14

An En/Decryption Machine Based on Statistical Physics

Annie Perez, Céline Huynh Van Thieng,
Samuel Charbouillot and Hassen Aziza
Aix-Marseille Univ., IM2NP ; CNRS, IM2NP (UMR 6242)
IMT, Technopôle de Château-Gombert, Marseille Cedex 20
France

1. Introduction

Internet-based communications, multimedia systems, telemedicine or military communications, need high-speed data encryption. Thus, in various fields, high-speed cryptosystems are necessary for the large data size transfers and real-time constraints. A secret key stream cipher is suitable for these high-speed encryption systems.

The stream cipher generates long unpredictable key sequences. These pseudo-random key bits, or Cipher Keys, are then bitwise XORed with the data to encrypt/decrypt. Since many processes in the nature include the randomness in themselves, the main idea of this study is to use this natural randomness to generate Cipher Keys.

Physical systems containing randomness appear to follow no definite rules, and to be governed merely by probabilities. Moreover, there are systems that can also generate apparent randomness internally, without an external random input. For instance, a cellular automaton (Bagnoli & Francescato, 1990; Sarkar, 2000; Vichniac, 1984; Wolfram, 1983) evolving from a simple initial state can produce a pattern so complicated that many features of it seem random.

The physical system considered in this chapter is a ferromagnetic material. As the temperature increases, thermal oscillation, or entropy, competes with the ferromagnetic tendency for dipoles to align. At high temperature the magnetization is destroyed and the dipoles are disordered.

In order to simulate this system at high temperature to obtain these predicted disordered dipoles features, we used the well-known two-dimensional Ising model (Ising, 1925; Onsager, 1944) where a spin (encoded on one bit) represents a dipole. Space and time are discrete in this model. The evolution law of the spin lattice (or bit array) is defined by local rules between neighbour spins.

A mapping between the spin lattice and a cellular automaton cell array seems obvious. What are the more suitable local rules for a fast and few resources consuming secret key cryptosystem? We shall try to answer this question. In the framework of secret key cryptography (Chen & Lai, 2007; Sathyanarayana et al., 2011; Seredynski et al., 2004), we shall propose an Ising Spin Machine (ISM) as a feasibility model for data stream encryption.

ISM is synchronous and needs an initialization phase through a parameter set. Then, at each time step, ISM generates a pseudo-random array of bits, shifts the data flow to encrypt from south towards north, and combines (logic XOR operation) the data with the random bits. The decryption process is identical to the encryption one.

ISM can be used to secure communication over an unsecure channel. If Alice wants to send to Bob a secret data flow which may be trapped by an adversary, she can encrypt this data flow using an ISM. Alice initializes her ISM and communicates the initialization parameter set to Bob (through a secure channel). This set of parameters builds the secret key. Then Bob initializes his own ISM and waits. Alice introduces the data flow to encrypt into her ISM which generates the encrypted flow that is sent to Bob. When Bob receives the first encrypted word, he enables his ISM for a real-time data flow decryption process. The estimated throughput of this enc/decryption process is 2 Gbps.

The rest of this chapter is organized as follows: Section 2 presents the Ising model. Then two algorithms to simulate the 2D-Ising model are described in Section 3. Next, section 4 proposes a parallel implementation of the Reservoir algorithm to generate Cipher Keys. Section 5 is dedicated to the architecture and performances of the Ising Spin Machine, and an image encryption/decryption application example is proposed. Finally, section 6 concludes the chapter.

2. A model for nature randomness

An example of randomness in the nature can be found in an iron bar. Consider this iron bar in a strong magnetic field, H, parallel to its axis. In these conditions the bar is almost completely magnetized. Its magnetization is M_1. Now decrease H to zero: the magnetization will decrease but not to zero. Rather, at zero field a spontaneous magnetization M_0 will remain. Now suppose that the temperature T is increased slightly. It is found that M_0 decreases. Finally, if T is increased to a critical value T_C (the Curie point), M_0 vanishes. The spontaneous magnetization curve $M_0(T)$ is given in Figure 1.

Spins models were invented as simple statistical physics models of ferromagnetism. In most cases they exhibit the cooperative behaviour found in phase transitions. The well known Ising model describes the phase transition occurring at the temperature T_C, between a low temperature phase (called ferromagnetic phase) with a spontaneous magnetization M_0 and a high temperature phase (called paramagnetic phase) where the magnetization M_0 vanishes.

The Ising model is the most famous model in Statistical Physics (Onsager, 1944). The aim of statistical physics is to predict the relations between the observable macroscopic properties of the system, given only the knowledge of the microscopic forces between its components. In the Ising model the magnet is made up of molecules that are constrained to lie on the sites of a regular lattice. Suppose that there are N such sites and molecules, labeled i = 1, 2, …, N.

Let us consider a molecule as a microscopic magnet which either points along some preferred axis, or points in exactly the opposite direction. So molecule *i* has two possible states, which can be described by a spin variable S_i with a value "up" when it is parallel to the axis, and "down" when it is anti-parallel to the axis. Thus there are 2^N configurations of the lattice, called "spin configurations" or "micro-states" of the system. The spin-spin interaction is described by the coupling constant J. Figure 1 gives two spin configurations:

one at the transition temperature T_C (a), and the other one above T_C (b). Spins are organized in clusters at T_C and begin to be disordered at 1.28 T_C. More details can be found in (Perez et al, 1995).

