Designing Robust LMCA-based Threshold Secret Sharing Scheme for Digital Images Using Multiple Configurations Assignment

Djihed Anani, and Kamel Mohamed Faraoun

Original scientific paper

***Abstract* — In this paper, we present a new (t,n)-threshold secret images sharing scheme based on linear memory cellular automata (LMCA). While all existing LMCA-based sharing scheme are not robust, the proposed one provides full robustness property. Precisely, any subset of t participants can collude to recover the shared secret, in contrast to existing LMCA-based schemes when this is possible only for participants having consecutive shares. To achieve robustness, produced shares are constructed using subsets of different LMCA's configurations instead of using single ones. The subsets are defined according to an assignments matrix that is generated using a specific heuristic. The proposed scheme is shown to be robust, and its security is experimentally evaluated with respect to the problem of secret color image sharing. Obtained results illustrate the secrecy of the produced shares, while comparison gives an accurate evaluation with respect to existing schemes.**

***Index Terms* — Threshold secret sharing, linear memory cellular automata, sharing robustness, assignment matrix.**

I. INTRODUCTION

Secret sharing schemes are cryptographic procedures used for sharing a given secret among a set of n different participants. Each one receives a different data block named a share, and when required, only qualified subset of participants can colludes and combines their shares to recover the original secret. Particularly, a (t,n)-threshold secret sharing scheme allows secret's reconstruction only for subsets of t or more different participants, while any subset of (t-1) or less participants is unable to recover any useful information about the secret. Secret sharing schemes have many applications in different areas such as electronic-voting, threshold access control, e-auction and anonymous token to name a few.

Several secret sharing schemes have been proposed during the last decades. The first (t,n)-threshold scheme was proposed in 1979 by Shamir [1] based on Lagrange polynomial interpolation, using the fact that at less t different points are necessary to define a $(t-1)^{th}$ degree polynomial. In the same year, Blackley [2] proposed another threshold scheme based on plane geometry using the fact that any n nonparallel (n-1)-dimensional hyperplanes intersect at a specific point, and the secret may be encoded as any single coordinate of the point of intersection. Each participant is given enough information to define a hyperplane, and the secret is recovered by calculating the plane's point of intersection and then taking a specified coordinate of that intersection. In 1983, Asmuth, Bloom [3] and Mignotte [4] proposed independently another threshold secret sharing schemes using the Chinese Remainder Theorem (CRT): the shares are generated by reducing the secret modulo a set of relatively primes integers $m_1,m_2,\ldots m_n$, when the construction can be performed by essentially solving the system of t congruence using the CRT.

All mentioned sharing schemes are unconditionally secure and permit to solve the sharing problem in an efficient manner. However, their computational complexity for sharing and reconstructing secrets is polynomial. When dealing with large sized secrets such as digital images or multimedia content, these approaches are not suitable, and can difficultly be adapted to real-time scenarios. Recently, a new model of threshold secret sharing approach exploits the cellular automata paradigm and more precisely the linear memory ones (LMCA). The LMCA model provides linear complexity for both sharing and reconstruction phases, and leads to best runtime performances for large-scaled secrets. The first attempt for using LMCA in secret sharing has been proposed in [5] by considering the secret as an initial configuration of a t-order LMCA, and randomly generate the remaining (t-1) configurations. The evolution of the constructed LMCA produces n consecutive configurations used to define the n different shares distributed among the n participants. Running the LMCA backward starting from any set of t consecutive shares, permits a perfect reconstruction of the shared secret in a linear time with respect to the size of secret. The LMCA based sharing approach has been enhanced later in [6] using two-dimensional cellular automata to handle images sharing, then recently, many other variants have been developed: in [7], the authors use steganography to build a lossy sharing scheme, when in [8] a discrete logarithm based signature verification is combined with LMCAs to provide authentication of the different shares. Wu and al proposed in [9] another combination of steganography and LMCA to build a user-friendly secret sharing scheme but only a lossy secret

Manuscript received March 11, 2015; revised June 19, 2015

D. Anani is with the Department of computer science, Djilalli Liabbes University, Sidi Bel Abbès, Algeria.(e-mail: jihed-91@hotmail.com).

K. M. Faraoun is with the Department of Computer science, Djilalli Liabbes University, Sidi Bel Abbès, Algeria. (e-mail: kamel_mh@yahoo.fr).

1845-6421/06/8390

image can be reconstructed. The same authors proposed an authenticated image sharing approach in [10] by combining wavelets transforms and LMCAs, but unfortunately, they detailed only (3,n) threshold instance of the sharing problem and assumed that extension to the (t,n) general case is feasible.

Even if all LMCA based approaches provide best performances with respect to the standard Shamir's sharing scheme, they all have a major drawback that make them un-useful for real applications: not all possible sets of t shares permit to recover the secret, but only those having consecutive ones. Unfortunately, for any given values of t and n ($t \leq n$), the number of possible sets of t shares having consecutive elements equal to (n-t+1) is very small with respect to the whole set of possible sets of t shares equal to ($C_n^t = \frac{n!}{t!(n-t)!}$). For example, when (t=3) and (n=10), we have ($C_{10}^3 = 120$) possible set of t shares, when only (10-3+1=8) from them verify the property of consecutive elements. Hence, less than (7%) of the possible sets of t participants can recover the secret. Such problem makes the LMCA-based secret sharing scheme non-robust, and consequently inappropriate for the use in real scenario applications.

In the present work, we propose a solution to the robustness problem of the LMCA-based secret sharing schemes. In contrast to existing ones, the proposed LMCA-based (t,n)-threshold secret sharing scheme allows any subset of t participants (let C_n^t subsets) to reconstruct the secret using a specific matrix of configuration's assignment: instead of giving each participant a single configuration as a share, a specific set of different configurations is assigned to each participant. Configurations affected to each participant are constructed in a such a way that the union of t different subsets of t participants contain always t different consecutive configurations, while regrouping t-1 or less subsets do not permit to achieve such constraint. The assignment of the configurations is performed using a specific assignment matrix constructed using a heuristically proposed algorithm, while sharing and reconstruction are performed as usual using the mechanism of LMCA's evolution. The remaining of this paper is organized as follows: in Section 2, basic definitions about one dimensional cellular automata, LMCAs and reversibility are introduced briefly. In Section 3, the proposed scheme is described. Security analysis and proofs of the schemes are presented in section 4 with a set of experimental results. Finally, conclusions are drawn in Section 5.

