International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 1, Issue 2 (October 2012), PP.25-36
www.irjes.com

Encryption And Decryption Algorithm Using Two Dimensional Cellular Automata Rules And 1D CA Based S-Box (1D Rule-30) In Cryptography

Sambhu Prasad Panda
C V Raman Computer Academy,Bidya Nagar,Mahura,Janla, Bhubaneswar-752054, Odisha

ABSTRACT : *A cellular automaton is a collection of "coloured" cells on a grid of specified shape that envolps through a no of discreate time steps according to a set of rules based on states of neighbouring cells. Cellular automata are dynamic systems which are discrete in space and time, operate on a uniform, regular lattice and characterized by "local interaction". A cellular automata can be thought of as a stylist universe space is represented by a uniform grid, with each cell containing a few bits of data, time advances in discreate steps and the laws of the "universe" are expressed in, say, a small loop up table, through which at each step each cell computes its new state from that of its close neighbours.Cryptography is the science of writing in secret code.Secret key cryptography uses a single key for both encryption and decryption. The sender uses the key (or some set of rules) to encrypt the plaintext and sends the cipher text to the receiver. The receiver applies the same key (or rule set) to decrypt the message and recover the plaintext. Because a single key is used for both functions, secret key cryptography is also called symmetric encryption. In this paper we present a new encryption and decryption algorithm for block cipher based on the linear (periodic boundary-PB) and nonlinear cellular automata rules. First we apply non linear CA rules (complements) to both plain text and key. Then PB CA rule is applied to the above results separately followed by the XOR operation of above results. After that the result of XOR operation is fed to non linear one dimensional cellular automata rule-30 (CA based S-box) and again PB CA rules are applied followed by CA based S-Box. One dimensional CA based rule-30 can be used as an effective encryption scheme due to its randomness and balancedness property satisfying the cryptographic criteria of Boolean functions of cryptography. The decryption process is carried out just similar to that of encryption but in the reverse way by applying the reverse rules of cellular automata. Both the process of encryption and decryption is performed for 8 number of rounds in order to avoid the dependency between the plain text and cipher text so that the our proposed algorithm is more secure than that of AES and DES algorithms.*

Keywords *Cryptography, cellular automata, substitution bytes, affine functions, linear, non linear, periodic boundary, correlation immunity*

I. INTRODUCTION

Cryptography has become a basic requirement in this age of global electronic connectivity to secure data storage and transmission against the possibility of message eavesdropping and electronic fraud. The effective measure of a cryptosystem is how long it can be used to encrypt and decrypt messages without the 'key' being broken using cellular automata (CA) rules. Cellular automata (CA) based encryption algorithms provides good security due to the cryptographic criteria of Boolean functions satisfied by the cellular automata rules. Here we use the symmetric key cryptosystem where the sender and the receiver share only one key. The sender uses the key to encrypt the message using linear and non-linear rules of cellular automata. The actual message is called plain text and the encrypted message is called cipher text. Again the receiver decrypt the message using the same key (used by sender) to find the actual message using the complement or reverse rules of cellular automata which are linear and non-linear in nature.

The remainder of the paper is organized as follows. Section 2 introduces the concept of Boolean functions and cellular automata. In Section 3, we discuss some works of cryptography applied to one dimensional and two dimensional cellular automata. Section 4 describes our new encryption and decryption algorithm by using cellular automata rules.

II. BACKGROUND

1.1 Boolean Function and its properties

Any Boolean Function f in n variables is defined as a map

$$f : \{0,1\}^n \longrightarrow \{0,1\}$$

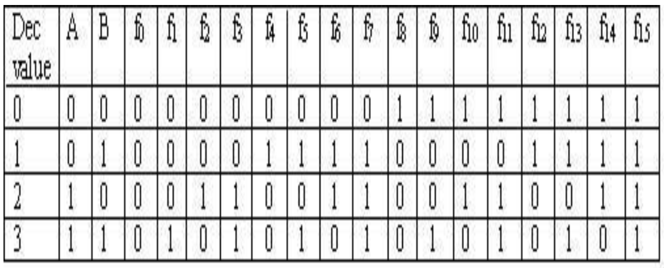
There are 2^{2^n} Boolean functions out of which 2^n are linear Boolean functions and 2^{2^n} - 2^n are nonlinear Boolean functions. For example if n=2, then

$f :\{ 0, 1\}^2 \longrightarrow \{0, 1\}$

$\Rightarrow f: \{0, 1\} \times \{0, 1\} \longrightarrow \{0, 1\}$

$\Rightarrow f: \{00, 01, 10, 11\} \longrightarrow \{0,1\}$

Figure1 shows the 2^{2^n} = 2^4 =16(n=2) number of Boolean functions in two-variables in which the function number uses convention that the decimal value is the output binary column vector taken from top to bottom.



Dec value	A	B	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
1	0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
2	1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
3	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Figure 1: Boolean Functions

There are two kinds of Boolean functions:

- Group of linear functions.
- Group of non-linear functions.

A boolean function f in n-variable is said to be linear if it satisfies the following linearity property: f(x+y) = f(x) + f(y), where x = (x_1,x_2,---,x_n) and y = (y_1,y_2,---,y_n).