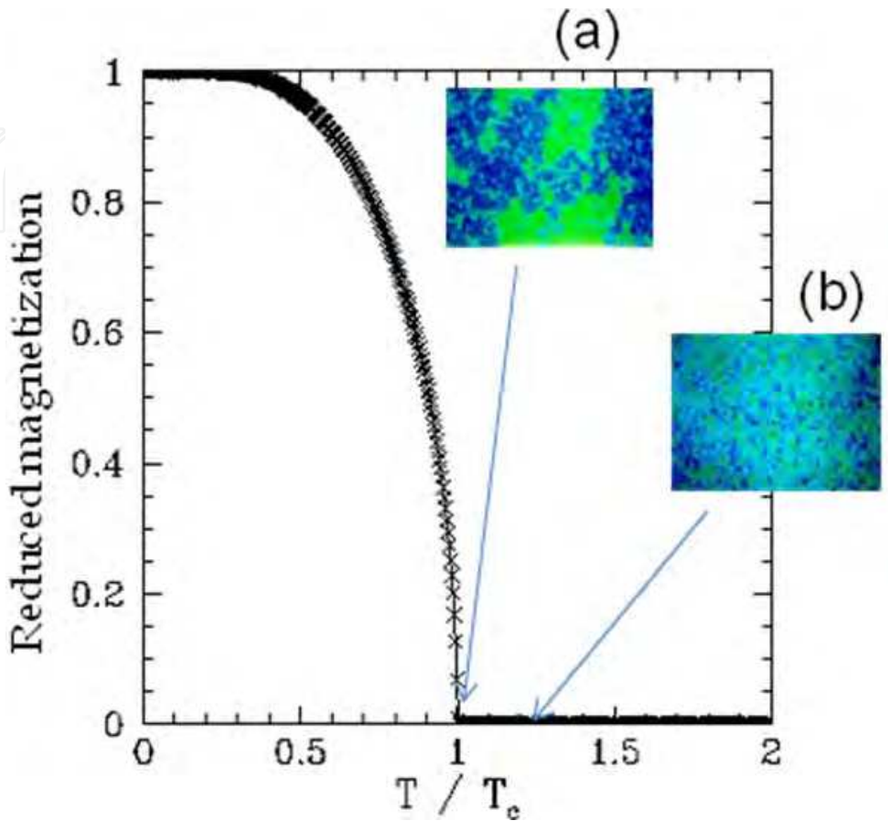


Fig. 1. Reduced magnetization versus temperature. Two spin configurations are shown: (a) clusters of spins at T_C, (b) disordered spins at 1.28 T_C. (Blue pixel = spin down, green pixel = spin up).

Once the spin model is established, it is simulated and the macroscopic properties of the statistic system are extracted from simulation results. For instance, the magnetization is extracted from the simulation results of the Ising model and we can see in Figure 1 that it vanishes when the system temperature reaches T_C. Physicists are interested in calculating the critical exponent of the magnetization at the phase transition.

In this study, we focus on the high temperature phase where the magnetization is equal to zero because fifty percent of the spins are "up". Moreover, the spins are entirely disordered at these temperatures. The time evolution of the disordered spin configurations can generate series of pseudo-random array of bits (since one spin can be coded on one bit). This feature makes this Pseudo-Random Number Generator (PRNG) usable in a symmetric crypto-system.

After adopting the Ising model, we have to choose the corresponding simulation algorithm. In addition, we want an algorithm suitable for an optimized hardware implementation of the PRNG and the associated crypto-system.

3. An algorithm suitable for hardware implementation

3.1 Introduction

Except some spin models solved analytically (Baxter, 1982), statistical models are more generally solved by numerical techniques. The most popular technique is the Monte Carlo computer simulation (Baillie, 1990; Metropolis & Ulam, 1949). The goal of computer simulations is to generate spin configurations typical of statistical equilibrium, in order to obtain the physical observable value of the macroscopic system.

Starting from any spin configuration, the algorithms used to simulate the Ising model aim to generate a series of spin configurations appearing with a probability in accordance with the statistical thermodynamics, i.e. proportional to $e^{-(E/kT)}$ where E is the internal energy of the configuration, T the system temperature and k the Boltzmann constant.

The trajectory through the configuration space is induced by local microscopic rules that can be probabilistic or determinist. In this chapter, we are only interested in the algorithms based on local microscopic rules completely determinist in order to design a machine dedicated to a symmetric crypto-system. In this case, the machines used for encryption and for decryption process must be identical.

First, we describe the standard and most common example of Ising simulation algorithm: the Metropolis algorithm (Metropolis et al., 1953). Then we focus on the "Reservoir algorithm" particularly suitable for the hardware implementation of our enc/decryption machine. We will not describe the huge number of algorithms proposed in the literature for the 2D-Ising model, because most of them are intended to improve the so called "critical slowing down" (Selke, 1993) appearing at the phase transition. Since we want to study the physical system at high temperature, far from the transition, we do not need so sophisticated algorithms.

3.2 Metropolis algorithm

We focus on the 2D-Ising model. Let us consider a square lattice of N sites with one spin *S* at each site. Each site interacts with its four nearest neighbours. The spins may be "up" or "down". A spin "up" is coded "1". A spin "down" is coded "0". The energy of a link between two neighbour spins pointing towards the same direction (parallel spins) is 0. This link energy is equal to 1 if the two spins are anti-parallel. The total energy of this system is the sum of the energies of the 2N links.

The main idea of the Metropolis algorithm (Metropolis et al.,1953) is to slightly modify a spin configuration and to accept or not this modification versus a probabilistic rule related to the Boltzmann weight. The Metropolis algorithm generates a Markow chain of spin configurations. Starting from any initial spin configuration, the successive configurations lead to the macroscopic equilibrium. The algorithm itself is described thereafter:

Metropolis algorithm

a. Choose an initial spin configuration
b. Select one spin S_i (represented by the red arrow in Figure 2) to be updated and try to flip its spin. Nevertheless flipping a spin has a cost in terms of magnetic energy. Indeed, if the spin S_i of site *i* flips, the magnetic energy varies as:

$$\Delta M_i = -2\,[\Sigma_j(S_i \text{ xor } S_j) - 2] \quad (1)$$

where j refers to the four neighbours of the site i.