II. THEORETICAL PRELIMINARIES

In this section, we briefly present the main definitions of one dimensional cellular automata, linear memory cellular automata (LMCA) and the main existing LMCA-based secret sharing schemes with related definitions and security aspects.

A. One dimensional cellular automata

A cellular automata consist of a number of cells arranged in a regular lattice, each cell has its own state that can change in a discrete time step. States of the whole CA's cells are updated synchronously using a local transition rule that define each new cell's state using its old state, and the states of the corresponding neighbors. The neighbors are a specific selection of cells relatively chosen with respect to a given cell's position that can be defined for each cell using a radius r on the lattice. This will give 2r+1 different neighbor including the cell itself. The boundaries cells of the lattice are concatenated together in a cyclic form to deal with finite size automaton. If the same update rule is used for all the cells then the resulting CA is named uniform. Otherwise, if a different transition rule is used each time the cell's position change, the resulting CA is named non-uniform.

Formally, if we define the state of a cell i at the time t with $s^t_{i,}$, its state on time t+1 will depend only on the states of the corresponding neighborhood at the time t, by applying a transition rule that define the way states are updated. If the neighborhood radius is r, and if only two cell states are defined (0 or 1), the length of each transition rule is then 2^{2r+1} bit, and the number of possible rules is equal to $2^{2^{2r+1}}$. The transition rule of one dimensional binary CAs is generally coded using the integer value of the corresponding binary representation, when the different CA's configurations are represented by binary blocks.

In contrast to elementary cellular automata, reversible cellular automata (RCAs) are a specific case in which every configuration has only one unique predecessor. Precisely, RCAs are constructed in such a manner that state of each cell prior to an update is determined uniquely from the updated states of all the cells. Several approaches have been defined to construct reversible cellular automata rules. The second-order cellular automaton method introduced firstly in [10], in which the update rule combines states from two previous steps of the automata, permits to turn any one-dimensional binary rule into a reversible one using the fact that the state of a cell at time t depends not only on its neighborhood at time t-1, but also on its state at time t-2. This is ensured by combining the i^{th} cell state at time t with the state of the same cell in time t-2 using the xor operator.

If the configuration of a given CA at each time step t is defined by C^t, then we can build a second-order RCA using the following equation:

$$C^t=F(C^{t-1}) \oplus C^{t-2} \quad (1)$$

when the map "F" denotes the global transition function of the related basic CA. Such defined RCA can be reversed trivially using the following equation:

$$C^{t-2}=F(C^{t-1}) \oplus C^t \quad (2)$$

The RCAs defined according to equations (1) can always be reversed reversible even if the basic underlying CA defined by F is not. Hence, we can construct as mush RCAs as possible existing CAs.

Instead of using one initial configuration like standard one-dimensional CA, two initial configurations are used to evolve a second-order RCA. After applying m iteration steps on two initial configurations C^0 and C^1 we can obtain two consecutive configurations C^m and C^{m+1}. When running the same RCA backward starting from C^m and C^{m+1} as initial configurations,

we recover the two configurations C^0 and C^1 after exactly m iteration using the same transition rule. Reversion of RCAs is raising qualitatively the same behavior of one-order CAs as pointed by Wolfram [11].

B. Reversible Linear memory cellular automata (LMCA)

An extension of the second order reversible cellular automata is defined by m-order reversible cellular automata (m-order RCAs). The same principle is applied since m consecutive configurations are used to build a new one, and by the same manner, m consecutive configurations are used to run the automata backward and recover initial states. Particularly, a specific case of m-order RCA is the linear memory cellular automata (LMCAs) that use specific linear transition rules. Let's consider the set of CAs of size n and symmetric neighborhoods of radius r, whose local transition function are of the following form:

$$s_i^{(t+1)} = \sum_{j=-r}^{r} \lambda_j s_{i+j}^{(t)} (mod\ 2) \quad (3)$$

where $0 \leq i \leq n-1$, and $\lambda_j \in \{0,1\}$. These are called linear cellular automata (LCAs). As there are 2r+1 cells in the symmetric neighborhood of radius r, then there exist 2^{2r+1} different LCAs which goes from 0 to $2^{2r+1}-1$, and each LCA is conveniently specified by a decimal integer ω representing the rule number by:

$$\omega = \sum_{j=-r}^{r} \lambda_j 2^{r+j} \quad (4)$$

In the same way like RCAs, the state of every LMCA's cell at time t+1 depends on the states of its neighbor cells at different time steps t, t-1, t-2, …,t-m. Particularly, using linear transition rules defined by equation (4), one can define m-order LMCA whose local transition function takes the following form:

$$s_i^{(t+1)} = f_{\omega_1}(V_i^{(t)}) + f_{\omega_2}(V_i^{(t-1)}) + \cdots + f_{\omega_m}(V_i^{(t-m+1)}) (mod\ 2) \quad (5)$$

where $0 \leq i \leq n-1$, and $\omega_1, \omega_2, \ldots, \omega_m \in \{0,1,\ldots,2^{2r+1}-1\}$. In this case, in order to start the evolution of the LMCA, m initial configurations are required. The following proposition describes how to construct a reversible LMCA.

Proposition II.1. If $f_{\omega_m}(V_i^{(t-m+1)}) = s_i^{(t-m+1)}$, then the LMCA expressed by:

$$s_i^{(t+1)} = f_{\omega_1}(V_i^{(t)}) + \cdots + f_{\omega_{m-1}}(V_i^{(t-m+2)}) + s_i^{(t-m+1)} (mod\ 2) \quad (6)$$

is reversible and its reverse is another LMCA with the following local transition function:

$$s_i^{(t+1)} = f\left(V_i^{(t)}, \ldots, V_i^{(t-m+1)}\right) (mod\ 2)$$

$$= f_{\omega_{m-1}}\left(V_i^{(t)}\right) + \cdots + f_{\omega_1}\left(V_i^{(t-m+2)}\right) + s_i^{(t-m+1)} (mod\ 2) \quad (7)$$

where $0 \leq i \leq n-1$, and $\omega_1, \omega_2, \ldots, \omega_m \in \{0,1,\ldots,2^{2r+1}-1\}$. Proof of this proposition can be found in Fredkin [12].

