Participant Authenticating, Error Detecting, and 100% Multiple Errors Repairing Chang-Chen-Wang's Secret Sharing Method Enhancement

Alexander G. Chefranov
Eastern Mediterranean University
Department of Computer Engineering, EMU,
Famagusta, North Cyprus, via Mersin 10, Turkey
+90 392 630 1190
Alexander.chefranov@emu.edu.tr

Amir Narimani
Eastern Mediterranean University
Department of Computer Engineering, EMU,
Famagusta, North Cyprus, via Mersin 10, Turkey

ABSTRACT
Chang-Chen-Wang's *(3,n)* Secret grayscale image Sharing between *n* grayscale cover images method with participant Authentication and damaged pixels Repairing (SSAR) properties is analyzed; it restores the secret image from any three of the cover images used. We show that SSAR may fail, is not able fake participant recognizing, and has limited by 62.5% repairing ability. We propose SSAR *(4,n)* enhancement, SSAR-E, allowing 100% exact restoration of a corrupted pixel using any four of *n* covers, and recognizing a fake participant with the help of cryptographic hash functions with 5-bit values that allows better (vs. 4 bits) error detection. Using a special permutation with only one loop including all the secret image pixels, SSAR-E is able restoring all the secret image damaged pixels having just one correct pixel left. SSAR-E allows restoring the secret image to authorized parties only contrary to SSAR. The performance and size of cover images for SSAR-E are the same as for SSAR.

CCS Concepts
• **Security and privacy**

Keywords
Steganography; secret image; cover image; secret sharing; error detection; repairing; authentication.

1. INTRODUCTION
Chang-Chen-Wang's *(3,n)* Secret Sharing with Authentication and Repairing (SSAR) method [1] is intended for sharing a secret grayscale image *SI* by $n \geq 3$ cover grayscale images *{C1,..Cn}* so that exact reconstruction of the secret is possible having any three of the *n* cover images. It is considered as one of the best secret sharing algorithms (see [4]; Table 4, p. 186, in [5]; Tables 1, 2, pp. 1076, 1077 in [8]; p. 198 in [9]). The secret image is embedded into each *k*-th cover image in the form of base 5 numbers N_{ik} obtained for *i*-th *8*-bit pixel of a secret image SI_i, *i=1,..,S, k=1,..,n,* using Least Significant Base-*5* Digit (LSB5D) method similar to Least Significant Bit (LSB) [1, 6]. SSAR provides authentication and error detection by the use of a *4*-bit hash function of the obtained for *i*-th secret image pixel authenticator au_{ik}, *i=1,..,S, k=1,..,n*. Repairing ability in the SSAR method is based on the use of a complementary dual chain somehow resembling reparation facilities of DNA [3]. We show that SSAR algorithm has such problems as not ability to counter fake participant attack, limited opportunity for error detection based on four-bit only hash function value, only five out of eight bits (62.5%) recovery in the case of damaging, and is not correct under made assumption of mutually exclusive cover identifiers. We propose a *(4,n)* enhancement of SSAR, SSAR-E, solving all the problems mentioned above and allowing also exact repairing of up to *S-1* damaged secret image pixels out of *S*. The rest of the paper is organized as follows. Section 2 introduces SSAR details and analysis revealing its problems. Section 3 describes the proposed enhancement, SSAR-E, and proves its features. Section 4 concludes the paper.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
SIN'16, July 20–22, 2016, Newark, New Jersey, USA.
Copyright 2016 ACM 978-1-4503-4764-8 …$15.00.
DOI: http://dx.doi.org/10.1145/2947626.2951958

2. REVIEW AND ANALYSIS OF SSAR
The SSAR method takes as input the secret image, *SI*, which may be represented as a sequence of byte-size pixels, *LSi, i=1,..,S*, and a set of cover images, *Ci, i=1,..,n >=3*. Each cover image, *Ci*, has a unique identifier, id_i. SSAR has embedding, extraction, and repairing parts as follows.