For instance, if we focus on the first configuration presented in Figure 2, the four neighbours spins are parallel to the central spin, so the magnetic energy (sum of the link energies) is equal to 0. Now, if we flip the central spin, the four links become twisted and the magnetic energy becomes equal to 4. So, flipping the central spin costs $\Delta M_i = 4$. Notice that ΔM_i is always even.

c. Will the central spin flip or not? If ΔM_i lowers the system energy or let it unchanged ($\Delta M_i \leq 0$) then the spin flips. Otherwise the spin flips only with the probability $p = e^{-\Delta Mi/kT}$. Notice that, at constant ΔM_i, this probability increases with T. In practice, a random number r ($0 \leq r \leq 1$) is generated and, if $r \leq e^{-\Delta Mi/kT}$, the spin S_i can flip. Otherwise, it remains unchanged.

d. Return to step (b) until all the spins are updated.

The description of this algorithm leads to two important remarks: during step (c) the Metropolis algorithm needs a real number r randomly chosen (this is not suitable for an optimized hardware implementation of the Metropolis algorithm) and the control parameter is the temperature T.

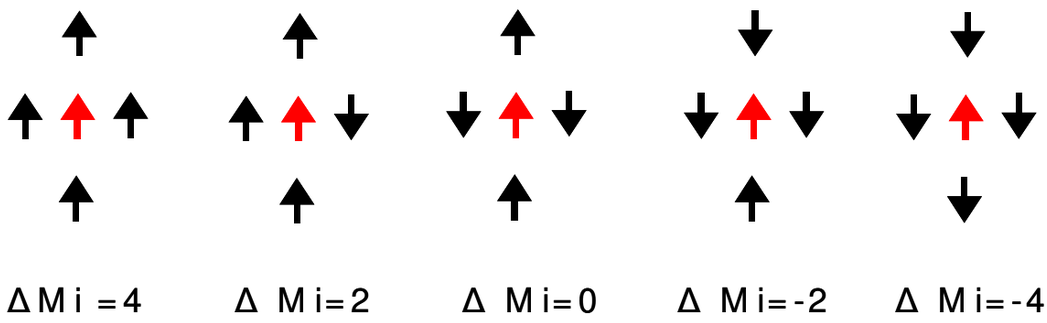


Fig. 2. Magnetic energy cost ΔMi to flip the red spin of the central site i

3.3 Microcanonical Reservoir algorithm

First, notice that the Reservoir algorithm (Ottavi & Parodi, 1989; Perez & all, 1995) and the Creutz microcanonical algorithm (with fixed demons) (Creutz, 1986) are very similar. However the Reservoir algorithm offers solutions for some low temperature simulation problems. These improvements are not considered in the present work since we are only interested in simulating the high temperature phase of the Ising model.

The statistical system to simulate is the same as the previous one except that each site i has one spin S_i and also a reservoir containing an energy E_{ri}. So, two kinds of energies are involved in this model. The first one is the magnetic energy, sum of all the link energies of the system. The second kind of energy is called "reservoir" energy, sum of all the private site reservoir energies E_{ri} of the system.

The algorithm itself is described thereafter:

Reservoir algorithm

a. Choose an initial spin configuration and an initial reservoir energy configuration.
b. Select one spin S_i (represented by the red arrow in Figure 2) to be updated and try to flip its spin. Nevertheless flipping a spin has a cost in terms of magnetic energy. Indeed if the spin S_i of site i flips, the magnetic energy varies as:

$$\Delta M_i = -2\,[\Sigma_j(S_i \text{ xor } S_j) - 2] \quad (1)$$

where j refers to the four neighbours of the site i.
c. Will the central spin flip or not? The local rule is: if ΔM_i is smaller than or equal to E_{ri}, the spin S_i flips. Otherwise, S_i does not change. In other words, if the site has enough reservoir energy to pay the flip then the spin can flip effectively.
d. Return to step (b) until all the spins are updated.

Some important comments can be made:

1. The total energy E of the system is the sum of the magnetic energy and of the reservoir energy. E remains constant since the Reservoir algorithm is a microcanonical algorithm. This energy conservation is a very useful tool to test a hardware (or software) implementation of this algorithm.
2. The control parameter is the total energy E and no more the temperature T as is the case in Metropolis algorithm. Nevertheless, we shall see later that we can establish a relation between E and T (Equ. 3).
3. Another important point is that no random number r ($0 \leq r \leq 1$) is needed as input. Moreover, the reservoir energy can be encoded on few bits.

We can conclude that the Reservoir algorithm is more suitable for hardware implementation than the Metropolis algorithm. Next sections will describe the parallelization of the selected algorithm in a Cellular Automata (CA) fashion, and its hardware implementation in a symmetric cryptosystem.

4. Parallel implementation of Reservoir algorithm

4.1 Mapping a cellular automaton?

The qualities of simplicity, parallelism, and locality of the CA are very appreciated for hardware implementations. Moreover, a uniform two-dimensional CA with the von Newman neighbourhood is quite similar to a two-dimensional Ising spin lattice.

We already described (Charbouillot & all, 2008) the software implementation of Reservoir algorithm rules in a multi-purpose hardware cellular automaton named Programmable Hardware Cellular Automata (PHCA). Here, we present a hardware parallel structure dedicated to the Reservoir algorithm. The design of this fine-grained structure was inspired by the mapping between the 2D-Ising model and a 2D-Cellular Automaton.

Cellular Automata are dynamical systems where space, time, and variables are discrete. They are traditionally implemented as an array of cells with a specific rule. The rule can be

seen as a function whose arguments are the states at time t of the neighbouring cells (and possibly the state of the considered cell itself) and whose value is the state of the considered cell at time t+1. If all the cells obey to the same rule, the CA is uniform.

Let us focus on two-dimensional CAs. Two kinds of neighbourhoods are usually considered: five cells, consisting of the considered cell and its four nearest neighbours respectively situated at East, South, West and North. This is the von Newman neighbourhood. The second kind of neighbourhood is obtained by also including the cells situated at East-South, South-West, West-North and North-East. This is the Moore neighbourhood implying eight surrounding cells.