There are 2^n linear Boolean functions in n-variables. Those are

- The zero function f (x_1, x_2, ---, x_n) = 0 is always a linear function and is termed as Rule 0.
- f (x_1,x_2,---,x_n) = x_i, (i= 1,2,3,---,n) are n-linear boolean functions which are termed as fundamental linear rules.
- The combinations of these n-linear functions taking some or all at a time, give rest of the 2^n-1 linear Boolean functions.

For n variables, there are 2^{2^n} Boolean functions, out of which 2^n linear Boolean functions and rest are non-linear Boolean functions.

From Figure 1, we get 16(=2^4) Boolean functions (f_0, f_1, f_2, …, f_{15}) in 2 variables (A, B), out of which there are 4(=2^2) linear Boolean functions (f_0,f_3,$f_5$and f_6) and 12 (=2^4-2^2) nonlinear Boolean functions (f_1,f_2,f_4,f_7,f_8,f_9,f_{10}, f_{11},f_{13},f_{14},f_{15}) in which f_9 , f_{10}, f_{12} , and f_{15} are 4 affine functions.In order to identify all these functions we proceed as follows:

The linearity property of any Boolean function is

f(X+Y)= f (X)+f (Y), Where X = (X_1, X2, ---, Xn) and

Y = (Y_1, Y2, ---, Yn)

Example: - Let S = {(0, 0), (0, 1), (1, 0), (1, 1)} = two variable inputs and f_3= $(0, 0, 1, 1)^T$ = output Boolean function(from figure1)

Let X = (0, 0), and Y = (0, 1) then X + Y = (0+0, 0 +1) = (0,1) => f (X +Y) = f (0,1) = 0

f (X) = f (0,0) = 0, f (Y) = f (0,1) =0 => f (X) + f (Y) = 0 +0 = 0

Hence f(X+Y) = f (X) +f (Y) => function f3 is linear. In this way we can find all other linear Boolean functions. In order to get all affine functions , we perform the exclusive-OR on column matrix $(1,1,1,---,1)^T$ in linear Boolean functions .

ie, Affine functions = Linear boolean function + $(1,1,1,---,1)^T$

f (A,B) = $(0,0,0,0)^T$ + $(1,1,1,1)^T$ = $(1,1,1,1)^T$ = f_{15}

f (A,B) = $(0,0,1,1)^T$ + $(1,1,1,1)^T$ = $(1,1,0,0)^T$ = f_{12}

f (A,B) = $(0,1,0,1)^T$ + $(1,1,1,1)^T$ = $(1,0,1,0)^T$ = f_{10}

f (A,B) = $(0,1,1,0)^T$ + $(1,1,1,1)^T$ = $(1,0,0,1)^T$ = f_9

So f_9 , f_{10}, f_{12} , and f_{15} are affine functions .But all affine functions are non-linear. In cryptography all linear Boolean functions and affine functions are not generally used. We cannot use only the linear or non linear Boolean functions in encryption and decryption because a secure cryptography algorithm contains both confusion as well as diffusion. Our conclusion is that to create a secure algorithm we have to use more number of non-linear Boolean functions for confusion and less number of linear Boolean functions for diffusion in different stages of the algorithm. Now we can calculate the non-linearity of non-linear Boolean functions. For 2-variables, we find both linear functions and affine functions. Now we calculate the non-linearity of f_1, f_2, f_4, f_7, f_8,

f_{11}, f_{13}, f_{14} from the affine functions $f_9, f_{10}, f_{12}, f_{15}$. In order to calculate the non linearity of functions, we have to find the hamming distance. The hamming distance d (x_1, x_2) between two vectors (x_1, x_2) is the number of components where vectors x_1, x_2 differ. For two Boolean functions f_1, f_2..

On V^n, we define the distance between f_1 and f_2 by

$$d(f_1, f_2) = \#\{x \in V^n / f1(x) != f_2(x)\}$$

Non-linearity of f1 = min (distance from f_1 to $f_9, f_{10}, f_{12}, f_{15}$)

= min (1, 3, 3, 2) = 1

(e.g Distance between f_1=(0,0,0,1) and f_9=(1,0,0,1) = 1 because only one component (i.e. the 1st component of f_1 (=0) and f_9 (=1) are not equal to each other. Similarly distance between f_1=(0,0,0,1) and f_{10} =(1,0,1,0) =3 because in this case 1st, 3rd and 4th components are not equal to each other. Similarly we can check for all functions one by one.)

Non-linearity of f2 = min (3, 1, 3, 3) = 1

Non-linearity of f_4 = min (3, 3, 1, 3) = 1

Non-linearity of f_7 = min (3, 3, 3, 1) =1

Non-linearity of f_{11} = min (1, 1, 3, 1) = 1

Non-linearity of f_{13} = min (1, 3, 1, 1) =1

Non-linearity of f_{14} = min (3, 1, 1, 1) = 1

High non-linearity of f = max ($f_1, f_2, f_4, f_7, f_8, f_{11}, f_{13}, f_{14}$) = 1

In this way we can calculate high non-linearity Boolean functions for 9-variables of 2D CA for security purpose in cryptography.

Functions of degree at most one are called *affine* functions. An affine function with constant term equal to zero is called a *linear* function. The set of all n-variable affine functions is denoted by A (n). The nonlinearity of a n-variable function f is the minimum distance between the affine functions f and g which is given by nl (f) = min (d (f, g)). The hamming distance or distance d (x_1, x_2) between two vectors is the number of components where vectors x_1, x_2 differ. We note that all affine functions are non linear.