SSAR Embedding Part (Steps 1-7):Step 1. Produce a dual sequence LS'=permutation(K,LS). Denote bits of the i-th pixel, LSi, as $a_{1i},..,a_{8i}$, and of its partner, LS'i, as $b_{1i},..,b_{8i}$, i=1,..,S.

Step 2. For each i, define three quantities,

$$\gamma_i = \sum_{k=1}^{2} a_{k+6,i} 2^{2-k} \in \{0,..3\},\ i=1,..,S,$$

$$\alpha_i = \sum_{k=1}^{3} a_{ki} 2^{3-k} \in \{0,..,7\}, \beta_i = \sum_{k=1}^{3} a_{k+3,i} 2^{3-k} \in \{0,..,7\}, \quad (1)$$

using the secret image chain, LS.

Step 3. Concatenate the bits, $b_{1i},..,b_{4i}$, of the partner pixel, LS'i, with the three quantities (1), getting

$$\alpha_i' = \alpha_i + 8b_{1i} \in \{0,..,15\}, \beta_i' = \beta_i + 8b_{2i} \in \{0,..,15\}, \gamma_i' = \gamma_i + 8b_{3i} + 4b_{4i} \in \{0,..,15\}, \quad i=1,..,S. \quad (2)$$

Step 4. Using (2), calculate an authenticator,

$$au_{ik} = (\alpha_i' + \beta_i' \cdot id_i + \gamma_i' \cdot id_i^2) \bmod 17 \in \{0,..,16\} \quad (3)$$

for each pixel and each cover, i=1,..,S, k=1,.., $n \geq 3$. Thus, for

each pixel, we get a system of $n \geq 3$ *equations that allows solving them with respect to* $\alpha_i', \beta_i', \gamma_i'$ *after embedding of the authenticators into the cover images by a sender, and the next extraction of the authenticators from the cover images by a receiver.*

Step 5. For each authenticator (3), au_{ik} *, calculate its 4-bit hash function value,*

$$hv_{ik} = hash(i \,||\, au_{ik}) \in \{0,..,15\}\ ,\ i=1,..,S, k=1,..,n\ , \quad (4)$$

where || is the concatenation operation.

Step 6. Using (3), (4), generate a number

$$N_{ik} = au_{ik} \cdot 34 + hv_{ik} \cdot 2 + b_{5i} \in \{0,..,575\} \quad (5)$$

by mixing an authenticator, its hash, and 5-th bit of LS'i in a way allowing getting back its constituent parts. The number can be represented as a four-digit base-5 number.

Step 7. Each cover image, Ck, is represented as a sequence of S 4-pixel blocks so that N_{ik} *in the form of four base-5 digit number,* $N_{ik}[1..4]$*, is embedded into four consecutive pixels Ck[4(i-1)+1,..,4i]:*

$$C_k[4(i-1)+j]' = C_k[4(i-1)+j] - C_k[4(i-1)+j] \bmod 5 + N_{ik}[j],\ j=1,..,4,\ i=1,..,S,\ k=1,..,n. \quad (6)$$

Embedding in [1], see Fig. 4 therein, uses shift of the result of (6) by $R \in \{-5,5\}$ to minimize the pixel distortion, however, as is, it increases it; to fix the problem, sign of R shall be inverted.

Consider now SSAR extraction part that takes *n* covers with embedded by (6) numbers N_{ik} representing parts of the secret image, *SI,* and restores them and their ingredients; restored values have "*R"* in their names.

SSAR Extraction Part (Steps 1-6)

Step 1. Restore digits of the embedded by (6) number

$$NR_{ik}[j] = C_k'[4(i-1)+j] \bmod 5,\ j=1,..,4,\ i=1,..,S,\ k=1,..,n. \quad (7)$$

Step 2. Restore constituent parts of the numbers (7) using (5)

$$auR_{ik} = NR_{ik} div 34; hvR_{ik} = (NR_{ik} \bmod 34) div 2; bR_{5i} = NR_{ik} \bmod 2 \quad (8)$$

Step 3. Check errors (authenticity) of the restored by (8) authenticator by restored authenticator, auR_{ik} *, hash recalculation and comparing it versus its restored hash function value (see (8)):*

$$hash(i \,||\, auR_{ik}) == hvR_{ik}\ . \quad (9)$$

If (9) is true then the authenticator is considered valid, otherwise an error is detected.