To implement the 2D-Ising model, we choose the von Newman neighbourhood, so the next state of cell(*i;j*) is defined by Equ. 2:

$$x_{i,j}(t+1) = f\,[x_{i,j}(t),\, x_{i-1,j}(t),\, x_{i+1,j}(t),\, x_{i,j-1}(t),\, x_{i,j+1}(t)] \tag{2}$$

Fixed or null boundary conditions can be added at the boundary of the external cells of the array. More often, to avoid finite-size array effects, cyclic boundary conditions are applied. In this last case, the two-dimensional array becomes a torus.

Starting from the 2D-Ising model, replace "lattice" by "array", "sites" by "cells" and "iteration step" by "time step" and you have a cellular automaton. However, a problem appears when we want to simultaneously update all the spins at each time step. The so-called "feedback catastrophe"(Vichniac, 1984) illustrates this problem as follows.

Start the simulation with an aligned configuration (all the spin are parallel) below the Curie temperature (say, 0.8 T_C). During the first time steps, some spins flip and flip back like in a standard Monte Carlo calculation, but as soon as two spins (or cells) with contiguous corners flip during the same time step, a spurious chessboard pattern starts to grow. This can lead to two antiferromagnet spin configurations (one corresponds to the last scheme of Figure 2, and the other one corresponds to the complementary situation). These two configurations alternate because each spin "up", surrounded by four spin "down", will flip in order to align itself with its neighbours, which themselves will also flip, doing "the same reasoning".

This problem can be solved if we distinguish two kinds of sites: the black ones and the white ones, distributed in a chessboard fashion in the 2D-lattice. With this process, all the white sites can be updated simultaneously. Then, at the next time step, all the black sites will be updated simultaneously. So, two time steps are necessary for an entire lattice update and the parallel array of cells is no more exactly a CA. When the state of a white site has to be updated, its four nearest neighbours are black and cannot be updated, and the "feedback catastrophe" is avoided.

In conclusion, the Reservoir algorithm was easily amenable to true parallel processing. The variables assigned to each site are: a reservoir energy coded on few bits (the number of bits depends on the number of sites in the lattice, and on the global reservoir energy), a spin encoded on one bit, and a colour encoded on one bit. However an "iteration step" is equal to two "time steps".

4.2 Initialization phase and result extraction

At the beginning of the simulation, an initial spin configuration must be established and the total reservoir energy must be shared among the lattice sites. Then, we have to choose an initial spin configuration, for instance uniform: all the spins are down. Notice that, in this particular case, the total energy E of the lattice is only constituted by the reservoir energy since the initial magnetic energy is equal to zero.

At the beginning, the system is not in its statistical equilibrium and physical quantities (magnetization, energies) fluctuate considerably. Even though, in this study, we do not want to use these quantities but only the spin configurations of the disordered phase, we need to know the temperature of the system.

Starting from any initial reservoir energy distribution, it is interesting to see that after a transient regime, the reservoir contents obey to the Boltzmann statistical law. At this step, Equ. 3 allows to compute the system temperature T.

$$T = \frac{2.J}{k.Ln[(2j/ < Er >) + 1]} \tag{3}$$

where $< Er >$ is the mean value of the reservoir energy.

Simulating the Ising model using either the Metropolis algorithm or the Reservoir algorithm, leads to the same statistical results. Indeed, the same M(T) curve is obtain by both algorithms and the same precision is reached for its critical exponent. Figure 1 presents the M(T) curve and some corresponding spin configurations. We are interested in the rightmost one, which corresponds to apparent disorder.

At each iteration step, the spin configurations evolved, from a simple initial state and under simple rules without external random input, towards more and more complicated patterns. Once the statistical equilibrium is reached, these patterns appear to be random. These successive bit array configurations could be the long unpredictable key sequences (or Cipher Keys) necessary for en/decrypting a data flow. However, it is necessary to test the quality of the generated randomness.

4.3 Test of randomness

Randomness is one of the crucial points of a key stream for secure stream ciphers. Various types of statistical tests for randomness have been proposed (Kim & Umeno, 2006, Tomassini et al., 2000). We will focus on the Diehard random number generator testing suite proposed in (Marsaglia, 1998). The list of the Diehard tests is given in Table 1. Generators that pass these tests are considered "good".

Most of the Diehard tests need a 12 Mbytes input file, but three of them need a 270 Mbytes input file. Most of these tests return "p-values", which should be uniform on [0,1] if input files contain truly random bits. If the PRNG is bad, most of the p-values will be 0 or 1.

We applied the Diehard tests to successive spin configurations generated by the simulation of the 2D-Ising model at high temperature paramagnetic phase. We have tested the Ising system for different initial reservoir energies.

1. Birthday Spacings
2. GCD
3. Gorilla
4. Overlapping Permutations
5. Ranks of 31x31 and 32x32 Matrices
6. Ranks of 6x8 Matrices
7. Bitstream
8. OPSO Overlapping-Pairs-Sparse-Occupancy
9. OQSO Overlapping-Quadruples Sparse-Occupancy
10. DNA
11. Count the 1's in a Stream of Bytes
12. Count the 1's in Specific Bytes
13. Parking Lot
14. Minimum Distance
15. 3D Spheres
16. Sqeeze Test
17. Overlapping Sums
18. Runs Up and Down
19. Craps

Table 1. List of Diehard tests

The tests were carried on under the following conditions:

- During the initialization phase, all sites have the same reservoir energy except some sites (called "hot points") which have a higher one.
- Then, lattice iterations are performed till the successive patterns of the spin configurations seem disordered as presented in Figure 1 or in Figure 7.

The input file for the Diehard test program is a binary file resulting from the concatenation of the random keys C_i generated by the Ising spin configuration. These keys are built as follows. Let $K_i(t)$ be the concatenation of all the spin values of lattice row i at time t, the first encryption key of the random sequence is:

$$C(t_m) = K_0(0) \text{ xor } K_1(1) \text{ xor } \ldots \text{ xor } K_m(t_m) \quad (4)$$

where t is the iteration step (equal to two time steps).