From figure1 it is clear that A=(0,0,1,1)=f_3,B=(0,1,0,1)=f_5, A XOR B=(0,1,1,0)=f_6 , and f_0 =(0,0,0,0) are four number of all linear Boolean functions . Here the output Boolean functions f_3 and f_5 are same as that of input Boolean functions A and B respectively. So in encryption or decryption if we get the output Boolean functions to be the same as that of input Boolean functions (places may change) then we call them as linear Boolean function providing diffusion property of cryptography. Now we take the NOT (or complement) of A, B, A XOR B, and f_0 and get the output Boolean functions as f_{12},f_{10},f_9 and f_{15} respectively, which are not permutations(or transpositions) but substitutions and hence the NOT (or complement) operation is the non linear Boolean function providing confusion property in cryptography .Now we apply OR and AND operations to input Boolean variables A and B to get the output Boolean functions A OR B= f_7 and A AND B= f_1 which completely substitutes the input Boolean variables to new Boolean variables and hence OR and AND operations provide non linear Boolean function providing confusion property in cryptography. Again we apply NOR and NAND operation to A and B to get $f_8$and f_{14}, which also provide non linearity or confusion. In this way we can identify the actual operations that are taking place between the inputs A and B for each of the output Boolean functions and apply them to cryptography for getting confusion as well as diffusion.

Therefore f_9 , f_{10}, f_{12} , f_{15} are affine functions .But all affine functions are non-linear. In cryptography all linear Boolean functions and affine functions are not generally used. We can not use only the linear or non linear Boolean functions in encryption and decryption because a secure cryptography algorithim contains both confusion as well as diffusion. Our conclusion is that to create a secure algorithim we have to use more number of non-linear Boolean functions for confusion and less number of linear Boolean functions for diffusion in different stages of the algorithm. Now we can calculate the non-linearity of non-linear Boolean functions. For 2-variables, we find both linear functions and affine functions. Now we calculate the non-linearity of $f_1, f_2, f_4, f_7, f_8,$

f_{11}, f_{13}, f_{14} from the affine functions $f_9, f_{10}, f_{12}, f_{15}$. In order to calculate the non linearity of functions, we have to find the hamming distance. The hamming distance d (x_1, x_2) between two vectors (x_1, x_2) is the number of components where vectors x_1, x_2 differ. For two Boolean functions f_1, f_2..

On V^n, we define the distance between f_1 and f_2 by

$$d(f_1, f_2) = \#\{x \in V^n / f1(x) != f_2(x)\}$$

Non-linearity of f1 = min (distance from f_1 to $f_9, f_{10}, f_{12}, f_{15}$)

= min (1, 3, 3, 2) = 1

(e.g Distance between f_1=(0,0,0,1) and f_9=(1,0,0,1) = 1 because only one component (i.e. the 1st component of f_1 (=0) and f_9 (=1) are not equal to each other. Similarly distance between f_1=(0,0,0,1) and f_{10} =(1,0,1,0) =3 because in this case 1st, 3rd and 4th components are not equal to each other. Similarly we can check for all functions one by one.)

Non-linearity of f2 = min (3, 1, 3, 3) = 1
Non-linearity of f_4 = min (3, 3, 1, 3) = 1
Non-linearity of f_7 = min (3, 3, 3, 1) =1
Non-linearity of f_{11} = min (1, 1, 3, 1) = 1
Non-linearity of f_{13} = min (1, 3, 1, 1) =1
Non-linearity of f_{14} = min (3, 1, 1, 1) = 1
High non-linearity of f = max (f_1, f_2, f_4, f_7, f_8, f_{11}, f_{13}, f_{14}) = 1
In this way we can calculate high non-linearity Boolean functions for 9-variables of 2D CA for security purpose in cryptography.

1.2 Cryptographic criteria for Boolean functions

- **Balanced ness:** A Boolean function must output zeroes and ones with the same probabilities.
- **Good non-linearity:** The Boolean function must be at the sufficiently high distance from any affine function.
- **High algebraic degree:** The Boolean function must be at high algebraic degree.
- **Good correlation-immunity** (of order m): The output of Boolean function must be statistically independent of combination of any m inputs. A balance correlation-immunity of order m Boolean function is called m-resilient**.**
- **Simple implementation in hardware:** Hardware implementation should be very simple.

1.3 Cellular Automata (CA)

A Cellular Automata (CA) is an idealized parallel processing machine in which the cell value is updated based on the updating rule, which involves the cell value as well as other cell values in a particular neighborhood.

2.3.1 Neighbourhood

Two-dimensional CA neighborhoods are:

1. VonNeumann:Only North, South, West and East neighbourhood.(Four neighbourhoods)
2. Moore: One adds the diagonals to Von Neumann to form nine neighbourhoods.
3. Extended Moore: One extends the distance of neighborhood beyond one.

Figure 2 and Figure 3 show the structure of one and two dimensional cellular automata neighborhood cells respectively. In both of these figures, central cell is denoted by CELL and its entire neighbourhood is denoted by N.

