not only shatters their information theoretic security, but also makes them computationally insecure as well. In this paper, after demonstrating the security weaknesses of previous schemes, we're going to propose our computationally secure (t,n)-threshold secret image sharing scheme, that uses an All-or-Nothing [16], [17] based transform to achieve computational security and a computationally efficient Information Dispersal Algorithm [17–20] to generate the shares of each participant.

The remaining of this paper is organized as follows, in Section 2 we provide an overview of existing secret image sharing schemes and their strengths and weaknesses, our proposed computationally secure scheme is presented in Section 3, the experimental results are given in Section 4, and finally the conclusions are drawn in Section 5.

II. Related Works

In this section, we are going to provide an overview of the secret image sharing schemes based on Shamir's approach. So, first we are going to briefly introduce Shamir's secret sharing scheme, and then explain the secret image sharing schemes which were based on it.

A. Shamir's Scheme for (t,n) Threshold Secret Sharing

In 1979, Shamir [15] proposed a polynomial based (t,n) threshold secret sharing scheme. This scheme divides the secret data S into n shares $S_1, S_2, ..., S_n$ in such a way that if t $(t \leq n)$ shares are present, secret S can be recovered using Lagrange polynomial interpolation. But if less than t shares are present, *no information* about S can be learned. In order to share the secret S among n participants, a random polynomial $f(x)$ of degree $t-1$ is generated:

$$f(x) = a_0 + a_1x + a_2x^2 + ... + a_{t-1}x^{t-1} \quad mod\ p \qquad (1)$$

In equation (1) p is a prime number bigger than both S and n, $a_0 = S$ and a_i $(i = 1, 2, ..., t-1)$ are random numbers chosen uniformly from $[0, p-1]$. Each participant i is assigned a unique identifier $\mathrm{x_i}$, and the share of ith participant is calculated as $S_i = f(x_i)\ mod\ p$.

The share of each participant is then given to him via a secure channel. As it is obvious the size of each share S_i is equal to the size of the secret itself, hence Shamir's scheme is an **ideal** secret sharing scheme [15], [21].

In order to reconstruct the secret, having t or more shares (i.e. $f(x_i)$ and their corresponding identifiers x_i), the $t-1$ degree polynomial can be reconstructed using Lagrange polynomial interpolation and after reconstructing the polynomial, the secret value S can be obtained as $S = f(0)\ mod\ p$

Shamir's secret sharing scheme is information theoretic secure, this means that $t-1$ or fewer participants can't learn *any information* about the secret [21].

B. Secret Image Sharing

Using Shamir's secret sharing scheme, in 2002, Thien and Lin [13] proposed the first secret image sharing scheme (abbreviated as *SISS*). In their scheme, a grayscale image is first permuted randomly and then it is divided into several non-overlapping blocks each containing t pixels, then this t pixels are used as the t coefficients $a_0, a_1, ..., a_{t-1}$ of the polynomial (1).

The biggest achievement of their scheme was the smaller size of shares ($\frac{1}{t}$ of the secret image), this was because they didn't used any random coefficients and all of the coefficients in polynomial (1) were used for embedding secret pixels. Because usually a secret image is very large, a method that reduces the size of shadows is very appealing because when the size of shares (which we call *Shadows* hereafter) are smaller, it is easier to transmit them over a network, store them, embed them in a cover media using steganography, etc. For this reason **most** of the secret image sharing schemes that were proposed later [6–12], [14] continued to use this approach in their schemes.

But using all the coefficients in polynomial (1) for sharing secret pixels, weakens the security of the scheme, it is no longer information theoretic secure [21] and also in some cases the shadows will leak key elements of secret image. In the next section we are going to describe this problem in details.

C. Information Leakage Problem in SISS

Most of the time in an image the neighboring pixels are highly correlated, so if we use the SISS method to share a secret images, some of the regions in the resulted shadows are similar to the secret image, hence revealing key elements of the secret image.