The curve in Figure 3 gives an example of test results. It is obtained by applying the Diehard tests to a sequence of 70M keys $C(t_m)$, $C(t_m + 1)$, …, $C(t_m + a)$ extracted from a 128x128 2D-Ising lattice (Figure 7). Figure 3 gives the proportion of pass tests versus R (where $R = E_R/2$). These results come from interpretation of p-values. If all the p-values within a test are greater than 0.01 and less than 0.99, the test is considered as "pass".

These results show that, in this example, R must be chosen between 1000 and 3000 to obtain high-quality randomness. The test fails for low reservoir energies because the system is not in the paramagnetic phase. It also fails for too higher energies because all spins flip simultaneously.

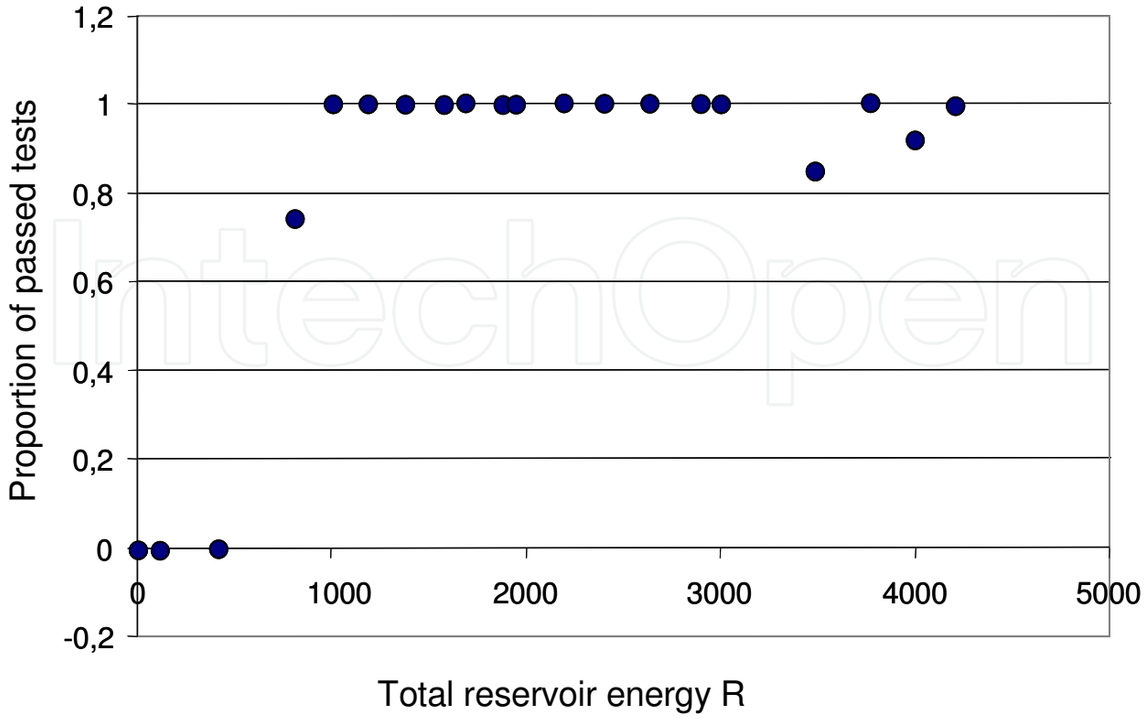


Fig. 3. Example of Diehard test results

5. Ising spin machine

We designed an Ising Spin Machine (ISM) dedicated to simulate the 2D-Ising Model with the Reservoir algorithm and to enc/decrypt a data flow.

5.1 ISM architecture

ISM is a parallel machine, entirely synchronous and autonomous, containing a finite state machine which controls a 2D-array of cells. This array contains *n x m* identical cells (Figure 4). Each cell is linked to its North, East, West, and South nearest neighbours (NEWS array). We implemented the NEWS array with cyclic boundary conditions: the North border is linked to the South border, and the West border is linked to the East one. All these local links are bidirectional.

The array has *n* parallel 1-bit data-in inputs, *n* parallel 1-bit data-out outputs, some control inputs and some state outputs. We add *n* global connection lines, with South to North direction (represented by grey arrows in Figure 4), to ensure data shifts.

The structure of a cell is detailed in Figure 5. Each cell is designed to manage a site of the Ising lattice under the established rule. So, a cell contains a combinational logic block ΔM', an adder, registers and multiplexors. ΔM' computes (-ΔMi/2) where ΔMi is the cost of the spin flip (Figure 2). ΔMi is divided by 2 since we noticed that this quantity is always even; in return the user has to distribute twice lower initial reservoir energy. The adder computes the reservoir energy which remains if the spin flips. If this energy is positive, this energy and the flip of the spin are registered.