N	CELL	N

Figure 2: 1D neighborhood

N	N	N
N	CELL	N
N	N	N

Figure 3: 2-D Moore neighborhood

2.3.2 Types of cellular automata (in terms of dimensions)

(a) One-dimensional cellular automata
(b) Two-dimensional cellular automata

(a) One-dimensional cellular automata

A one-dimensional cellular automaton consists of two things: a row of "cells" and a set of "rules”. Each of the cells can be in one of several "states". The number of possible states depends on the automaton. Think of the states as colors. In a two-state automaton, each of the cells can be either black or white. Of course, we might just as easily use purple and orange to represent the states, if we thing that's prettier. In a three-state automaton, the states might be black, red, and blue. Over time, the cells can change from state to state. The cellular automaton's rules determine how the states change. It works like this: When the time comes for the cells to change state, each cell looks around and gathers information on its neighbors' states. (Exactly which cells are considered “neighbors" is also something that depends on the particular CA.) Based on its own state, its neighbors' states, and the rules of the CA, the cell decides what its new state should be. All the cells change state at the same time. We should think of the row of cells as a miniature "world" that runs through a sequence of "years" or "generations." At the end of each year, all the cells simultaneously decide on their state for the next year. When a CA is simulated on a computer, it's a good idea to have two rows of cells. One row is the actual world; the other is used as a convenient place for computing the cells' states in the next year. After all the cells have been processed, the new world can replace the old.

(b) Two-dimensional cellular automata

In Von Neumann neighborhood five cells are considered. That is Only North, South, East, West, and itself. In Moore neighborhood nine cells are considered (as shown in figure 3). The central cell updates the value by looking around the value of its neighbor by applying the cellular automata rule.

2.3.3 1D CA rules as Boolean functions

A 1D cellular automaton consists of a set of cells which can be in one of two states, on (i.e.1) and off (i.e.0). Applying different rules to the cell we can change its state. At any particular time the next value of the cell is determined by three values, its state and the state of its neighbors. Figure 4 shows the one dimensional neighborhood of cellular automata.

N	CELL	N

Figure 4: 1D neighborhood

So the different transitions for three cells are as follows: 000->0, 001->1, 010->0, 011->1,100->0,101->0,110->0,111->0. This may be written as a binary string in reverse order as 00001010=10(decimal) = Rule 10. Similarly we can form many more rules.

1.3.4 2D CA rules as Boolean functions

Table 1 shows all the rules of two dimensional cellular automata.

Table 1: 8-neighborhood CA rules

64	128	256
32	1	2
16	8	4

In this case the next state of a particular cell is affected by the current state of itself and eight cells in its nearest neighborhood (Table 1). These dependencies are formed by various rules. The central cell represents the current cell (i.e. the cell being considered) and all other cells represent the eight neighbors of that cell. The number within each cell represents the rule number(i.e. Rule 1, Rule 2, Rule 4, Rule 8, Rule 16, Rule 32, Rule 64 and Rule 128) characterizing the dependency of the current cell on that particular neighbor only. These 8 rules are called fundamental rules of cellular automata and are known as linear rules of cellular automata. In case the cell has dependency on two or more neighboring cells, the rule number will be the arithmetic sum of the numbers of the relevant cells, which gives the linear rules of cellular automata. So XOR operation is also linear rule of CA.

For example the 2D CA rule 170 (=2+8+32+128) refers to the 4 neighborhood dependency of the central cell on right, bottom, left and top. The number of such rules is ${}^8C_0+{}^8C_1+....+{}^8C_8=256$. Rule-170 will be applied uniformly applied to each cell. From the figure below, it is clear that rule-170 is same as cell changes its state by adding the states of its 4-orthogonal neighbors and the resultant matrix is:

$$\begin{bmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \xrightarrow{\text{Rule 170}} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & & 1 & 0 \;\; 0 \\ 0 & 1 & 1 & 1 \end{bmatrix}$$

1.3.5 Definitions

Null Boundary Cellular automata (NB CA): A null boundary CA is the one in which the extreme cells are connected to logic - 0 states.

Periodic boundary cellular automata (PB CA): A periodic boundary (PB) CA is the one in which the extreme cells are connected to each other.

Uniform Cellular Automata (UCA): A uniform cellular automata is the one in which same rules are applied to each of the cells of a Boolean matrix.

Hybrid Cellular Automata (HCA): If different rules are applied to different cells of a Boolean matrix, then we call it as hybrid cellular automata..

Group CA and Non-Group CA (GCA and NGCA): A group CA whose transformation is invertible (i.e. all the state in the state transition diagram lie in some cycle) is called a group CA; otherwise it is called a non-group CA. Group CA rules are used for encryption and decryption algorithm. Therefore for the application in cryptography it is necessary to find out the entire group CA and characterize them. We note that the XOR (linear rule of CA), compliment (non linear rule of CA), PB CA rule 2 (linear CA rule of CA), PB CA rule32 (linear CA rule of CA), PB CA rule8 (linear CA rule of CA), and PB CA rule128 (linear CA rule of CA) are all belongs to the category of group CA. The verifications of these groups CA are illustrated through examples in Section 4.3.1, Section 4.3.2, Section 4.3.4, Section 4.3.5, and therefore all these group CA rules of cellular automata can be used for encryption and decryption in cryptography.