Fig. 1: Lena

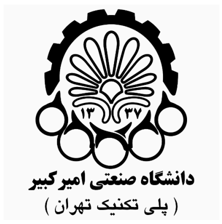


Fig. 2: AUT Logo

To demonstration this security issue, we use a (2,4)-threshold SISS scheme to share the natural image (1), the resulted shadows are presented in Figure (3). As it is obvious, key elements of the secret image such as its edges are completely visible. But when the secret is a Textural image, such as a logo, the result is much worse and almost all the elements of secret image is visible in the shadows[21]. For example the shadows of a (2,4)-threshold SISS scheme for textural image (2) is depicted in Figure (4).

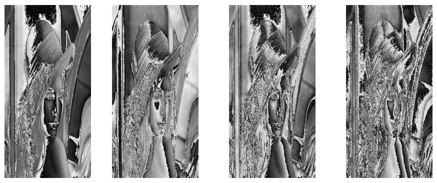


Fig. 3: Shadows Generated from Image (1)

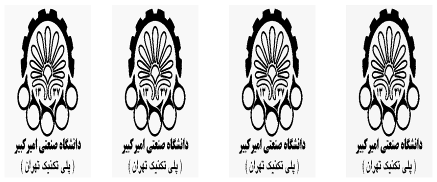


Fig. 4: Shadows Generated from Image (2)

So the SISS on itself cannot provide confidentiality for the secret image, and this issue was known from the beginning, to solve this problem it was suggested [13] to apply a permutation to the secret image before the share generation process. For example if we apply *Arnold Transformation* to the secret image (2) for 25 iterations and then share it via SISS method, the resulted shadows (illustrated in Figure (5)) won't leak any information about the original secret.

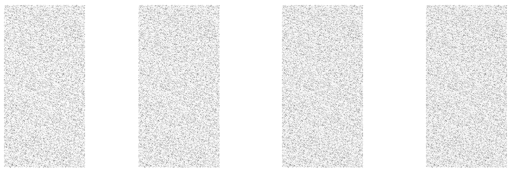


Fig. 5: Shadows Generated from Figure (2) after Arnold Transformation

But unfortunately this solution also cannot guarantee security completely. For example if participant 1 is dishonest and **knows** the permutation key, in order to learn some information about the secret [14], he can:

1) Copy his shadow image.
2) Disguise this copied shadow as shadow 2.
3) Gives his shadow and **fake** shadow 2 to dealer to reconstruct the secret.
4) Apply reverse of permutation to reconstructed secret.

We apply this process to the shadows of figure (5), the result is depicted in figure (6) and it is clearly visible that the resulted image contains some elements of original secret image.

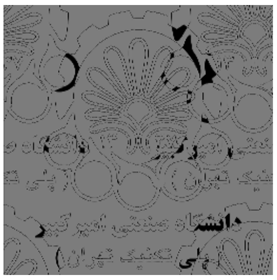


Fig. 6: Reversed Permutation

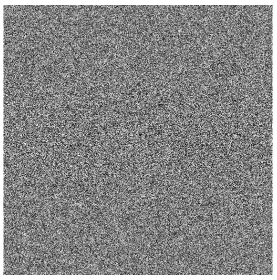


Fig. 7: AONT Result

Also it was proven that a *Permutation-Only* image encryption scheme is not secure against *chosen plaintext* attacks [22]. So a secret image sharing scheme that relies on permutation-only image encryption for confidentiality -in the sense of indistinguishability- is not secure.

One way of dealing with this security issue is to use Shamir's scheme without any change [23], this means that every single pixel of the secret image will be embedded in the first coefficient of the polynomial (1) and the rest of the coefficients are random numbers. This approach guarantees information theoretic security but the size of its shadows is equal to the size of secret image and in some cases the large size of shadows is not acceptable, unfortunately it was proven [24] that the size of shadows in an information theoretic secure secret sharing scheme can't be smaller than the size of the secret itself. On the other hand it is known that a computationally secure secret sharing scheme can reduce the size of shadows to $\frac{1}{t}$ while guaranteeing the scheme's security in face of attackers with limited computational capability [25].

Besides, information theoretic security in practical scenarios isn't very useful, because as we mentioned before, the shadows should be transmitted to participants via a secure channel and most of the time the security of this channel is provided with a computationally secure encryption algorithm such as AES [25].