C. Secret sharing using LMCAs

Using linear memory cellular automata, a secret sharing scheme has been proposed initially by [5]. As the scheme is a (t,n)-threshold one, the secret is considered as an initial configuration of a t-order LMCA, while remaining (t-1) configurations are randomly generated. By evolving the constructed LMCA, n consecutive configurations are created to define the n different shares distributed among the n participants. When secret's reconstruction is desired, the LMCA is running backward starting from any set of t consecutive shares, permitting a perfect reconstruction of the initially shared secret in a linear time with respect to the size of secret.

Suppose that the secret is defined by the initial configuration C^0. Remaining t-1 configurations $C^1, C^2, \ldots C^{t-1}$ that are necessary to run the designed t-order LMCA are generated randomly (if the scheme share a unique secret) or are defined by the remaining secrets (if the scheme handle multiple secret). The set $\{C^0, C^1, \ldots, C^{t-1}\}$ is then used to build an (n+t-1)-th order evolution of the LMCA to obtain a set of n consecutive configuration $\{C^t, C^{t+1}, \ldots, C^{n+t-1}\}$ distributed among the n participants. When required, any set of t consecutive configurations $\{C^{t+\alpha}, C^{t+\alpha+1}, \ldots, C^{2t+\alpha-1}\}$ is used to define a set $\{\tilde{C}^{(0)}, \tilde{C}^{(1)}, \ldots, \tilde{C}^{(t-1)}\}$ as the initial configuration to run the inverse LMCA backward for $\alpha+t$ iterations and recover the secret [5]. The set of t-1 transition rules used for the LMCA evolution $\omega_1, \omega_2, \ldots, \omega_{t-1}$ is generated initially by the dealer and made public without affecting the security of the scheme.

Even if the LMCA's based secret sharing provides linear time complexity sharing and reconstruction with respect to existing schemes, it does not define a robust (t,n)-threshold mechanism since not all subsets of t participants can recover the secret, but only those having consecutive shares (configurations) (as explained in the introduction section). Several enhancements have been proposed later in [6,7,8] and [9], but they all targeted the enhancements of other sharing aspects such as multi-secrets support, share's verifiability, t-consistence and traceability. The main robustness's drawback of LMCA's based sharing scheme has not been addressed yet.

In the present work, we present for the first time a solution to the robustness problem of LMCAs based sharing schemes. The basic idea is simple: instead of defining the share of each participant by only one configuration, a subset of configuration is attributed to each participant such that the union set of t subsets from t different participants contain a unique sequence of t consecutive configurations, while the union of any less number of subsets do not permit to obtain such sequence. Details and proofs of the proposed scheme are presented in the following sections.

III. THE PROPOSED SCHEME

In this section, we present a new secret sharing scheme based on one dimensional LMCA. The proposed (t,n)-threshold scheme ($2<t<n-1$) is robust, such that any subset of at less t participant can fully recover the shared secret when pooling their shares together. As usual, three main phases are necessary for the secret sharing scheme: (1) the setup phase, during which the dealer generates parameters of the scheme and defines the t-order LMCA; (2) the sharing phase, when the dealer creates the n different shares using the defined LMCA and the secret' data, and finally (3) the reconstruction phase

permitting to recover the secret from any set of t different shares. Note that in the proposed work, we used LMCA have a radius r=3, so the transition rules belong to the set {0,...,127}.

A. Construction of the assignment matrix

As mentioned above, the dealer need an assignment matrix in order to distributed the LMCA's configuration among the n participants. The assignment matrix is used to decide which set of configurations should be given to each user, such that each one obtains configurations having as indexes the elements of the participant's corresponding column.

The assignment matrix noted A has n column (corresponding to n participant) and C_{n-2}^{t-2} rows of integer elements. It is built in a way that satisfy the following conditions:

(1) Combining t-1 columns does not permits to construct a sequence of t consecutive numbers;
(2) Combining any t columns permits to construct a sequence of t consecutive numbers;

Theoretically, building such matrix is a combinatorial hard problem. Hence, we have heuristically developed an algorithm that permits such construction and produces a matrix respecting the two conditions mentioned above. In addition, no restriction is imposed on the upper limit of the matrix's elements, and duplicated values are allowed. Each column i of the matrix correspond to a participant P_i for $1 \leq i \leq n$. The assignment matrix A is constructed according to the following steps :

1. Initially , the matrix A having n columns and C_{n-2}^{t-2} rows is initialized with zeros; let CS be the set of all possible C_{n-2}^{t-2} combinations of (t-2)-uplet of indexes from the set {2,...,n-1} that is previously constructed, and let's suppose that an integer value *id* defines the smallest value permitted for matrix's elements (can trivially be equal to 1). The matrix is constructed row by row starting by the first one.
2. The first elements of the row receives the value of *id*; then the first combination from CS is picked;
3. Elements of the currently picked combination are used as indexes to fill up the current row of A: if the combination is defined by $<i_1,i_2,\ldots,i_{t-2}>$ then we assign the value id+k to the each element $A[i_k]$;
4. Each one of the matrix's elements at the current row having index higher than i_{t-2} (the higher index value of the combination) receives the value id+t-1;
5. A last verification step consists in looking over the elements of the current row from 2 to n-1 and testing: if the element is still equal to 0, then the element receives the value of the prior one;
6. If all the C_{n-2}^{t-2} rows have been filled, then the algorithm ends. Otherwise, it increments the index of the current row, updates the value of id by id:=id+t+1 and returns to step 2.

A pseudo-algorithmic description of the matrix generation procedure is given in the following. We consider CS[0.. C_{n-2}^{t-2}-1] to be an array representing all possible t-2 combinations on the values {2,...,n-1} such that each element CS[k] is a possible combination $<i_1^k,i_2^k,\ldots,i_{t-2}^k>$.