III. RELATED WORK

Carlet [1] performed study on Boolean functions for cryptography. He utilized cryptographic criteria to identify the linearity (diffusion) and nonlinearity (confusion) operations involved in Data Encryption Standard (DES) and Advanced Encryption Standard (AES) algorithm. Wolfram [2] applied the one-dimensional cellular automata rules on stream cipher for security. Maitra et. al. [3] applied the method of generating key stream sequences for stream ciphers by combining the outputs of several linear feedback shift registers (LFSR) using a combined Boolean function. Das and Ray [7] present a new block encryption algorithm based on Reversible Programmable Cellular Automata theory. Their work ensures to generate 2^{256} potential keys. They use 128 bit block size and Reversible Programmable Cellular Automata. Tripathy and Nandi [10] proposed a light weight symmetric key cryptosystem using CA, called Lightweight Cellular Automata-based Symmetric-key Encryption (LCASE). LCASE meets the same specification as AES, that of satisfying the base security criteria (confusion and diffusion). They proposed a lightweight block cipher supports 128-bit block size with 128-, 192- and 256-bit keys, to confirm with the Advanced Encryption Standard (AES) specification. All these works were based on one dimensional cellular automaton for stream cipher. Also very few works have been carried out on two dimensional cellular automata for block cipher [9]. Research works have been carried out to characterize the linearity and non-linearity classes of cellular automata for cryptography [11].Research on randomness of rule-30 has been carried out [12].

IV. Encryption&Decryption Algorithm Using Cellular Automata Rules

1.4 Introduction

Two-dimensional Cellular Automata (CA) rules for encryption and decryption algorithm are used. Here both the sender and receiver use the same key.

1.5 Overall Structure

Table 2: Parameter of Algorithm

Key size in bits	128
Plain text block size in bits)	128
Number of rounds	8
Round key size in bits	128

Table 2 shows the parameters of the encryption and decryption algorithm. This algorithm contains substitution (non-linear cellular automata rule), permutation (linear cellular automata rule), complement (non-linear cellular automata rule), and XOR (linear cellular automata rule) operations. In crypto system, use of non-linear rule is more secure then use of linear rule. But creation of confusion (non linear CA rule) and diffusion(linear CA rule) operations are the two fundamental principles of cryptography .So in order to develop any encryption and decryption algorithm in cryptography we will apply both linear(diffusion) and non linear (confusion)operations or rules .But if we use more number of non linear rules and few linear rules for encryption and decryption then the algorithm will be more secure . In our encryption and decryption algorithm more number of non linear rules are applied as compared to Advanced Encryption Standard (AES) AES algorithm and also the number of rounds are less as comparison to AES and hence our algorithm is more secure than that of AES algorithm. Figure 5 shows the encryption and decryption for eight number of rounds where as Figure 10 and Figure 13 show one round encryption and one round decryption using cellular automata.

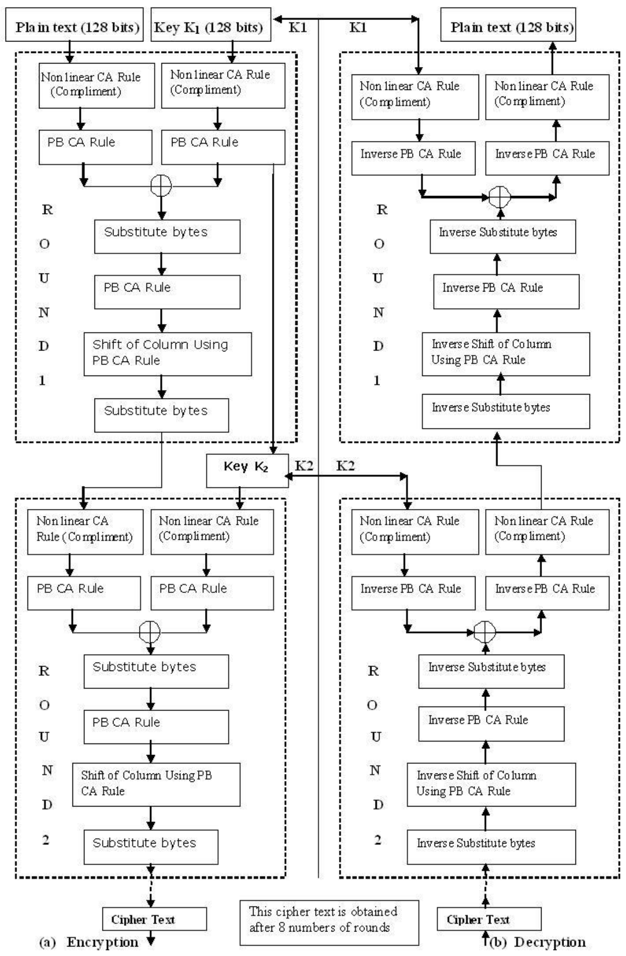
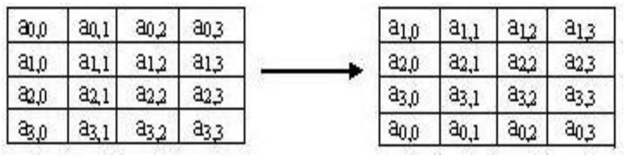