Step 4. If checking in (9) of Step 3 is true (no error) for any three covers, then three entities, $\alpha R_i', \beta R_i', \gamma R_i'$*, used in calculation of authenticators (3), are restored from (3) by solving a system of at least three linear modulo 17 algebraic equations with respect to the three unknown entities. In [1], see (12) therein, this solution is represented as three modulo 17 formulas having differences of the covers' identifiers in the denominators that is expected to be safe due to the condition of mutually exclusive covers' identifiers.*

Step 5. Having restored $\alpha R_i', \beta R_i', \gamma R_i'$*, the values* $\alpha R_i\ , \beta R_i\ , \gamma R_i$ *together with* $bR_{1i},..,bR_{4i}$ *from(1), (2) can be then*

obtained as follows

$$\alpha R_i = \alpha R_i' \bmod 8, \beta R_i = \beta R_i' \bmod 8, \gamma R_i = \gamma R_i' \bmod 4, bR_{1i} = \alpha R_i' div 8,$$
$$bR_{2i} = \beta R_i' div 8, bR_{3i} = \gamma R_i' div 8, bR_{4i} = (\gamma R_i' div 4) \bmod 2. \quad (10)$$

Step 6. From (10), the original pixel is restored using (1)

$$LS_i = 32 \cdot \alpha R_i + 4 \cdot \beta R_i + \gamma R_i\ . \quad (11)$$

SSAR Repairing Part

Obtained in (8) and (10) five bits $bR_{1i},..,bR_{5i}$ *of the partner pixel LS'i can be used for its restoration in the case of its damaging.*

Let's illustrate SSAR by the following Example 1.

Example 1. Let the secret image size *S=6*, secret image sequence of pixels is *LS=(15,34,49,105,217,28)*, permutation is *(4,2,1,3,6,5), LS'=(105,34,15,49,28,217)*. Let the number of cover images *n=5*, each of them having *4*S=24* pixels. Consider embedding of the pixel $LS_1=15$ into the *5* cover images. SSAR Embedding Part Step 1 is already applied. According to Step 2 equations (1) $\alpha_1 = 0, \beta_1 = 3, \gamma_1 = 3$. Using (11), we see that actually $LS_1=32*0+4*3+3=15$. According to Step 3 equations (2), and taking into account that $LS_4=105=(0110\ 1001)$, we get $\alpha_1' = 0, \beta_1' = 11, \gamma_1' = 11$. From the last values, according to (10), we can get back correct values $\alpha_1 = 0, \beta_1 = 3, \gamma_1 = 3, b_{11} = 0, b_{21} = 1, b_{31} = 1, b_{41} = 0$

Calculate now the authenticators of LS_1 pixel according to (3) using covers' identifiers as $id_1=1$, $id_2=2$, $id_3=3$, $id_4=4$, $id_5=5$:

$$(au_{11},..,au_{15}) = (5,15,13,16,7). \quad (12)$$

If any three of the authenticators (12) are available, we can get $\alpha_1', \beta_1', \gamma_1'$ by solving a system of equations (3). Say, if the first three authenticators are restored then we have the following system:

$$au_{11} = 5 = (\alpha_1' + \beta_1'' \cdot 1 + \gamma_1' \cdot 1) \bmod 17, au_{12} = 15 = (\alpha_1' + \beta_1'' \cdot 2 + \gamma_1' \cdot 4) \bmod 17, au_{13} = 13 = (\alpha_1' + \beta_1'' \cdot 3 + \gamma_1' \cdot 9) \bmod 17. \quad (13)$$