In this paper, our goal is to keep the advantage of previously mentioned SISS, which was its small shadow size while overcoming its security issues. Due to the above explanations, it is obvious that in SISS approaches, the confidentiality was not provided by Shamir's scheme but with a permutation before it. So, the role of Shamir's scheme was reduced to distributing the output of permutation layer among the participants. In this paper, we use the same two layered approach, but by utilizing the methods described in [17], instead of permutation we apply a modified AONT transformation to secret image in order to guarantee its security. Then, for share generation instead of Shamir's scheme, we use an Information Dispersal Algorithm.

III. The Proposed Scheme

The biggest security drawback of the SISS method is that the permutation applied to the secret image before share generation can not provide computational security. So, to overcome this drawback, we replace this permutation with a modified All-or-Nothing transformation.

Our *Computationally Secure Secret Image Sharing Scheme* consists of these two steps:

1) A security transform, which is a slightly modified version of Rivest's All-or-Nothing Transform [16].
2) An Information Dispersal Algorithm which is based on systematic Reed-Solomon coding and is used to generate n shadows from the output of security layer [19], [20].

In the next sections, we describe each of these steps individually and then combine them in such a way that the result will become a computationally secure secret image sharing scheme.

A. Security Layer : SI-AONT

In this step of the algorithm, as a preprocess we apply a modified version of Rivest's *All-or-Nothing Transform* (AONT) [16] to the secret image. The AONT can be viewed as a (n+1,n+1)-threshold scheme, the data is encoded into n+1 blocks in such a way that none of the original blocks can be decoded unless all of the n+1 encoded blocks are present.

In the original AONT [16] the data is first partitioned into n blocks $d_0, d_1, ..., d_{n-1}$, and a random key k is generated, then each block is encoded using the following formula:

$$c_i = d_i \oplus E_k(i+1) \quad for\ i = 0, 1, ..., n-1$$

where E is a symmetric encryption algorithm that it's output length is at least as long as the size of each block. Finally the $n+1$th block is calculated as follows:

$$c_n = k \oplus h_0 \oplus h_1 \oplus ... \oplus h_{n-1}$$

where

$$h_i = E_{k_0}(c_i \oplus i) \quad for\ i = 0, 1, ..., n-1$$

where k_0 is a fixed, publicly-known encryption key. [16], [17]

The AONT has computational security which means that in order to decode the blocks, an attacker should either possesses all of the $n+1$ blocks or be able to guess k, and if the length of k is large enough the process of guessing it becomes computationally *infeasible*.

Originally AONT was proposed as an encryption mode, so the result of it would have been encrypted afterwards, but in secret sharing this is not the case.

In order to be able to guarantee schemes security we should restrict the size and the number of blocks, so we *slightly* modify the original AONT in order to make it more applicable to secret image sharing:

1) Generate a **Random** key k.
2) Partition the secret image into exactly t equally sized blocks $B_1, B_2, ..., B_t$. (add padding if needed)
3) Using k, generate t (different) random masks $R_1, R_2, ..., R_t$, the size of R_is should be equal to the size of blocks. (this masks can be generated using any pseudo-random bit generator or even an encryption scheme)
4) Transform each of blocks using equation (2):
$$D_i = B_i \oplus R_i \quad for\ i = 1, 2, ..., t \tag{2}$$
5) Compute h as follows:
$$h = Hash(D_1 \oplus D_2 \oplus ... \oplus D_t) \tag{3}$$
The hash function of equation (3) should be a secure cryptographic hash function and its output length must be equal to the length of k. For example if k is a 256 bit key, we should use a hash function with output of 256 bits (e.g. SHA-256).
6) Compute $P = h \oplus k$, we can publish P publicly or share it among the participants using an IDA.

This modified version (which we call SI-AONT henceforth) restricts the number of blocks to t, this is because we have to guarantee that even if $t - 1$ participants pool their shares together, still one block will remain missing, and since guessing this last block should be infeasible, the size of blocks must be large enough (at least 256 bits). The result of SI-AONT is a noise like image that reveals no information about the secret image. For example the transformed version of figure (2) is depicted in figure (7).