Input : t,n :integers; CS ; id :integer (value of the smallest permitted element of the matrix);
Output :The assignments matrix A having n columns and C_{n-2}^{t-2} rows;
For i:=1 **to** C_{n-2}^{t-2} **do**
  For j:=1 to n **do** {A[i,j]:=0;}
For i:=1 **to** C_{n-2}^{t-2} **do** {A[i,1]:=id;
  For k:=1 **to** t-2 **do** {A[i,CS[i][k]]:=id+k;}
  For k:= CS[i][t-2]+1 **to** n **do** {A[i][k]:=id+t-1;}
  For k:=2 **to** n-1 **do** {if A[i][k]=0 then A[i][k]:=A[i][k-1];}
  id:=id+t+1; };

The proposed algorithm is designed to ensure the two conditions mentioned above. Incrementing the *id* value with a supplementary value 1 is performed from row to row in order to avoid that two consecutive rows contain a sequence of t consecutive elements by introducing a sequence gap. Let's illustrate in the following an example of the matrix construction using values n=6 and t=4:

Firstly, using the parameters values, it is clear that the matrix has 6 columns and $C_4^2 = 6$ rows. The set CS of C_4^2 possible index's combination form the set {2,3,4,5} is equal to {(2,3),(2,4),(2,5),(3,4),(3,5),(4,5)}. By applying the proposed algorithm, the following matrix is obtained :

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 4 & 4 \\ 6 & 7 & 7 & 8 & 9 & 9 \\ 11 & 12 & 12 & 12 & 13 & 14 \\ 16 & 16 & 17 & 18 & 19 & 19 \\ 21 & 21 & 22 & 22 & 23 & 24 \\ 26 & 26 & 26 & 27 & 28 & 29 \end{bmatrix} \quad (8)$$

We can easily verify that combining any 4 columns of the matrix permits to obtain a sequence of four consecutive numbers. For example combining the columns 1,2,3 and 4 gives the sequence of consecutive values {1,2,3,4}, while combining the columns 2,3,5 and 6 gives the sequence of consecutive values {21,22,23,24}. In contrast, combining any three columns do not permits to get any sequence of four consecutive numbers.
Another example can be illustrated for n=5 and t=3. Here, the set CS is simply the set of possible values from {2,3,4} having length one, that is trivially equal to the same set {2,3,4}. Applying the algorithm gives the following assignment matrix having three rows and five columns:

$$A = \begin{bmatrix} 1 & 2 & 3 & 3 & 3 \\ 5 & 5 & 6 & 7 & 7 \\ 9 & 9 & 9 & 10 & 11 \end{bmatrix} \quad (9)$$

The same properties exist in this matrix when grouping element of any three columns leads always to a sequence of three consecutive numbers.

Let's show in the following that for any values of the parameters n and t, the produced assignment matrix always verify conditions (1) and (2):

Lemma III.1. For any values of n and t, the assignment matrix constructed using the proposed algorithm has the following properties :

1. The union set of elements for any t columns contains always a sequence of t consecutive numbers;
2. The union set of any t-1 or less columns do not contain any sequence of t consecutive numbers.

Proof.

We start by proofing the first assumption. Suppose having a combination $B=<i_1,i_2,\ldots,i_t>$ of t different columns from the matrix A ($i_k \neq i_j \ \forall 1 \leq k,j \leq t$). We also suppose that the combination's elements are sorted in ascending order ($i_1<i_2<\ldots<i_t$).

Let's consider $B'=<i_2,i_3,\ldots,i_{t-1}>$ the sub combination of B restricted on the t-2 elements by deleting the first and the last one, and let Ord(B') be the order of B' within the set CS.

It is clear from the algorithm that since $B' \in CS$, t-2 following consecutive numbers:

$$\{id+Ord(B')*t+1,\ id+Ord(B')*t+2,\ id+Ord(B')*t+3,\ldots\ldots,\ id+Ord(B')*t+t-2\ \} \quad (10)$$

will be affected to the indexes of B' during the first loop of iteration number Ord(B') (the iteration using B' as combination base). In addition, the second loop will assign the value id+Ord(B')*t+t-1 to remaining indexes higher than i_{t-1}, and since $i_t>i_{t-1}$, the value of the current row at the index i_t will receive the value id+Ord(B')*t+t-1. As a result we get t-1 consecutive values {id+Ord(B')*t+1, id+Ord(B')*t+2, id+Ord(B')*t+3,……, id+Ord(B')*t+t-2 , id+Ord(B')*t+t-1} for the indexes $<i_2,i_2,\ldots,i_t>$.

Now we consider the first index i_1. In the same row (corresponding to iteration number Ord(B')), unless i_1 is equal to one ($i_1=1$) and in this case the value of the row at position i_1 is certainly equal to id+Ord(B')*t (since it is the initialization value of each row). Otherwise, if $i_1>1$, and since by assumption $i_1<i_2$, no value will be assigned to the value of the current row at position i_1 during the second loop (its value remain equal 0). During the third loop, this value will take the one of prior position than is certainly equal to the value of the first column in the current row id+Ord(B')*t. As a consequence, in all cases, a sequence {id+Ord(B')*t, id+Ord(B')*t+1, id+Ord(B')*t+2,……, id+Ord(B')*t+t-1} can be constructed for the combination B and it is clearly a sequence of t consecutive numbers.

Proofing the second assumption is easier : suppose we have a combination of t-1 columns $< i_1,i_2,\ldots,i_{t-1}>$ from A. It is clear that since a gap is introduced between each two consecutive rows (i.e. the value of the first element of the new row is always incremented with one according to the last value of the prior row), sequence of t consecutive numbers can only be obtained on a single row. Now since we have only t-1 different indexes for the t-1 columns, we can never collect a sequence of t numbers from the same row. Hence no t consecutive sequence of t numbers can be collected.

Based on the assignment matrix constructed using the proposed algorithm for any two values t and n verifying $2<t<n-1$, we propose an LMCA-based secret sharing scheme in the next section. A subset of configurations is assigned to each participant using the indexes of his corresponding column of the matrix. Note that the algorithm returns a new value of the parameter *id* that represents the highest value of the matrix's elements, and determines the number of LMCA's configurations to be constructed.