The system (13) has Vandermonde matrix [7] of the coefficients and may be solved, e.g., using Cramer's rule [2] (determinant of the matrix of the system coefficients with the column of the coefficients near the unknown value under consideration substituted by the column of the known values is divided by the determinant of the coefficient matrix):

$$\alpha_1' = = \frac{-34}{2} \bmod 17 = 0, \beta_1' = \frac{56}{2} \bmod 17 = 11, \quad \gamma_1' = \frac{-12}{2} = -6 \bmod 17 = 11. \quad (14)$$

As far as the identifiers appearing in (3) are mutually exclusive, Vandermonde determinant [7] used in denominators of (14) is non-zero, and the solution exists. Next, in Step 5, according to (4), hash function values are calculated for the authenticators. In [1],

equations (6)-(8), define how 4-bit resulting hash function value is to be calculated using some generic hash function. For simplicity, and getting particular results, let hash function be $h(x)=(13x+11)\bmod 16$. Then, according to (4) in Step 5,

$$hv_{11}=h(1\,||\,au_{11})=h(1\,||\,5)=h(37)=12,\\ hv_{12}=14, hv_{13}=4, hv_{14}=11, hv_{15}=6. \quad (15)$$

Now, let us apply Step 6 using (5), (12), and (15) and taking into account that $LS_4=(b_{11}..b_{81})$=(0110 1001), we get the numbers for embedding and their representation as 4-digit base-5 numbers:

$$N_{11}=au_{11}\cdot 34+hv_{11}\cdot 2+b_{51}=5\cdot 34+12\cdot 2+1=195=1240_5,\\ N_{12}=539=4124_5, N_{13}=451=3301_5, N_{14}=567=4232_5, N_{15}=251=2001_5. \quad (16)$$

Let us assume that the first four bytes of the five cover images, where the five numbers (16) are to be embedded, are as follows:

C_1=(12,240,251,128,...),C_2=(137,49,56,122,..),C_3=(17,1,67,78,..),
C_4=(190,217,218,16,..), C_5=(123,213,65,41,..). (17)

From (16), (17), according to (6) in Step 7,

C_1=(11,242,254,125,...), C_2=(139,46,57,124,..), $_3$=(18,3,65,76,..),
C_4=(194,217,218,17,..), C_5=(122,210,65,41,..). (18)

For the Step 1 of the SSAR Extraction Part, applying modulo 5 operations to (18), we restore the 4-digit base 5 numbers (16), 1240_5=195, 4124_5=539, 3301_5=451, 4232_5=567, 2001_5=251. From (16), applying Step 2 equations (8), we restore authenticators (12) as *5, 15, 13, 16, 7*, their hash function values (15) as *12,14,4,11,6,* and b_{51}=1. Checking of (9) in Step 3 returns all true values, hence, any three of the five authenticators obtained may be used for restoring $\alpha_1'=0, \beta_1'=11, \gamma_1'=11$, as it is made in (13), (14) for the first three authenticators in accordance with Step 4. And lastly, the original values $\alpha_1=0, \beta_1=3, \gamma_1=3$ are restored according to Step 5 equation (10) together with $(b_{11}, b_{21}, b_{31}, b_{41})=(0110)$. Thus, we see how the SSAR method works for embedding/extraction a secret image pixel and five bits of its partner into/from four consecutive pixels of the cover images. If in our example the partner pixel after extraction is found out in the checking (9) to be incorrect, its five most significant bits can be gained from the results of extraction of LS_1 containing in the case under consideration five correct most significant bits of LS_4. As it was stated in the Section 1, SSAR under the assumption used of the mutually exclusive covers' identifiers may be working incorrectly. This we prove by the following counterexample.