Figure 5: Encryption and Decryption algorithm applying CA rule

1.6 Encryption Algorithm

1.6.1 Non linear CA rule (Complement) and PB CA rule8

We take the length of the plain text as 128 bits and the length of key as 128 bits. First we convert the plain text as 4x4 matrix with each cell containing 1 bytes (= 8 bits). Now we apply non linear cellular automata rule (complement) to each bit of the plain texts. Similarly we write the 128-bits key in a 4x4 matrix and apply non linear CA rule (complement) to each cell of the key matrix. After the complementation we apply PB CA rule8 separately. Figure 6 shows the use of Periodic Boundary CA rule-8 to the matrix.



$$\begin{array}{|c|c|c|c|}\hline a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ \hline a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ \hline a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ \hline a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \\ \hline \end{array} \longrightarrow \begin{array}{|c|c|c|c|}\hline a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ \hline a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ \hline a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \\ \hline a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ \hline \end{array}$$

Figure 6: Periodic Boundary CA rule-8

For example, in the following matrix, the values in the second row have been shifted to first row, values in the third row have been shifted to second row and so on and values in the first row have been shifted to last row after applying periodic boundary (PB) CA rule-8 to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-8}} \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

4.3.2 XOR operation

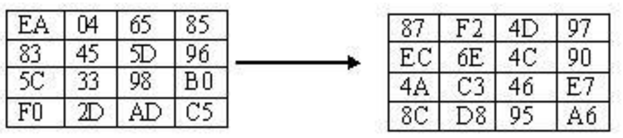
XOR operation is applied between the resulted cipher key and cipher text providing the linear rule of cellular automata and hence the diffusion property of the cryptography.

4.3.31D cellular automata based substitute bytes transformation (1D CA based S-Box):

Forward and Inverse Transformations:.CA based S-Box defines a 4 x 4 matrix of byte values . Each individual byte of state is mapped into a new byte by using 1D cellular automata rule-30 as per the following rules, which is known as forward transformation. Rule 30 is a function that maps three bits to one bit according to the equation:

f(a,b,c)=a+b+c+bc

Figure 7 shows an example of the 1D CA based sub bytes transformation. Figure 8 shows the general form of 1D CA based substitute byte transformation.



EA	04	65	85
83	45	5D	96
5C	33	98	B0
F0	2D	AD	C5

87	F2	4D	97
EC	6E	4C	90
4A	C3	46	E7
8C	D8	95	A6

Figure 7: 1D CA based (Rule-30) sub bytes transformation

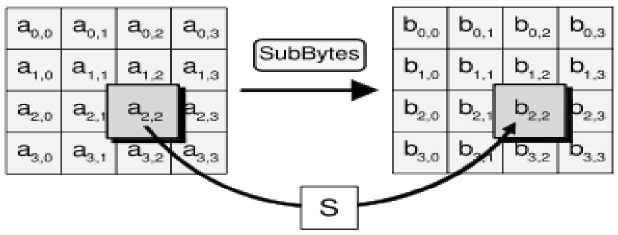

$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
SubBytes
$b_{0,0}$ $b_{0,1}$ $b_{0,2}$ $b_{0,3}$
$b_{1,0}$ $b_{1,1}$ $b_{1,2}$ $b_{1,3}$
$b_{2,0}$ $b_{2,1}$ $b_{2,2}$ $b_{2,3}$
$b_{3,0}$ $b_{3,1}$ $b_{3,2}$ $b_{3,3}$
S

Figure 8: General form of 1D CA based substitution byte (rule-30) transformation

The inverse transformation can be obtained by using the complement of rule-30. Since the complement of rule-30 is the rule-83, we can decrypt the message using rule-83, which is called as inverse transformation. Thus we can encrypt the message using rule-30 and decrypt the message using the rule-83.

4.3.4 PB CA rule128 and PB CA rule32

Now the output of S-Box is fed to PB CA rule128 and PB CA rule32 (Figure 9) consecutively so that permutation (transposition) operations are performed providing the linear rules of cellular automata and hence diffusion property of cryptography.

For example, in the following matrix, applying the PB CA rule128 to each cell of the matrix, the values in the first row have been shifted to second row, values in the second row have been shifted to third row and so on and values in the last row have been shifted to first row after applying periodic boundary (PB) CA rule-128 to each value in the cell of the matrix.

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-128}} \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

For example, in the following matrix, applying PB CA rule 32 to each cell, the values in the first column have been shifted to second column, second column values have been shifted to third column and third column values have been shifted to first column, and so on.

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \xrightarrow{\text{PB CA rule-32}} \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

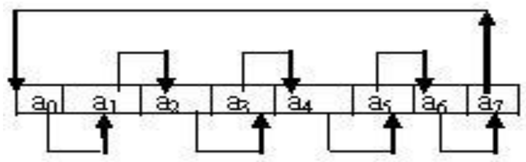


Figure 9: CA rule-32 Periodic Boundary on each cell of plaintext

1.6.5 1D CA based Substitute Bytes Transformation(S-Box): After the operation of PB CA rule128 and PB CA rule32, we fed the data to 1D CA based S-Box(rule-30) to get the final cipher text of original plain text.