B. The setup phase

During the setup phase, the dealer responsible for the shares generation should firstly define the parameters of the scheme. The following steps are performed during this phase:

1. The dealer generates t-1 random integer from the set {0,...,127} in order to define the t-1 transition rules ω_1, ω_2,....., ω_{t-1} defining the t-1 local transition functions $f_{\omega 1}$, $f_{\omega 2}$,....., $f_{\omega t-1}$.
2. The dealer divides the secret S on |S| byte into t parts $PS_1,PS_2,\ldots,PS_t$ having sizes equal to [|S|/t], each part defines a configuration of LMCA as follows:
$$C^0 = PS_1, C^1 = PS_2, \ldots, C^{t-1} = PS_t \quad (11)$$
If the value of |S| does not divide t, a padding scheme is used to complete cells in C^{t-1}.
3. The dealer generates a public random integer number $\alpha>n+1$. This parameter is used to introduce a sufficient diffusion and confusion between the LMCA configurations and hence produces more randomness in the resulting shares.
4. The dealer constructs an assignment matrix A using the parameters t, n and id=α. The returned value of *id* (the highest element of the matrix) is assigned to a new parameter β used in the following steps. It is clear that α is the smallest element of the matrix A, while β is the highest one.
5. The dealer computes the evolution of (β)-th order of the LMCA, starting from the initial configurations $\{C^0,C^1,.....,C^{t-1}\}$:
$$\{C^0, C^1, \ldots, C^{t-1}, C^t, \ldots, C^n, \ldots, C^\alpha, \ldots C^{\beta-1}, C^\beta\} \quad (12)$$

C. The sharing phase

During the sharing phase, the dealer uses the $\beta-\alpha+1$ $\{C^\alpha,C^{\alpha+1},....,C^\beta\}$ last configurations generated during the setup phase to build the shares distributed among the n participants $P_1,P_2,\ldots,P_n$ like the following:

1. For each participant P_i, the dealer assigns a subset Sb_i of configurations from the set $\{C^\alpha,C^{\alpha+1},....,C^\beta\}$ having as indexes the values of the i^{th} column from the assignments matrix A:
$$Sb_i = \{C^{A[1,i]}, C^{A[2,i]}, \ldots\ldots, C^{A[C_{n-2}^{t-2}-1,i]}, C^{A[C_{n-2}^{t-2},i]}\ \} \quad (13)$$
Each participant will receive exactly C_{n-2}^{t-2} different configurations.
2. For each participants P_i, the configurations of its corresponding set Sb_i are concatenated to form the

corresponding share. The shares are finally distributed among the participants among a secure channel.

Note that the number of all distributed configurations among all participants is equal to the number of elements of the matrix (equal to n* C_{n-2}^{t-2}) that is lower than the number of total generated configurations equal to β-α+1. This difference is due to two factors: the gaps introduced between consecutive rows during matrix creation, and the repetition procedure introduced to satisfy combinations belonging to the same row.

D. The reconstruction phase

During the construction phase, the combiner reconstructs the secret from at least t different shares from t distinct participants $\{P_{i1},P_{i2},....,P_{it}\}$ according to the following steps :

1. For each participant P_{ik}, the corresponding share is transformed into a set of configurations according to the values that correspond to elements of the $(i_k)^{th}$ column of A. The share of participant P_{ik} defines the following set of C_{n-2}^{t-2} configurations:

$$S_{i_k} = \{C^{A[1,i_k]}, C^{A[2,i_k]}, \dots\dots, C^{A[C_{n-2}^{t-2},i_k]}\} \quad (14)$$

2. The combiner constructs the union set S_U of all the t participant's subsets of configurations like the following :

$$S_U = \cup_{i=i_1 \dots i_t} S_i \quad (15)$$

3. According to the Lemma III.1, there exists always a sequence of t consecutive numbers in each union of t different columns from the assignment matrix A. The combiner determines the set of consecutive indexes from the matrix noted Seq=$\{v_1,v_2,...,v_t\}$. Using Seq, The combiner construct the following sequence :

$$\{C^{v_1}, C^{v_2}, \dots\dots, C^{v_t}\} \quad (16)$$

that is a set of consecutive configurations in the set S_U.

4. Since the set Seq is ordered, v_1 is the smallest integer of the sequence, the combiner than computes the (v_1)-th order evolution of the inverse LMCA constructed during the sharing step (using the same transition rules ω_i's). The inverse LMCA is run using the following configurations:

$$\{\check{C}^0 = C^{v_t}, \check{C}^1 = C^{v_{t-1}}, \dots\dots, \check{C}^{t-1} = C^{v_1}\} \quad (17)$$

and the inverse LMCA is run for v_1 iteration to obtain the t configurations representing the initially shared secret.

According to the proposition II.1, the initial configurations can always be recovered. Hence the secret can be reconstructed by concatenating recovered configurations to build the initially shared secret.

IV. SECURITY ANALYSIS OF THE PROPOSED SCHEME

We show in the following that the proposed scheme verify robustness and secrecy. Robustness of the scheme means that only a set of at less t participants can recover the shared secret, while t-1 or less number of them cannot reveal any information about it. Secrecy of the scheme means that individual shares indistinguishable from randomly generated data. Robustness is shown in the following using theoretic assumptions, while secrecy of the scheme is experimentally demonstrated using randomness's measurements of produced shares.

A. Robustness of the scheme

Robustness of the proposed scheme relays on the following facts :

a) A t-order LMCA's mechanism can reconstruct initial t configurations from any subset of t consecutive ones using the corresponding inverse LMCA with defined linear transitions rules;
b) Reconstruction of initial t configurations of an LMCA is impossible when using any t-1 or less number of configurations;
c) The assignment matrix constructed using the proposed corresponding algorithm ensures that union of the elements of any t columns permits to build a sequence of t consecutive values, while the union of any t-1 or less columns does not permits it.

The first fact is established by the proof of the proposition II.1 form section 2. Is has been shown in [12] that LMCA constructed according to equations (6) and (7) are always reversible, while their inversion using t-1 or less configurations is a computationally hard problem. Hence the facts (a) and (b) are proofed.

The fact (c) can be shown using Lemma III.1 from section 3. We have established that the proposed matrix construction algorithm produces always valid matrices. Their validity is considered with respect to the two constraints of the lemma, that correspond exactly to the requirement of the fact (c). By combining the verifiability of the three facts (a),(b) and (c), we conclude that the proposed scheme is robust.

B. Illustrative experiments and secrecy analysis

In order to illustrate the steps of the proposed scheme and to experimentally show the secrecy of the produced shares, we develop in the following an illustrative example of secret sharing using the proposed approach. We choose to apply the scheme to share the digital (512x512) color image illustrated in figure 1. Digital images sharing is one of the most active research area related to secret sharing due to their specific characteristics such as redundancy, bulky data capacity and high correlation across blocks of pixels.

Let's build a (3,7)-threshold secret sharing scheme (i.e t=3 and n=7). We firstly define the parameters of the scheme by generating the transition rules. Two values ω_1 and ω_2 are chosen from {0,…,127}: let's assume ω_1=45 and ω_2=138. We also assign a random value to the parameter α (to ensure a sufficient diffusion and randomization of the shares), so let's choose α=25.