Example 2. Let's consider the same conditions as in the Example 1, but the identifiers of the cover images are *(1,2,18,4,5)*, instead of the used *(1,2,3,4,5)*. Then in (12), (13)

$$au_{13}=(0+11\cdot 18+11\cdot 324)\bmod 17=5, \quad (19)$$

and the determinant in the denominators of (14) is as follows

$$\det\begin{vmatrix}1 & 1 & 1\\ 1 & 2 & 4\\ 1 & 18 & 324\end{vmatrix}\bmod 17=272\bmod 17=0 \quad (20)$$

As far as the determinant modulo 17 is equal to zero in (20), the solution of (13) does not exist, and SSAR fails. It is claimed in [1] that SSAR can recognize malicious users with the help of the hash function but the latter is known to an opponent and can be used by fake users to get a valid hash for the artificially created authenticators. Also, the hash function used is only four-bit, and needs extension for greater reliability.

3. PROPOSED SSAR-E

Our proposed enhancement basically is the same as SSAR but has the following presented below.

Modification 1 (to have the SSAR correctly working). Covers' identifiers shall be not just mutually exclusive as required in [1], p. 3075, Section 3.1, but mutually exclusive modulo 17, i.e.

$(\forall i,j=1,..,n)(i\neq j\rightarrow id_i-id_j\neq 0\bmod 17$. In that case, actually, systems like (13) are always solvable because the determinant of the coefficient square Vandermonde matrix [8] is not zero modulo 17 as it is required by the algorithm.

Modification 2 (to have an opportunity of repairing all eight bits of the partner pixel, not just five bits). In SSAR-E, we work with four two-bit entities, and SSAR Embedding Part, Steps 2-4, are rewritten as follows.

Step 2. For each i, define four quantities,

$$\alpha_i=\sum_{k=1}^{2}a_{ki}2^{2-k}\in\{0,..,3\}, \beta_i=\sum_{k=1}^{2}a_{k+2,i}2^{2-k}\in\{0,..,3\}, i=\overline{1,S}\\ \gamma_i=\sum_{k=1}^{2}a_{k+4,i}2^{2-k}\in\{0,..,3\}, \delta_i=\sum_{k=1}^{2}a_{k+6,i}2^{2-k}\in\{0,..,3\} \quad (21)$$

Step 3. Concatenate the bits $b_{1i},..,b_{8i}$, of the partner pixel, LS'_i, with the four quantities (21), getting

$$\alpha_i'=\alpha_i+8b_{1i}+4b_{2i}\in\{0,..,15\}, \beta_i'=\beta_i+8b_{3i}+4b_{4i}\in\{0,..,15\}\\ \gamma_i'=\gamma_i+8b_{5i}+4b_{6i}\in\{0,..,15\}, \delta_i'=\delta_i+8b_{7i}+4b_{8i}\in\{0,..,15\}, \quad (22)$$

i=1,..,S.

Step 4. Using (22), calculate an authenticator,

$$au_{ik}=(\alpha_i'+\beta_i'\cdot id_i+\gamma_i'\cdot id_i^2+\delta_i'\cdot id_i^3)\bmod 17\in\{0,..,16\} \quad (23)$$

for each pixel and each cover, i=1,..,S, k=1,.., $n\geq 4$.

Also Steps 4-6 of the SSAR Extraction Part are as follows.

Step 4. If checking in (9) of Step 3 is true (no error) for any four covers, then four entities, $\alpha R_i', \beta R_i', \gamma R_i', \delta R_i'$, used in calculation of authenticators (23) are restored from (23) by solving a system of at least four linear modulo 17 algebraic equations with respect to the four unknown entities.