This transformation on itself provides no security because by knowing all the transformed blocks one can easily reconstruct the original secret image. To achieve security, each of image blocks should be dispersed among n participants in such a way that if $t - 1$ participants try to reconstruct the data, a complete block remains missing. Next we first introduce the Information Dispersal Algorithms and then discuss how to disperse transformed image to meet the mentioned requirement.

B. Information Dispersal Algorithm

For the dispersal algorithm, we use the IDA proposed in [17], [19] which is based on the well known Reed-Solomon codes. Every information dispersal algorithm can be illustrated as a matrix-vector product [17]. A $n \times t$ generator matrix G is multiplied by a data vector D of length t and the result will be stored in the vector R of length n.

$$\begin{bmatrix} g_{1,1} & g_{1,2} & \cdots & g_{1,t} \\ g_{2,1} & g_{2,2} & \cdots & g_{2,t} \\ g_{3,1} & g_{3,2} & \cdots & g_{3,t} \\ g_{4,1} & g_{4,2} & \cdots & g_{4,t} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n,1} & g_{n,2} & \cdots & g_{n,t} \end{bmatrix} \times \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_t \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ \vdots \\ r_n \end{bmatrix}$$

The size of every element in matrix G and vectors D and R is 8 bits, we consider this size because this way each pixel can be stored in one element and also all arithmetic operations can be in $GF(2^8)$, which is easy to perform and implement [19].

The *generator matrix* G should be constructed in such a way that all combinations of t rows of it result in an invertible matrix. In this way the rows of the corresponding matrix for every t element of vector R, result in a $t \times t$ invertible matrix. We call the inverted matrix G', and t available elements of R multiplied by G' will result in vector D, so every t elements of R can reconstruct the data vector D. An example of such matrix is the *Vandermonde matrix*. A $m \times n$ Vandermonde matrix is generated using the following formula:
$$G_{i,j} = i^{j-1}$$

For $i = 1, 2, ..., m$ and $j = 1, ...n$. An example of a 5×3 Vandermonde matrix in $GF(2^8)$ is:

$$\begin{bmatrix} 1^0 & 1^1 & 1^2 \\ 2^0 & 2^1 & 2^2 \\ 3^0 & 3^1 & 3^2 \\ 4^0 & 4^1 & 4^2 \\ 5^0 & 5^1 & 5^2 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 5 \\ 1 & 4 & 16 \\ 1 & 5 & 17 \end{bmatrix}$$

Using Vandermonde matrix is an example of non-systematic coding [18], [19], on the other hand in a *Systematic* code, the first t elements of vector R are exactly the t elements of data vector D, and the remaining $n - t$ elements of R are calculated as a combination of the elements of D. So the first t rows of the generator matrix in this type of coding is a $t \times t$ identity matrix [17], [19], [20].

$$\left[\begin{array}{cccc} 1 & 0 & \ldots & 0 \\ 0 & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & 1 \\ \hline g_{1,1} & g_{1,2} & \cdots & g_{1,t} \\ \vdots & \vdots & \ddots & \vdots \\ g_{n-t,1} & g_{n-t,2} & \cdots & g_{n-t,t} \end{array}\right] \times \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_t \end{bmatrix} = \left[\begin{array}{c} d_1 \\ d_2 \\ \vdots \\ d_t \\ \hline r_1 \\ \vdots \\ r_{n-t} \end{array}\right]$$

the process of creating such a matrix from Vandermonde matrix is described in [19], [20].

Using a systematic coding for information dispersal improves the overall performance of the algorithm, because any of the first t elements of R can reconstruct their corresponding element in D without any encoding. On the other hand this type of coding does not obfuscate the shared data and hence is not secure, but its not a problem for our scheme because as we mentioned earlier the security of our scheme is provided by the SI-AONT transformation beforehand.