The assignment matrix A having 7 columns and $C_{7-2}^{3-2} = C_5^1 = 5$ rows is generated using the proposed algorithm: the set CS of t-2=1 possible combinations on the set {2,3,4,5,6} is

trivially CS={2,3,4,5,6} (since the length of combinations is equal to one). We consequently obtain the following matrix:

$$A = \begin{matrix} P_1 & P_2 & P_3 & P_4 & P_5 & P_6 & P_7 \end{matrix} \begin{bmatrix} 25 & 26 & 27 & 27 & 27 & 27 & 27 \\ 29 & 29 & 30 & 31 & 31 & 31 & 31 \\ 33 & 33 & 33 & 34 & 35 & 35 & 35 \\ 37 & 37 & 37 & 37 & 38 & 39 & 39 \\ 41 & 41 & 41 & 41 & 41 & 42 & 43 \end{bmatrix} \quad (18)$$

when each column correspond to a given participant. As output of the algorithm, we obtain the value of the parameter β=43 (the highest value of the matrix's elements).

The secret image is then decomposed into three blocks defining the three configurations of the LMCA C^0,C^1 and C^2. Since the image contain 512x512=262144 pixel each one on 3 bytes (the image is a 24bit color one), the size of each configuration is 262144*3/3=262144 byte. We finally construct the 3-order LMCA and evolve it using the configurations C^0,C^1 and C^2 for β = 42 iteration using the rules ω_1 and ω_2 . We finally obtain a set of 42 consecutive configurations $\{C^0,C^1,\ldots,C^{42}\}$.

Fig. 1. The secret image (512x512) used to illustrate the proposed scheme.

Using the assignment matrix A, obtained configurations are assigned according to equation (13) in order to construct the different participant's shares S_i ($1 \le i \le 7$) like the following :

$$\begin{aligned} S_1 &= \{C^{25}, C^{29}, C^{33}, C^{37}, C^{41}\} \\ S_2 &= \{C^{26}, C^{29}, C^{33}, C^{37}, C^{41}\} \\ S_3 &= \{C^{27}, C^{30}, C^{33}, C^{37}, C^{41}\} \\ S_4 &= \{C^{27}, C^{31}, C^{34}, C^{37}, C^{41}\} \\ S_5 &= \{C^{27}, C^{31}, C^{35}, C^{38}, C^{41}\} \\ S_6 &= \{C^{27}, C^{31}, C^{35}, C^{39}, C^{42}\} \\ S_7 &= \{C^{27}, C^{31}, C^{35}, C^{39}, C^{43}\} \end{aligned} \quad (19)$$

For each participant, the configurations are assembled and concatenated to form the final share. Since each configuration is on 262144 byte, the size of the share is equal to 262144*5=1310720 byte, and can then be represented as a color image of 661x661 pixels (using a padding scheme to complete remaining 14 pixels). Figure 2 illustrates the obtained seven shares when sharing the image of figure 1 using the proposed approach with respect to a (3,7)-threshold sharing scheme.

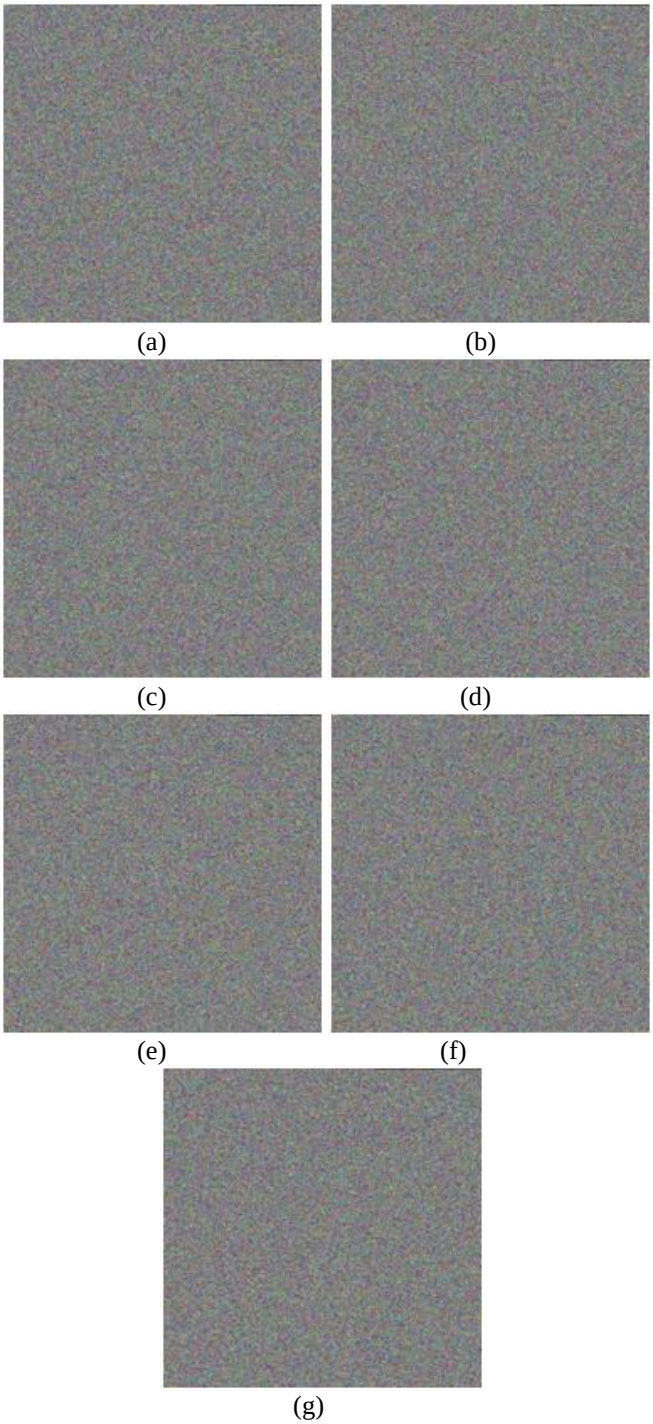

Fig 2. Obtained shares (661x661) for the secret image of figure (a),(b),(c),(d),(e),(f) and (g) are respectively the shares of P_1,P_2,P_3,P_4,P_5,P_6 and P_7.