Step 5. Having restored $\alpha R_i', \beta R_i', \gamma R_i', \delta R_i'$, values $\alpha R_i, \beta R_i, \gamma R_i, \delta R_i$ together with $bR_{1i},..,bR_{8i}$ from (22):

$$\alpha R_i=\alpha R_i'\bmod 4, \beta R_i=\beta R_i'\bmod 4, \gamma R_i=\gamma R_i'\bmod 4, \delta R_i=\delta R_i'\bmod 4,\\ bR_{1i}=\alpha R_i' div 8, bR_{2i}=(\alpha R_i' div 4)\bmod 2, bR_{3i}=\beta R_i' div 8,\\ bR_{4i}=(\beta R_i' div 4)\bmod 2, bR_{5i}=\gamma R_i' div 8, bR_{6i}=(\gamma R_i' div 4)\bmod 2,\\ bR_{7i}=\delta R_i' div 8, bR_{8i}=(\delta R_i' div 4)\bmod 2 \quad (24)$$

Step 6. From (24), the original pixel is restored using (21)

$$LS_i = 64 \cdot \alpha R_i + 16 \cdot \beta R_i + 4\gamma R_i + \delta R_i \quad . \qquad (25)$$

Obtained in (24) eight bits $bR_{1i},..,bR_{8i}$ of the partner pixel LS'$_i$ can be used for its restoration in the case of its damaging.

Modification 3 (to have SSAR resistant to fake participant attack and hash function value bit number extension from 4 to 5). To resist the attack, the hash function shall be cryptographic. Also, one bit used in the number (5) for keeping 5-th bit of the partner pixel, now can be used for the hash function value keeping. Thus, Steps 5-6 of the SSAR Embedding Part are rewritten as follows.

Step 5. For each authenticator (23), au_{ik} , calculate its 5-bit hash function value,

$$hv_{ik} = hash_{SK}(i \,||\, au_{ik}) \in \{0,..,31\},\ i=1,..,S,\ k=1,..,n, \qquad (26)$$

where SK is a secret key shared by the valid communicating parties.

Step 6. Using (23), (26), generate a number

$$N_{ik} = au_{ik} \cdot 34 + hv_{ik} \in \{0,..,575\} \qquad (27)$$

by mixing an authenticator and its hash in a way allowing getting back its constituent parts. The number can be represented as a four-digit base-5 number.

SSAR Extraction Part Steps 2-3 are also to be rewritten accordingly:

Step 2. Restore constituent parts of the numbers (7) using (27)

$$auR_{ik} = NR_{ik} div 34; hvR_{ik} = NR_{ik} \bmod 34. \qquad (28)$$

Step 3. Check errors (authenticity) of the restored by (28) authenticator by recalculating of the hash of the restored authenticator, auR_{ik} , and comparing it versus its restored hash function value (see (28)):

$$hash_{SK}(i \,||\, auR_{ik}) == hvR_{ik}. \qquad (29)$$

If (29) is true then the authenticator is considered valid, otherwise an error is detected.

Modification 4 (to have an opportunity of repairing *S-1* damaged pixels out of *S*). In the Example 1, the permutation *(4,2,1,3,6,5)* was used so that the partner for LS_1 is LS_4 (LS_1=>LS_4), for LS_2 is LS_2 (LS_2=>LS_2), for LS_3 is LS_1 (LS_3=>LS_1), for LS_4 is LS_3 (LS_4=>LS_3), for LS_5 is LS_6 (LS_5=>LS_6), and for LS_6 is LS_5 (LS_6=>LS_5), where A=>B denotes that B is a partner of A. The permutation defines three loops of partnership: LS_1=>LS_4=>LS_3=>LS_1, LS_2=>LS_2, LS_5=>LS_6=>LS_5. If a loop of partnership of length three, then if two its elements are damaged, they can be restored from only one correct remained in the loop pixel. Hence, the permutations used in the SSAR-E have only one loop involving all pixels of the image providing an opportunity of repairing of *S-1* pixels out of *S* just from one remaining correct pixel. Thus, we rewrite Step 1 as follows.

Step 1. Produce a dual sequence LS'=permutation(LS), where permutation() involves all S elements in a single loop. Denote bits of LS_i as $a_{1i},..,a_{8i}$, and of its partner LS'$_i$ as $b_{1i},..,b_{8i}$, i=1,..,S.