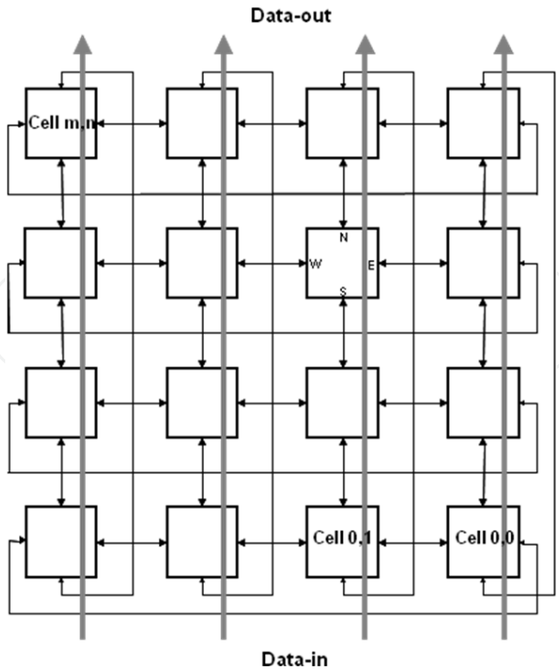


Fig. 4. Cell array architecture

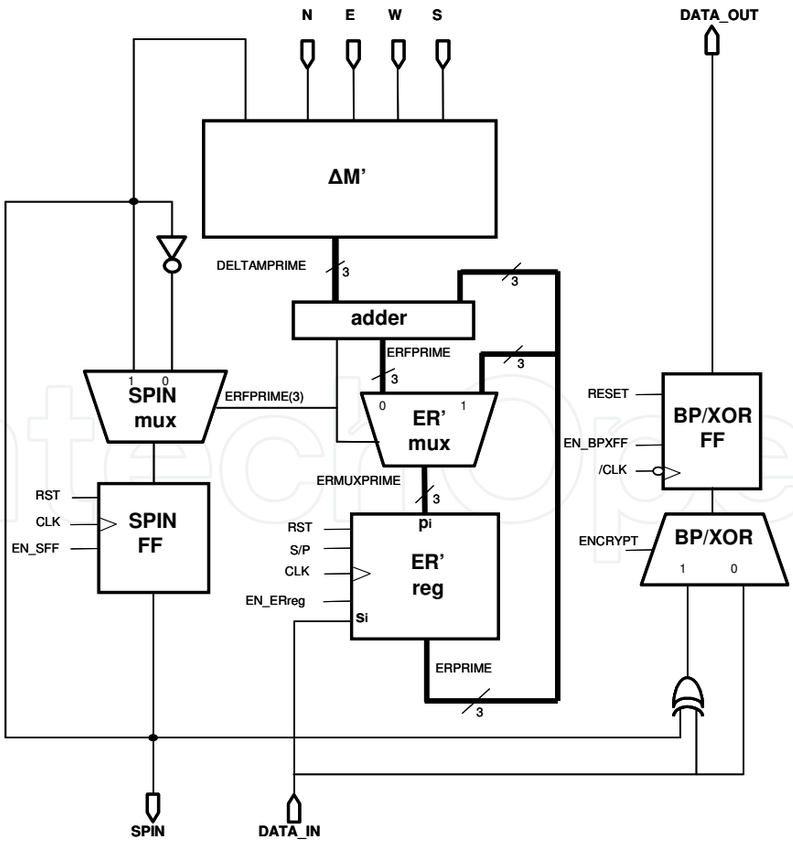


Fig. 5. Cell architecture

The data-in input is used during the initialization phase (encrypt = 0) to introduce the reservoir energy, and during the encryption phase (encrypt = 1) to "XOR" each bit of data with the registered spin bit.

5.2 Data encryption with ISM

At time t = 0, at the beginning of the encryption process illustrated in Figure 6, the first row R_0 of clear data is introduced through the south input of the PE array and "XORed" with $K_0(0)$. Then, at time t = 1 (time is here the iteration step), the result $D_0(0)$ is shifted to the north and "XORed" with $K_1(1)$ and so on.

At time $t = t_m$, the first encrypted data row $D_0(t_m)$ available at the north of the PE array, is given by :

$$D_0(t_m) = R_0 \text{ xor } C(t_m) \tag{5}$$

where $C(t_m) = K_0(0)$ xor $K_1(1)$ xor … xor $K_m(t_m)$ is the first encryption key.

The second encrypted data row $D_1(t_m+1)$is

$$D_1(t_m+1) = R_1 \text{ xor } C(t_m+1) \tag{6}$$

where $C(t_m+1) = K_0(1)$ xor $K_1(2)$ xor … xor $K_m(t_m+1)$ is the second encryption key and so on.

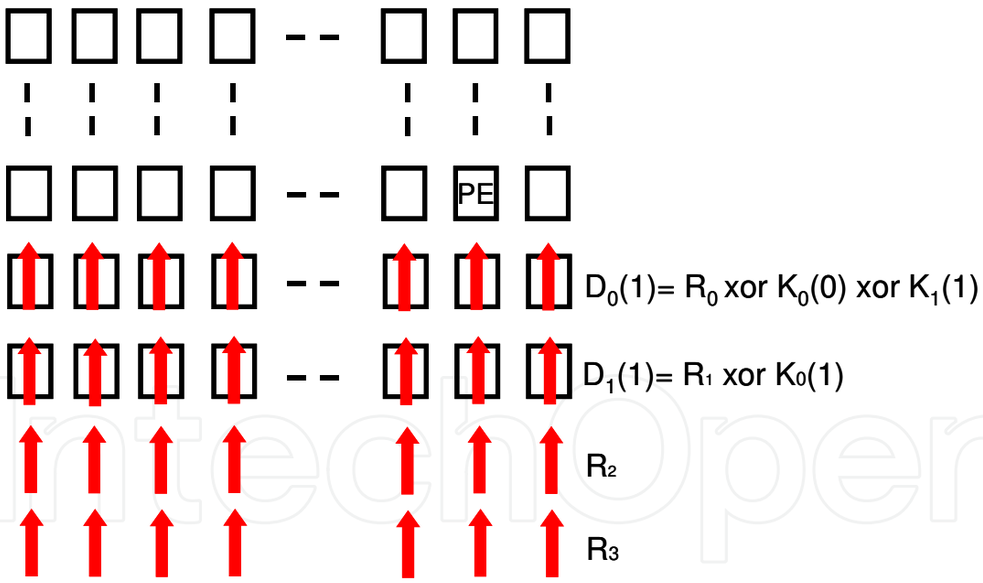


Fig. 6. South-North shift of the data to encrypt

5.3 ISM-based image encryption/decryption system

An application example of a 128x128 cell array ISM, is the colour image enc/decryption system shown in Figure 7. The clear original image given in Figure 8 is a colour image of size 640x853 pixels. Each pixel is coded on 3 bytes (Red, Green, Blue) so each line of this image can be divided into 120 128-bit words to fit in the cell array horizontal size. This resizing operation is not presented in Figure 7.