It is clear from figure 2 that obtained shares have random aspect and hence do not reveal any useful information about the originally shared secret. In order to approve this fact, several statistical experiments have been performed on the shares in order to show that they are indistinguishable from random noise. The shares have been analyzed using both

Diehard and ENT statistical Tests batteries [13], when obtained results are averaged and reported in Tables 1 and 2. We can easily note from the results that the shares content has very good statistical properties since it pass majority of applied tests. Such results imply that the shares are indistinguishable from random images, and as a result, the sharing scheme ensures secrecy.

Now suppose that a given subset of participants colludes to recover the secret image. Since the scheme is (3,7)-threshold one, at least the shares of three different participant are required. Let's suppose that P_2,P_4 and P_7 are those participants, so the combiner uses the shares illustrated by figures 2.(b), 2.(d) and 2.(g). Each share is decomposed into its composing configurations, to get the three configuration's sets S_2,S_4 and S_7 defined in equation (19). The combiner then uses the public assignment matrix and construct the union set $S_U=S_2 \cup S_4 \cup S_7$. It is clear that after ordering, the set S_U is defined by:

$$S_U = \{C^{26}, C^{27}, C^{29}, C^{31}, \mathbf{C^{33}}, \mathbf{C^{34}}, \mathbf{C^{35}}, C^{37}, C^{39}, C^{41}, C^{43}\} \quad (20)$$

The set SU contains the sequence of three consecutive configurations C^{33},C^{34} and C^{35}. According to the reconstruction scheme, and since 33 is the smallest configuration's index, the combiner run the inverse LMCA (using the same public transition rules ω_1 and ω_2) for 33 iteration starting from the initial configurations C^{35},C^{34} and C^{33} (in inverse order) to reconstruct the configurations C^2,C^1 and C^0 that define the initially shared secret image. When applied on the shares of figures 2.(b), 2.(d) and 2.(g), the reconstruction scheme recover exactly the same image of figure 1 since the proposed approach is lossless.

C. Share's size and efficacy analysis

Since the proposed scheme assigns multiple configurations to the same participant, the size of the share is always higher than the size of the secret. According to the sharing scheme, if we consider that the size of the secret is given by |S|, the threshold and the number of participants are t and n respectively, then the estimated size of each share is given by :

$$|\text{Share}| = \frac{C_{n-2}^{t-2} * |S|}{t} \quad (21)$$

TABLE I
AVERAGED RESULTS OF THE DIEHARD TESTS BATTERY APPLIED ON THE PRODUCED SHARES

Test Name	Averaged P-value	Interpret-ation
BIRTHDAY SPACINGS	0 .910134	Pass
OVERLAPPING PERMUTATION	0 .982213	Pass
RANK TEST 31x31	0 .692522	Pass
RANK TEST 32x32	0 .562338	Pass
MONKEY DNA	0 .781003	Pass
COUNT-THE-1's TEST	0 .578523	Pass
PARKING LOT	0 .359782	Pass
MINIMUM DISTANCE	0 .294752	Pass
RUNDOM SPHERE	0.568922	Pass
The SQEEZE test	0 .847215	Pass
OVERLAPPING SUMS	0.684211	Pass
The RUNS -up test, down test-	0 .920325 0.531017	Pass
CRAPS -no of wins ,thrwos/game-	0 .875423 0.72087	Pass
RANK TEST	0.835214	Pass
MONKEY 20 BITS PER WORD	0.870254	Pass
MONKEY OPSO, OQSO	0.915472	Pass

TABLE II
AVERAGED RESULTS OF THE ENT TESTS BATTERY APPLIED ON THE PRODUCED SHARES

Test	Value	Norm
Entropy	7.999925	Max=8.0
Arithmetic mean	127.209	127.5=random
Monte Carlo	3.1485967(error=0.24%)	π value
Serial correlation coefficient	0.00059	0.0
Optimum compression	0.00001	0.0

Depending of the threshold and the participants number, the size of the share may differ and be extremely high for some combinations of t and n. We have experimentally studied the evolution of the share's size with respect to the parameters of the scheme in order to define its efficiency conditions. Figure 3 shows the evolution of the scaling factor between the secret's size and the share's size with respect to t and n. It is clear that the scheme is effective in a sufficiently large region of the space and becomes impractical in the mentioned region. According to these results, we conclude that using the proposed scheme is conditioned by the relation between the values of t and n, and even if the scheme ensures a full robustness, it is largely non-ideal when t is in the neighborhood of n/2. So in general, LMCS-based sharing scheme can ensures either a full robustness without ideality or ensures ideality without robustness like provided with existing LMCA-based schemes.

For further illustrations of the proposed scheme properties and capabilities, a comparative study is given in table 3. The scheme is compared to some recent existing threshold sharing schemes with respect to several performances parameters. It is clear that with respect to CA-based schemes, the proposed one is the only that ensures robustness, while ideality is only ensured if t and n do not belongs to the impractical region. Linear computational complexity is another advantage of the scheme that is not ensured by almost all non-LMCA schemes having at least a polynomial complexity.

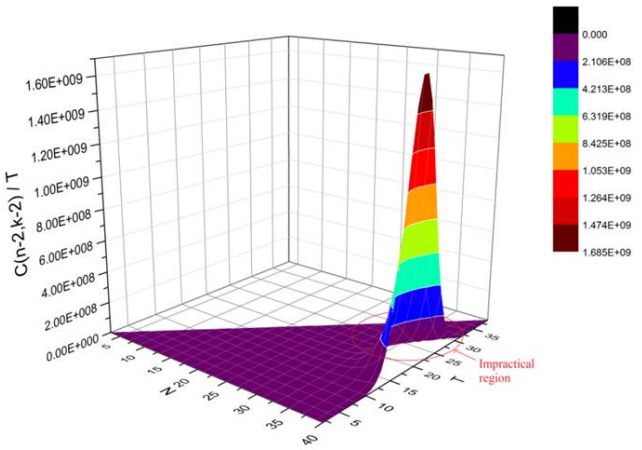

Fig. 3. Estimated scaling factor between the secret's size and the share's size with respect to possible values of t and n.