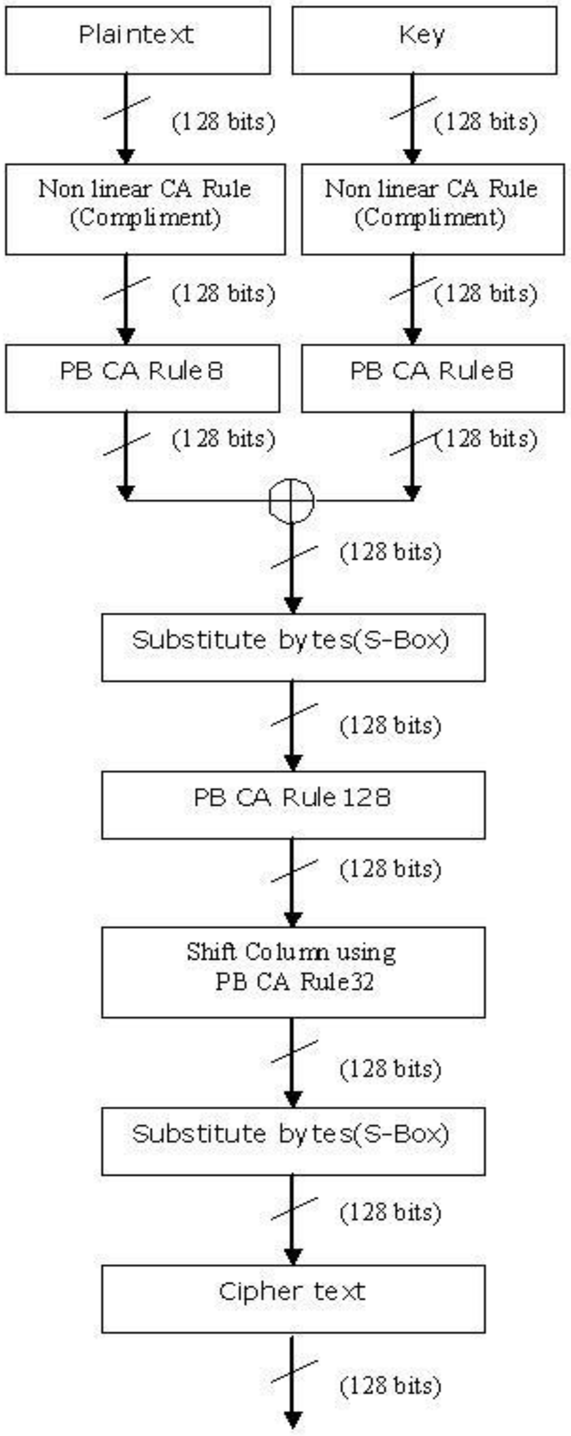
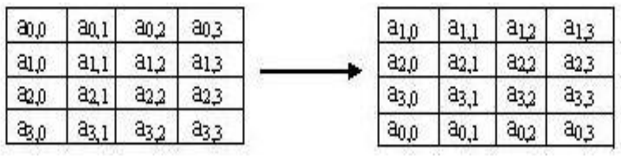


Figure 10: one-round of encryption algorithm applying CA rule

1.7 Decryption algorithm

Since non linear CA rule (compliment), PB CA rule8 (linear), XOR (linear), S-Box (non linear), PB CA rule128 (linear) , PB CA rule 32 (linear) and 1D CA rule-30 are reversible, we can decrypt the cipher text to plain text in reverse way. Here the inverses of compliment, XOR and 1D CA based S-Box(rule-30) are compliment, XOR and inverse CA S-Box(rule-83) respectively. Also the inverses of PB CA rule 8, PB CA rule128 and PB CA rule 32 are PB CA rule 128; PB CA rule 8 and PB CA rule 2 respectively.



$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$

Figure 11: Periodic Boundary CA rule-8

For example, in the following matrix, the values in the second row have been shifted to first row, values in the third row have been shifted to second row and so on and values in the first row have been shifted to last row after applying periodic boundary (PB) CA rule-8 to each value in the cell of the matrix.

$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} \xrightarrow{\text{PB CA rule-8}} \begin{pmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{pmatrix}$$

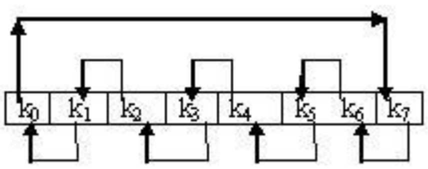


Figure 12: Shifting of rows by PB CA rule-2

Figure 12 shows the use of CA rule-2 Periodic Boundary on each cell of key. For example, in the following matrix, the values in the second column have been shifted to first column, values in the third column have been shifted to second column and so on and values in the first column have been shifted to last column after applying periodic boundary (PB) CA rule-2 to each value in the cell of the matrix.

$$\begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{pmatrix} \xrightarrow{\text{PB CA rule-2}} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