C. Proposed Scheme : Combining SI-AONT and IDA

In the proposed computationally secure secret image sharing scheme (CS-SISS) we use the information dispersal algorithm of the last section to generate the shares for each of the participants. For achieving security the shares should be generated in such a way that only *t or more* participants can reconstruct all of the t blocks of the data. The Share generation algorithm is as follows:

1) Apply the process of SI-AONT to the secret image, the output of this process is a public value P and t transformed blocks $B_1, B_2, ..., B_t$

2) Take the first (unprocessed) pixels of each of these t blocks, $b_1, b_2, ..., b_t$ and create vector D, multiply D by a systematic generator matrix G and store the results in vector R.
3) The ith element of the vector R is a pixel for the ith shadow.
4) Repeat steps 2 and 3 until all pixels of the blocks are processed.
5) Send the ith shadow to participant i (via a secure channel).
6) Publish the value P in a public directory. (or share it among participants using steps 2 and 3)

This way of dispersing the blocks of transformed secret image guarantees that even if $t-1$ participants pull their shadows together, still one block would be missing, so *XOR*ing them wouldn't result in h, and without having h the public value P won't leak any information about the random key and k will remain unknown, because of that $t-1$ participants won't be able to regenerate the R_is and hence their blocks will remain encoded and won't leak any information about the secret image.
There are some security considerations about the proposed scheme:

- The size of the key should be large enough to make the exhaustive key search attacks infeasible. It is suggested to use at least a 256 bit key.
- During secret reconstruction if only one of the blocks were missing, those $t-1$ participants can try to guess it, so in order to make this effort infeasible, the size of each block should be at least 256 bit (or 32 pixels).
- Considering the above requirement, the minimum size of secret image is $32 \times t$ pixels, if the secret image is smaller than that, the size of the blocks will be smaller than 32 and guessing a missing block becomes easier for the attacker.
- Because the output of SI-AONT is randomized, the probability of each of the 2^{256} possible outputs being the correct one is the same, so theoretically, the attacker should try 2^{256} different values in order to guess the correct one.
- If the secret is a *Black and White* image, because the possible values of a pixel is only 0 and 1, guessing a block for attacker is much easier, so in order to guarantee security, the recommended minimum block size and minimum image size for a B&W image are 256 and $256 \times t$ pixels respectively.

IV. Experimental Results

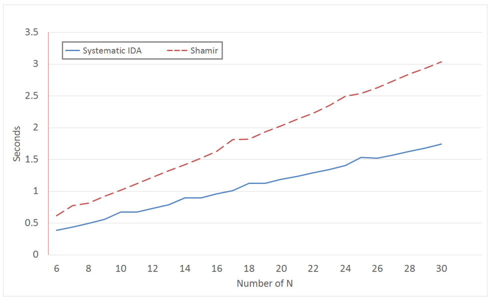
The experimental results of the proposed (2,4)-CS-SISS scheme for secret images of Figures (1) and (2) can be found in Figures (10) and (11) respectively. In these experiments the secret image's size is 512×512 and the size of each shadow image is 256×512. So as expected the size of each shadow is reduced to $\frac{1}{t}$ of the original secret.

If instead of publishing the public value P, we decide to share it among the participants using an IDA, $\frac{SIZE(P)}{t}$ bits will be added to the size of shadows. For example if the size of key is 256 bit and we use SHA-256 as hash function, the size of P will be 256 bit, so in a (2,4)-CS-SISS, 128 bit of data will be added to the shares, meaning that the final size of each shadow will be $((256 \times 512) + 16)$ pixels.

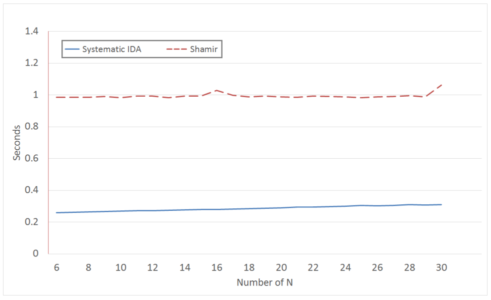
Our Proposed SI-AONT can also be used instead of permutation in the traditional SISS schemes (as in [13]) to achieve computational security, but as we mentioned earlier the performance of information dispersal using a Systematic IDA is much better than Shamir's scheme. In order to show this performance superiority we compared *Reed-Solomon based Systematic IDA* [19], [20] with *SISS* [13] in terms of the speed of share generation and secret reconstruction using the same secret image (1) according to the following two experiments:

The first experiment compares the schemes using a fixed value of n equal to 30 and a threshold t varying from 4 to 30, the result of this experiment for share generation and secret reconstruction is illustrated in Figures (9a) and (9b) respectively.