TABLE III
COMPARISON AMONG EXISTING (T,N)-THRESHOLD SECRET SHARING SCHEMES

(t,n) Sharing Scheme	Sharing complexity	Reconstruction complexity	Lossless	Scalability for large data	Robustness
Thien et al.[14]	$O(\|S\|*nlog^2n)$	$O(\|S\|*tlog^2t)$	**Yes**	**No**	**Yes**
Yang et al.[15]	$O(\|S\|*nlog^2n)$	$O(\|S\|*tlog^2t)$	No	**No**	No
Lin et al. [16]	$O(\|S\|*nlog^2n)$	$O(\|S\|*tlog^2t)$	**Yes**	**No**	**Yes**
Hadian et al.[17]	$O(\|S\|*nlog^2n)$	$O(\|S\|*tlog^2t)$	**Yes**	**No**	**Yes**
Eslami et al.[8]	$O(\|S\|*n)$	$O(\|S\|*t)$	**Yes**	**Yes**	No
Wu et al. (2012)[9]	$O(\|S\|*n)$	$O(\|S\|*t)$	No	**Yes**	No
Wu et al. (2013) [18]	$O(\|S\|*n)$	$O(\|S\|*t)$	No	**Yes**	No
Proposed	$O(\|S\|*n)$	$O(\|S\|*t)$	**Yes**	**Yes**	**Yes**

V. CONCLUSIONS

The present paper aims to solve the robustness problem of cellular automata based threshold secret sharing schemes. While all existing CA-based schemes provide linear computational complexity for sharing and reconstructing secrets, they all fail to ensure a robust sharing mechanism since only the shares defined by consecutive shares can be used to recover the secret. Hence, a very large number of authorized participant's subsets are unable to reconstruct the initially shared data. To solve the problem, we proposed to assign multiple configurations to each user in order to permit to each subset of at least t participants to get an access to the full secret reconstruction. A specific assignment matrix is heuristically generated using a proposed algorithm, and used to define a new sharing/reconstruction mechanisms. The proposed scheme has been shown to be robust, and has undergo several experimental benchmarking to illustrate the secrecy of its produced shares. Security of the proposed scheme is established, and conditions of its ideality are illustrated with respect to the values of the parameters t and n.

With respect to exiting non-CA secret sharing schemes, the proposed one ensures a linear sharing/reconstruction complexities that leads to faster and scalable performances, while it provides a full robustness property compared to exiting CA-based schemes. However, an expensive cost of large shares size with respect to the secret size can arias for some values of t and n, inducing a non-ideality of the scheme.

We conclude according to our studies that cellular automaton cannot provide an ideal solution to the sharing problem without losing robustness, and we are working as future works on optimization of the assignment matrix to find a solution with less row's number that can further optimize the size of the produced shares.

REFERENCES

[1] Shamir, A. (1979). How to share a secret. Communications of the ACM, 22(11), 612-613.

[2] Blakley, G. R. (1899, December). Safeguarding cryptographic keys. In Managing Requirements Knowledge, International Workshop on (pp. 313-313). IEEE Computer Society.

[3] Asmuth, C., & Bloom, J. (1983). A modular approach to key safeguarding. IEEE Transactions on Information Theory, 30(2), 208-210.

[4] Mignotte, M. (1983). How to share a secret. In Cryptography (pp. 371-375). Springer Berlin Heidelberg.

[5] Martín del Rey, A., Mateus, J. P., & Sánchez, G. R. (2005). A secret sharing scheme based on cellular automata. Applied mathematics and computation, 170(2), 1356-1364.

[6] Marañón, G. Á., Encinas, L. H., & Del Rey, A. M. (2005). A new secret sharing scheme for images based on additive 2-dimensional cellular automata. In Pattern Recognition and Image Analysis (pp. 411-418). Springer Berlin Heidelberg.

[7] Z. Eslami, S.H. Razzaghi, J. Zarepour Ahmadabadi, Secret image sharing based on cellular automata and steganography, Pattern Recognition 43 (2010), 397-404.

[8] Z. Eslami and J. Zarepour Ahmadabadi, A verifiable multi-secret sharing scheme based on cellular automata, Inform. Sciences 180 (2010), 2889-2894.

[9] X. Wu, D. Ou, Q. Liang and W. Sun, A user-friendly secret image sharing scheme with reversible steganography based on cellular automata, J. Systems Software 85 (2012), 1852-1863.

[10] T. Toffolli, and N. Margolus, "Invertible cellular automata: A review", Physica D, Vol 45, pp 229-253, 1990.

[11] S. Wolfram, "A New Kind of Science", Wolfram Media, pp. 437–440, ISBN:1-57955-008-8, World Scientific, 2002.

[12] Fredkin, Edward, An informational process based on reversible universal cellular automata, Physica D 45(1990), 254-270.

[13] J. Soto, "Statistical Testing of Random Number Generators," Proceedings of the 22nd National Information Systems Security Conference, Crystal City, Virginia, October 1999.

[14] THIEN, Chih-Ching et LIN, Ja-Chen. Secret image sharing. *Computers & Graphics*, 2002, vol. 26, no 5, p. 765-770.

[15] YANG, Ching-Nung et CHU, Yu-Ying. A general (k, n) scalable secret image sharing scheme with the smooth scalability. *Journal of Systems and Software*, 2011, vol. 84, no 10, p. 1726-1733.

[16] LIN, Pei-Yu et CHAN, Chi-Shiang. Invertible secret image sharing with steganography. *Pattern Recognition Letters*, 2010, vol. 31, no 13, p. 1887-1893.

[17] Dehkordi, Massoud Hadian et Mashhadi, Samaneh. New efficient and practical verifiable multi-secret sharing schemes. *Information Sciences*, 2008, vol. 178, no 9, p. 2262-2274.

[18] Wu, X., & Sun, W. (2013). Secret image sharing scheme with authentication and remedy abilities based on cellular automata and discrete wavelet transform. *Journal of Systems and Software*, *86*(4), 1068-1088.

Djihed Anani. Is a Phd student at the computer science department of the UDL-University in Algeria. She get his master's degree in 2013 in the same university. She is working on digital images secret sharing, visual cryptography and information security.

Kamel Mohamed Faraoun. received his master's degree in computer science at the computer science department of Djilali Liabbes University- Sidi-Bel-abbes – Algeria in 2002, his Ph.D degree in computer science, in 2006, and his HDR degree in computer science and intelligent systems, in 2009 From UDL-University. His current research areas include computer security systems; cryptography; genetic algorithms; cellular automata; evolutionary programming and information theory.