Modification 5 (to allow the secret image disclosing to the authorized parties only). Cryptographic hiding of the sequence of authenticators may be made, e.g., by the use of the Counter mode of any block cipher (see [6], p. 97 therein. Initial counter value and the cipher key shall be shared by the authorized sender and receiver. Thus, we introduce in SSAR-E Embedding Part the Step 4' for hiding the authenticators, and Step 3' in the SSAR-E Extraction Part just after Step 4 and Step 3, respectively.

SSAR-E Embedding Part Step 4'. Encrypt the authenticator, au_{ik} , using an encryption algorithm, E , in the Counter Mode with secret key, SK1, and current Counter value as follows:

$$au_{ik} = (au_{ik} + E_{SK1}(Counter)) \bmod 17; Counter = Counter + 1. SSA$$

R-E Extraction Part Step 3'. Decrypt the authenticator, au_{ik} , using an encryption algorithm, E in the Counter Mode with secret key, SK1, and current Counter value as follows:

$$auR_{ik} = (auR_{ik} - E_{SK1}(Counter)) \bmod 17; Counter = Counter + 1.$$

4. CONCLUSION

In [1], the SSAR algorithm is proposed claiming to solve the problems of authentication and repairing. It uses three-entity representation of every secret image byte and mixes the entities with four bits of the partner byte from the dual byte sequence. For each such set of three entities and each of *n* cover images having a unique identifier, an authenticator is calculated and embedded with its hash value into respective cover images. It I supposed that any three authenticators can be used for unique restoration of the three entities and the secret image byte may be restored, so SSAR is a *(3,n)* secret sharing method. We show by a counterexample that SSAR may fail because mutually exclusive modulo 17 identifiers are not used; they used in SSAR-E. In SSAR-E, *(4,n)* secret sharing, all eight bits of the partner byte may be repaired (100%) contrary to (62.5%) in SSAR. Also, in SSAR-E, a 5-bit cryptographic hash function is used providing greater reliability and countering fake participant attack, and the permutation for the dual sequence generating is selected so that up to *S-1* corrupted bytes in the partnership loop can be restored from one correct byte.

5. REFERENCES

[1] Chang, C.-C. , Chen, Y.-H., and Wang, H.-C. 2011. Meaningful secret sharing technique with authentication and remedy abilities. *Inf. Sci.* 181, 3073-3084. DOI= 10.1016/j.ins.2011.03.002

[2] https://en.wikipedia.org/wiki/Cramer%27s_rule, Cramer's rule, last access date 19.05.2016.

[3] https://en.wikipedia.org/wiki/DNA_repair, DNA repair,last access date 19.05.2016.

[4] Lin, P.-Y. , Chen, Y.-H. , Hsu, M.-C., and Juang, F.-M. 2013. Secret sharing mechanism with cheater detection. In *Proceedings of 2013 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA 2013* (Kaohsiung, Oct. 29 -Nov. 1, 2013), IEEE, 1-4. DOI= 10.1109/APSIPA.2013.6694288.

[5] Salehi, S. and Balafar, M.A. 2015. An investigation of image secret sharing. *Int. J. Sec. and its Appl.*, 9, 3, 163-190. DOI= http://dx.doi.org/10.14257/ijseia.2015.9.3.16

[6] Stallings, W 2011. Cryptography and network security. Principles and practice. 5th Edition, Pearson Education, Inc., Boston. ISBN 13: 978-0-13-609704-4

[7] https://en.wikipedia.org/wiki/Vandermonde_matrix, Vandermonde matrix,last access date 19.05.2016.

[8] Wu X. and Sun W. 2013. Secret image sharing scheme with authentication and remedy abilities based on cellular automata and discrete wavelet transform. *J. Syst. Softw.* 86, 4 (Apr 2013), 1068-1088.

[9] Yuan H.-D. 2014. Secret Sharing with Multi-Cover Adaptive Steganography. *Inf. Sci.* 254, 197-212. DOI= http://dx.doi.org/10.1016/j.ins.2013.08.012