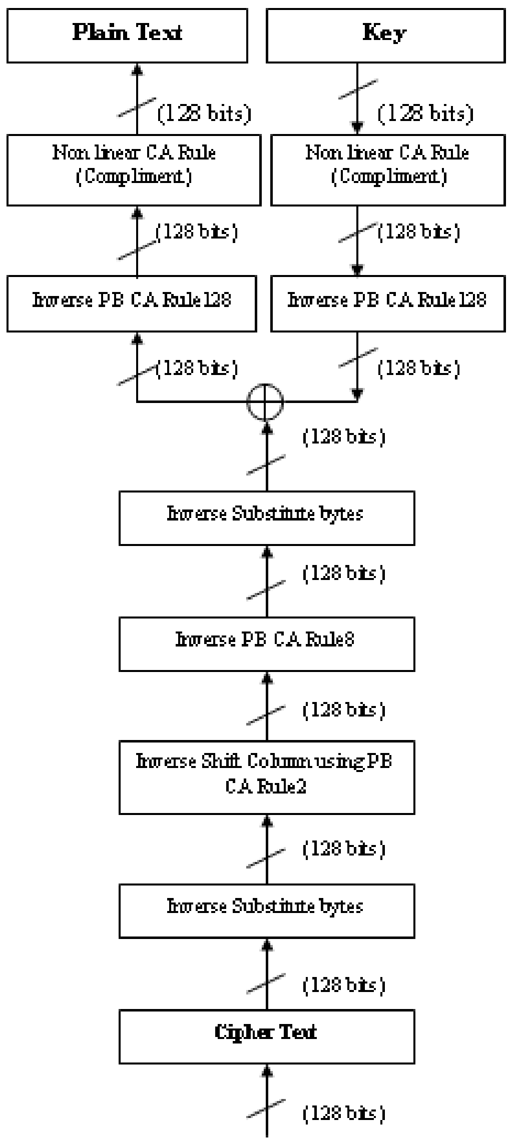


Figure 13: One-round of Decryption algorithm applying CA rule

V. CONCLUSION AND FUTURE WORK

Our algorithm provides both linear as well as non-linear Boolean functions with the help of linear and non- linear rules of cellular automata so that the diffusion as well as confusion property of the cryptography is achieved. This algorithm is more secure than DES because the key is used in the beginning of algorithm unlike it is in DES at a later stage and the number of rounds in DES is 16 whereas it is 8 in our algorithm. This algorithm is also more secure than AES because the number of non linear functions in our algorithm is more and the 1D cellular automata rule30 provides randomness and the number of rounds in our algorithm is 8 whereas it is 10 in AES.

Our algorithm, being based on concept of CA, helps parallel processing of text and more randomness due to rule-30. Besides, due to availability of chip level design cellular automata machine (CAM), our algorithm can encrypt and decrypt the text at very high speed in the order of nano seconds.

Since 1D CA based S-Box using rule-30 provides only one of the criteria of cryptography (i.e. randomness or balanced ness), we have planned to develop a new cellular automata based substitute box (CA based S-Box) using any other non linear cellular automata rule (other than rule-30) so that it will satisfy all the cryptographic properties which will be more secure than that of our proposed algorithm.

REFERENCES

[1.] Claude Carlet,"Boolean functions for cryptography and error correcting codes". PhD Thesis.

[2.] Stephen Wolfram, "Cryptography with cellular automata". Lecture Notes in Computer Science, 218 (Springer-Verlag, 1986) , pages 429-432, 1986.

[3.] Subhamoy Maitra and Enes Pasalic, "Further Constructions of Resilient Boolean Functions With Very High Nonlinearity".*IEEE Transactions on Information Theory*, Vol. 48(7), July 2002.

[4.] William Stallings and Lawrie Brown, *Computer Security: Principles and Practice*, Pearson Education Inc., New Delhi.

[5.] Charles P. Pfleeger and Shari Lawrence Pfleeger, *Security in Computing*. Pearson Education Inc., New Delhi.

[6.] XIA Xuewen, LI Yuanxiang, XIA Zhuliang, and WANG Rong, "Data Encryption Based on Multi-Granularity Reversible Cellular Automata". Proceedings of *International Conference on Computational Intelligence and Security*, 2009, pages: 192-196.

[7.] Debasis Das and Abhishek Ray, "A Parallel Encryption Algorithm for Block Ciphers Based on Reversible Programmable Cellular Automata". J*ournal of Computer Science and Engineering*, Volume 1, Issue 1, May 2010, pages: 82-90.

[8.] Anirban Kundu, Alok Ranjan Pal, Tanay Sarkar, Moutan Banerjee, Sutirtha Kr. Guha, and Debajyoti Mukhopadhyay, "Comparative Study on Null Boundary and Periodic Boundary 3-Neighborhood Multiple Attractor Cellular Automata for Classification". Proceedings of *third International Conference on Digital Information Management, ICDIM 2008*, pages: 204-209.

[9.] Irfan Siap, Hasan Akin, Ferhat Sah, "Garden of eden configurations for 2-D cellular automata with rule 2460 N. *Information Sciences*, Volume 180, Issue 18, 15 September 2010, Pages 3562-357.

[10.] Somanath Tripathy and Sukumar Nandi, "LCASE: Lightweight Cellular Automata-based Symmetric-key Encryption". *International Journal of Network Security*, Vol.8, No.2, Pages: 243-252, Mar. 2009.

[11.] Pabitra Pal Choudhury, Sudhakar Sahoo and Mithun Chakraborty,"Characterization of the Evolution of Nonlinear Uniform Cellular Automata in the Light of Deviated States".International Journal of Mathemeatics and Mathematical Sciences, Vol10 1155/2011,605098.

[12.] Dustin gage, Elizabeth Laub, Briana Mcgarthy. "CELLULAR AUTOMATA: IS RULE 30 RANDOM?"