In the second experiment, the value of t is fixed at 6 and the value of n is increased from 6 to 30, the result of this comparison for share generation and secret reconstruction is depicted in Figures (8a) and (8b) respectively.



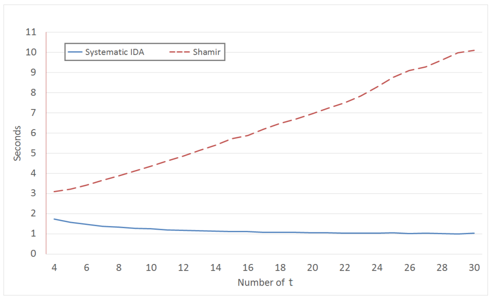
(a) Share Generation



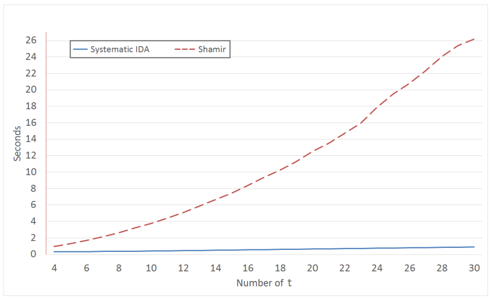
(b) Secret Reconstruction

Fig. 8: Performance Comparison for $t = 4$ and $n = 6, 7, ..., 30$

As it can be seen in Figures (8) and (9), while the security is provided via SI-AONT, for the propose of information dispersal, the Reed-Solomon based IDA has much better performance, hence while it is still possible to use Shamir's Scheme as IDA, for practical purposes (specially when dealing with large images) it is recommended to use a more optimized IDA, such as the one introduced in the last section.



(a) Share Generation



(b) Secret Reconstruction

Fig. 9: Performance Comparison for $n = 30$ and $t = 4, 5, ..., 30$

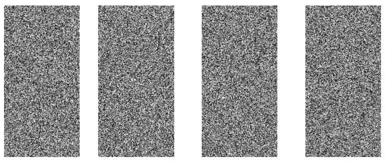


Fig. 10: Shadows Generated from Image (1)

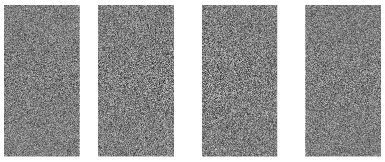


Fig. 11: Shadows Generated from Image (2)

V. Conclusion

In this paper, we first demonstrated that the extensively used secret image sharing scheme of [13] is not secure and described some examples of its security defects. Then we proposed a computationally secure secret image sharing (CS-SISS) scheme using SI-AONT and IDA which has both the benefits of computational security and small shadow size. Our experimental results showed that the generated shadows reveal no information about the secret and also the performance of proposed scheme is much better than the methods based on Shamir's secret sharing scheme.

References

[1] H.-K. Tso, "Sharing secret images using blakleys concept," *Optical Engineering*, vol. 47, no. 7, pp. 077 001–077 001, 2008.

[2] G. R. Blakley, "Safeguarding cryptographic keys," *Proc. of the National Computer Conference1979*, vol. 48, pp. 313–317, 1979.

[3] Z. Eslami, S. Razzaghi, and J. Z. Ahmadabadi, "Secret image sharing based on cellular automata and steganography," *Pattern Recognition*, vol. 43, no. 1, pp. 397–404, 2010.

[4] G. Alvarez, L. H. Encinas, and A. M. del Rey, "A multisecret sharing scheme for color images based on cellular automata," *Information Sciences*, vol. 178, no. 22, pp. 4382–4395, 2008.