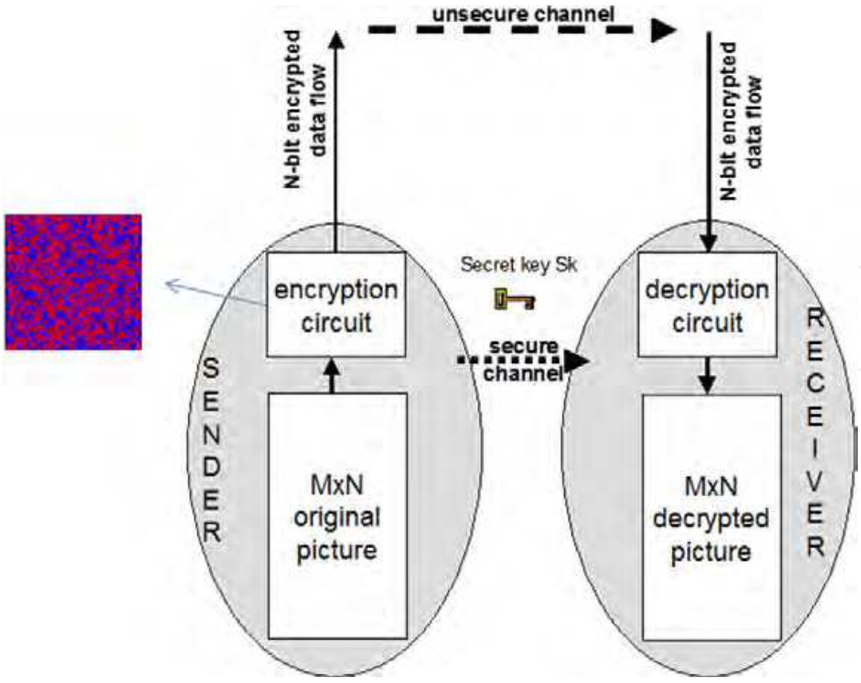


Fig. 7. Image encryption/decryption system. A disordered spin configuration generated by the Ising Spin Machine is shown at the left side.

In order to ensure a secure data exchange, both the Sender and the Receiver need an ISM. The operations to encrypt and decrypt the data are detailed thereafter:

- The Sender imposes the initial spin values S and distributes the total reservoir energy R. Then, the U initial spin lattice configuration updates are performed. The spin configuration shown in Figure 7 was obtained after 2000 iterations, starting with all the spins "down" (S = 0) and with a reservoir energy of 2 for each cell except for 3 cells (called "hot points") which received an energy of 4. Hot points coordinates constitute the information H. The concatenation of S, H and U builds the secret key S_k that must be transmitted to the Receiver through a secure channel, before the encrypted image is sent.
- After a resizing operation, the clear image is introduced through the South data-in of the Sender's ISM, one 128-bit word at a time. These data shift to the North and are encrypted at each iteration step. The resulting encrypted image is shown in Figure 8 (after inverse resizing). One can notice that the initial picture is completely scrambled at this step.
- Before receiving the encrypted image, the receiver gets the secret key S_k through a secure channel. He initializes its ISM with S and H, and controls U spin lattice configuration updates.
- Then, the encrypted image is introduced into the Receiver's ISM for the decryption process. After an inverse resizing operation, the decrypted image is exactly identical to the original clear image.

Two types of keys are involved in this process: the long sequence of the C_k Cipher Keys generated by ISM and the secret key S_k (Seredynski et al., 2004).

In order to test the feasibility of the Ising Spin Machine we have implemented an ISM version with a 32 x 32 cell array into a XC3S5000-5 Xilinx FPGA. As expected, the hardware implementation was easy since the architecture is simple, regular, and involves integer arithmetic and logic operations. This ISM provides good performances since the throughput is 2.02 Gbps (a 32-bit data is encoded every 2 time steps). However, ISM is resource consuming compared to other simple stream ciphers (Chen & Lai, 2007; Machhout et al., 2009). This is due essentially to its fully parallel structure: 5123 Flip-Flops are needed (five per cell, and three for the Finite State Machine).

An important point is related to the microcanonical quality of the Reservoir algorithm. The fact that the total energy is kept constant is a powerful tool to test the Ising Spin Machine. One erroneous bit either in the reservoir energy or in the spin has irreversible consequences when simulations are in progress. The fault is immediately detected.

Fig. 8. Clear image and encrypted image.

6. Conclusion

This work starts with a Physicist point of view on some algorithms invented for Statistical Physics, and moves towards a Cryptosystem Designer point of view. This approach is not new since the Ising model invented for a physical system was very useful in a large spectrum of domains. Our contribution consists in taking a "determinist Monte Carlo" method to simulate the Ising model for finally generating pseudo-random bit streams. This method, called "Reservoir algorithm", involves only integer arithmetic and logic operations and can be easily implemented either in hardware or in software.

We designed the Ising Spin Machine by adding data flow encrypting capabilities to the hardware implementation of the Reservoir algorithm. ISM has a fine-grained parallel structure and is based on Statistical Physics. In the Metropolis algorithm, the Boltzmann law, basis of the Statistical Physics, is introduced through the local rule to flip a spin. In the Reservoir algorithm no such a law is introduced. However, it is very interesting to see that the system itself, more exactly its reservoir energies, finally obey to this law. So, the reservoirs of the Ising Spin Machine intrinsically obey to the Boltzmann law and, under this energy condition, its spin configurations can generate the Cipher Keys to encrypt/decrypt data streams.