[5] S. J. Shyu and Y.-R. Chen, "Threshold secret image sharing by chinese remainder theorem," in *Asia-Pacific Services Computing Conference, 2008. APSCC'08. IEEE*. IEEE, 2008, pp. 1332–1337.

[6] C.-C. Wu, S.-J. Kao, and M.-S. Hwang, "A high quality image sharing with steganography and adaptive authentication scheme," *Journal of Systems and Software*, vol. 84, no. 12, pp. 2196–2207, 2011.

[7] Z. Eslami and J. Z. Ahmadabadi, "Secret image sharing with authentication-chaining and dynamic embedding," *Journal of Systems and Software*, vol. 84, no. 5, pp. 803–809, 2011.

[8] C.-C. Chang, Y.-P. Hsieh, and C.-H. Lin, "Sharing secrets in stego images with authentication," *Pattern Recognition*, vol. 41, no. 10, pp. 3130–3137, 2008.

[9] C.-C. Thien and J.-C. Lin, "An image-sharing method with user-friendly shadow images," *IEEE Transactions on circuits and systems for video technology*, vol. 13, no. 12, pp. 1161–1169, 2003.

[10] R.-Z. Wang, Y.-F. Chien, and Y.-Y. Lin, "Scalable user-friendly image sharing," *Journal of Visual Communication and Image Representation*, vol. 21, no. 7, pp. 751–761, 2010.

[11] R.-Z. Wang and C.-H. Su, "Secret image sharing with smaller shadow images," *Pattern Recognition Letters*, vol. 27, no. 6, pp. 551–555, 2006.

[12] Y.-S. Wu, C.-C. Thien, and J.-C. Lin, "Sharing and hiding secret images with size constraint," *Pattern Recognition*, vol. 37, no. 7, pp. 1377–1385, 2004.

[13] C.-C. Thien and J.-C. Lin, "Secret image sharing," *Computers & Graphics*, vol. 26, no. 5, pp. 765–770, 2002.

[14] J. He, W. Lan, and S. Tang, "A secure image sharing scheme with high quality stego-images based on steganography," *Multimedia Tools and Applications*, pp. 1–22, 2016.

[15] A. Shamir, "How to share a secret," *Communications of the ACM*, vol. 22, no. 11, pp. 612–613, 1979.

[16] R. L. Rivest, "All-or-nothing encryption and the package transform," in *International Workshop on Fast Software Encryption*. Springer, 1997, pp. 210–218.

[17] J. K. Resch and J. S. Plank, "Aont-rs: Blending security and performance in dispersed storage systems," in *Proceedings of the 9th USENIX Conference on File and Stroage Technologies*, ser. FAST'11. USENIX Association, 2011, pp. 14–14.

[18] M. O. Rabin, "Efficient dispersal of information for security, load balancing, and fault tolerance," *Journal of the ACM (JACM)*, vol. 36, no. 2, pp. 335–348, 1989.

[19] J. S. Plank *et al.*, "A tutorial on reed-solomon coding for fault-tolerance in raid-like systems," *Softw., Pract. Exper.*, vol. 27, no. 9, pp. 995–1012, 1997.

[20] J. S. Plank and Y. Ding, "Note: Correction to the 1997 tutorial on reed–solomon coding," *Software: Practice and Experience*, vol. 35, no. 2, pp. 189–194, 2005.

[21] T. Guo, F. Liu, C. Wu, C. Yang, W. Wang, and Y. Ren, "Threshold secret image sharing," in *International Conference on Information and Communications Security*. Springer, 2013, pp. 404–412.

[22] A. Jolfaei, X.-W. Wu, and V. Muthukkumarasamy, "On the security of permutation-only image encryption schemes," *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 2, pp. 235–246, 2016.

[23] C.-C. Lin and W.-H. Tsai, "Secret image sharing with steganography and authentication," *Journal of Systems and software*, vol. 73, no. 3, pp. 405–414, 2004.

[24] A. Beimel, "Secret-sharing schemes: a survey," in *International Conference on Coding and Cryptology*. Springer, 2011, pp. 11–46.

[25] H. Krawczyk, "Secret sharing made short," in *Annual International Cryptology Conference*. Springer, 1993, pp. 136–146.