A FPGA implementation of a 32x32 cell array version of ISM is used in a symmetric stream cipher crypto-system for an image enc/decryption process. It performs 2 Gbps and could be used for real-time video applications. Moreover, ISM throughput could be improved. We saw that the chessboard trick is a solution to avoid the "feedback catastrophe" that could occur when all the sites are updated concurrently. The consequence of this solution is that two time steps are necessary to update the whole array of cells. Instead of the chessboard trick, a solution could be to endow each cell with four states in order to accommodate for two spins. This last solution could multiply the ISM throughput by a factor 2.

A same approach consisting in modeling the nature with fine-grained fully parallel systems can be adopted in other investigation domains. Adding high degree of redundancy to such systems is our inspiring source to design nanotechnology device architectures.

7. References

Bagnoli, F. & Francescato, A. (1990). Cellular Automata and Modeling of Complex Physical Systems, Boccara; N, Vichniac, G. & Bidaux, R. Springer-Verlag, (Ed.) (1990), pp. 312

Baillie, C. (1990). Lattice spin models and new algorithms : a review of Monte Carlo computer simulations. *International Journal of Modern Physics C*, Vol.1, Issue 01, (1990), pp.91-117

Baxter, R. (1982). *Exactly Solved Models in Statistical Mechanics* (1982), Academic Press Inc. LTD, ISBN 0-12_083180-5, London

Charbouillot, S.; Perez, A. & Fronte, D. (2008). A programmable Hardware Cellular Automaton: Example of Data Flow Transformation. *VLSI Design Journal,* Vol.2008, (2008), pp. 1-7

Chen, R. & Lai, J. (2007). Image security system using recursive cellular automata substitution, *Pattern Recognition,* Vol.40, (2007), pp.1621-1631

Creutz, M. (1986). Deterministic Ising Dynamics. *Annals of Physics,* Vol.167, (1986), pp. 62-72

Ising, E. (1925). Beitrag zur Theorie des Ferromagnetismus. *Zeitschrift fur Physik,* Vol.31 (1925), pp. 253-258

Kim, SJ. & Umeno, K. (2006). Randomness Evaluation and Hardware Implementation of Nonadditive CA-Based Stream Cipher, Available from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.135.3466

Machhout, M.; Guitouni, Z.; Zeghid, M. & Tourki, R. (2009). Design of reconfigurable image encryption processor using 2-D Cellular Automata generator. *International Journal of Computer Science and Applications,* Vol.6 (2009), pp. 43-62

Marsaglia, G. (1998) "Diehard", Available from http://www.cs.hku.hk/cisc/projects/va/diehard.html

Metropolis, N & Ulam, S. (1949). The Monte Carlo method. *Journal of American Statistical Association,* Vol.44 (1949), pp. 335-341

Metropolis, N; Rosenbluth, AW.; Rosenbluth, MN.; Teller, AH. & Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics,* Vol.21, No.6, (1953), pp. 1087-1092

Onsager, L. (1944). Crystal Statistics. I. A Two-Dimensional Model with a Order-Disorder Transition. *Physical Review,* Vol.65 (1944), pp. 117-149

Ottavi, H. & Parodi, O. (1989). Simulation of the Ising Model by Cellular Automata. *Europhysics Letters,* Vol.8 (1989), pp. 741

Perez, A.; Ottavi, H. & Cotton, M. (1995). The Ising model as a test for a personal parallel computer. *Computational Materials Science,* Vol.4 (1995), pp. 133-142

Sarkar, P. (2000). A Brief History of Cellular Automata. *Journal of ACM Computing Surveys,* Vol.32, No.1, (March 2000), pp.80-107

Sathyanarayana, S.; Aswatha, M. & Hari Bhat, K.. (2011). Symmetric Key Image Encryption Scheme with Key Sequences derived from Random Sequence of cyclic Elliptic Curve points, *International Journal of Network Security,* Vol.12, No.3, (2011), pp.137-150

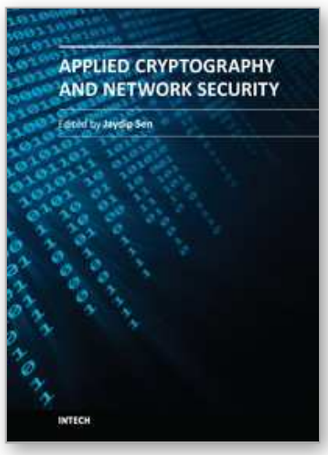
Selke, W; Talapov, A. & Schur, L. (1993). Cluster-flipping Monte Carlo algorithm and correlations in "good" random number generators. *JETP Letters,* Vol.58, No.8, (1993), pp. 665-668

Seredynski, F.; Bouvry, P. & Zomaya, Y. (2004). Cellular automata computations and secret key cryptography, *Parallel Computing,* Vol.30, (2004), pp.753-766

Tomassini, M; Sipper, M. & Perrenoud, M. (2000). On the Generation of High-Quality Random Numbers by Two-Dimensional Cellular Automata. *IEEE Transactions on Computers,* Vol.49, (2000), pp. 1146-1151

Vichniac, G. (1984). Simulating Physics with Cellular Automata. *Physica 10D* (1984), pp. 96-116

Wolfram, S. (1983). Statistical mechanics of Cellular Automata. *Reviews of Modern Physics,* Vol.55, No.3, (1983), pp. 601-644



Applied Cryptography and Network Security

Edited by Dr. Jaydip Sen

ISBN 978-953-51-0218-2

Hard cover, 376 pages

Publisher InTech

Published online 14, March, 2012

Published in print edition March, 2012

Cryptography will continue to play important roles in developing of new security solutions which will be in great demand with the advent of high-speed next-generation communication systems and networks. This book discusses some of the critical security challenges faced by today's computing world and provides insights to possible mechanisms to defend against these attacks. The book contains sixteen chapters which deal with security and privacy issues in computing and communication networks, quantum cryptography and the evolutionary concepts of cryptography and their applications like chaos-based cryptography and DNA cryptography. It will be useful for researchers, engineers, graduate and doctoral students working in cryptography and security related areas. It will also be useful for faculty members of graduate schools and universities.

InTech Europe

University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China

Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen