International Journal of Communication Networks and Security

Volume 1 | Issue 1 Article 10

April 2011

Equivalence of DES and AES Algorithm with Cellular Automata

Sambhu Prasad Panda
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, ambhu.prasad.panda@gmail.com

Madhusmita Sahu
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, madhu_sahu@yahoo.com

Umesh Prasad Rout
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, umesh.upr@gmail.com

Surendra Kumar Nanda
C V Raman Computer Academy, Bidyanagar, Mahura, Janla Bhubaneswar-752054, Orissa, India, situnanda@gmail.com

Equivalence of DES and AES Algorithm with Cellular Automata

Sambhu Prasad Panda, Madhusmita Sahu, Umesh Prasad Rout and Surendra Kumar Nanda
Department of MCA
C V Raman Computer Academy, Bidyanagar, Mahura, Janla
Bhubaneswar-752054, Orissa, India
sambhu.prasad.panda@gmail.com, madhu_sahu@yahoo.com, umesh.upr@gmail.com, situnanda@gmail.com

Abstract— ***In this paper we present the equivalence of the operations involved in DES and AES algorithm with operations of cellular automata. We identify all the permutation and substitution operations involved in DES and AES algorithm and compare these operations with the cellular automata rules. Then we find that permutation operations involved in DES and AES are equivalent to linear cellular automata rules providing diffusion property of cryptography whereas substitution operations involved in DES and AES are equivalent to non linear cellular automata rules providing the confusion property of cryptography. Hence instead of using operations involved in DES and AES algorithm, we can apply linear as well as non-linear cellular automata rules in cryptography for better security and parallel processing.***

Keywords-Cellular Automaton, DES, AES, Cipher text, Plain text, Boolean function, Affine function, Periodic boundary, Null boundary, Hybrid cellular automata, Uniform cellular automata

I. INTRODUCTION

Cellular automata is an idealized parallel processing machine, which is an array (1-D, 2-D, 3-D) of numbers or symbols called cell values together with an updating rule. A cell value is updated based on this updating rule, which involves the cell value as well as other cell values in a particular neighborhood. Instead of applying substitution and permutation operations involved in DES and AES algorithm, we can use linear and non-linear rules of cellular automata respectively. The cellular automata rules can be executed in parallel and hence the execution speed is faster than the operations involved in DES and AES algorithm.

The organization of this paper is as follows. Section II introduces the Boolean functions and its criteria. In Section III, we discuss the concept of Cellular Automata. Section IV discusses some related works. In Section V we discuss some important theorems. Section VI and VII describe the equivalence of operations involved in DES and AES algorithm with cellular automata for achieving security in cryptography.

II. BACKGROUND

A. 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^n are nonlinear Boolean functions.

For example: - n=2,

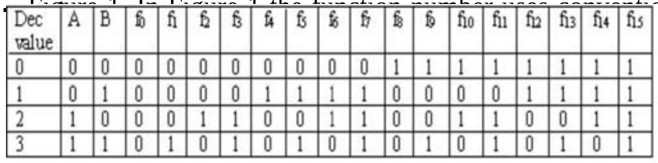
Then $f:\{0,1\}^2 \quad \{0,1\}$

This implies $f:\{0,1\} \times \{0,1\} \quad \{0,1\}$

And $f:\{00, 01, 10, 11\} \quad \{0,1\}$

Then the total numbers of possible functions are shown in Figure 1. In Figure 1 the function number uses convention, which is the decimal value of the output binary column vector from top to bottom.

Then the total numbers of possible functions are shown in
on,
tor



Dec value	A	B	f0	f1	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12	f13	f14	f15
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

Thus $2^{2^n} = 2^4 = 16$ boolean functions in two-variables appears.

The Boolean functions are classified into two categories

- 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 Boolean functions, out of which 2^n linear Boolean functions and rest are non-linear Boolean functions.

From Figure 1, it is clear that out of 16(=2^4) Boolean functions (f_0, f_1, f_2, …, f_{15}) in 2 variables (A, B), 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}).

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.

A. Computation for linear, affine and nonlinearity of Boolean functions

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

f :$\{0,1\}^n$ {0,1}. There are Boolean functions.

For example for n=2, we have

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

=> f: {0,1}x {0,1}   {0,1}

=> f: {00,01,10,11 }   {0,1}

Here we have 16 Boolean functions (as shown in figure1) and we have to identify which functions are linear, which are affine, and which are non linear and what is the value of non linearity. We know the linear properity of Boolean functions 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)}

(Using Figure 1)

And f_3= (0, 0, 1, 1)^T (using Figure 1)

f (X+Y)= f (X) + f(Y) where X = (0,0), Y = (0,1) and

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

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

RHS = f (X) = f (0,0) = 0,   f (Y) = f (0,1) =0

f (X) + f (Y) = 0 +0 = 0

Therefore LHS = RHS and so f3 is a linear function. In this way we can find all the linear boolean functions.Now, in order to get all affine functions , we perform the exclusive-OR on a 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

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 e 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.

A. 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.

I. CELLULAR AUTOMATA (CA)

Cellular Automata (CA) were originally conceived by Ulman and Von Neumann in the 1940. CA provides a formal framework for investigating the behavior of complex and extended system. Cellular automata are simple mathematical idealizations of natural system. This is an idealized parallel processing machine, which is an array (1-D, 2-D, 3-D) of numbers or symbols called cell values together with an updating rule. A cell value is updated based on this updating rule, which involves the cell value as well as other cell values in a particular neighborhood. Figure 2 and Figure 3 show the structure of one and two dimensional cellular automata neighborhood cells respectively.

Figure 2: 1-D neighborhood

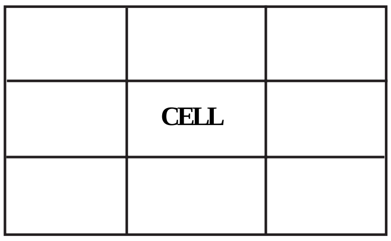


Figure 3: 2-D neighborhood

A. Definitions

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

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

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

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

B. CA rules as Boolean functions

Two-dimensional cellular automaton consists of an infinite (or finite) grid of cells, each in one of a finite number of states. Time is discrete and the state of a cell at time t is a function of the states of its neighbors at time t-1. For cryptographic application 8-neighborhood CA rules are newly introduced. Table 1 shows all the rules of two dimensional cellular automata.

64	128	256
32	1	2
16	8	4

Table I: 8-neighborhood CA rules

In 2-D eight neighborhoods CA the next state of a particular cell is affected by the current state of itself and seven cells in its nearest neighborhood (as shown in Table 1). Such dependencies are accounted by various rules. The central cell represents the current cell (i.e. the cell being considered) and all other cells represent the seven 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. No change operation can be performed by CA rule1, which we call as linear rule of cellular automata.Circular left shift or column shift operation can be performed by using PB CA rule2 performing the linear rules of cellular automata providing diffusion (permutation) property of cryptography. Similarly PB CA rule32 can be performed to get the circular right shift operation providing the diffusion property of cryptography. PB CA rule 2 and PB CA rule 32 are inverses of each other. Hence if we take PB CA rule2 operation for encryption then PB CA rule 32 operation will be used for decryption and vice versa. Similarly PB CA rule 8 and PB CA rule 128 are used for circular row shift upwards or downwords and these two operations are linear rules of CA providing the diffusion property of cryptography. PB CA rule 8 and PB CA rule128 are inverses to each other and so when one rule is used for encryption then other will be used for decryption. Hence PB CA rule 2 , 32, 8 and 128 can be used for encryption and decryption in cryptography for security of plaintexts.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.

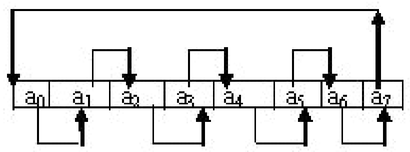


Figure 4: PB CA rule-32 on each cell of plaintext

Figure 4 shows the use of PB CA rule-32 on each cell of plaintext.

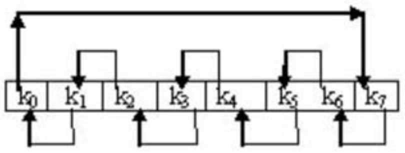


Figure 5: PB CA rule-2 on each cell

Figure 5 shows the use of CA rule-2 Periodic Boundary on each cell. Figure 6 shows the use of Periodic Boundary CA rule-8 to the matrix.

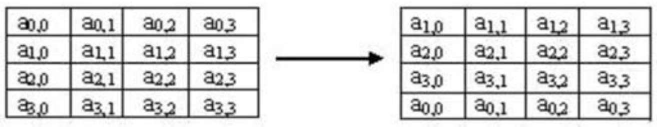

$a_{0,0}$ $a_{0,1}$ $a_{0,2}$ $a_{0,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$ $a_{1,3}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$ $a_{2,3}$
$a_{3,0}$ $a_{3,1}$ $a_{3,2}$ $a_{3,3}$
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$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

Figure 6: Periodic Boundary CA rule-8

I. RELATED WORK

Carlet [1] performed study on Boolean functions for cryptography. He found all the criteria for cryptography. 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 speciûcation 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 conûrm with the Advanced Encryption Standard (AES) speciûcation. All these works were based on Boolean functions and its applications in cellular automata in cryptography in stream cipher. Also very few works have been carried out on two dimensional cellular automata for block cipher [9].

V. THEOREMS

Theorem 1: Every permutation function is linear providing linear rules of cellular automata helps to provide diffusion property in cryptography and therefore the initial permutation(IP), expansion permutation(EP),permutation box (P-Box), permutation choice1(PC1),permutation choice2(PC2), circular left shift(CLS) and final permutation(FP) functions which are equivalent to linear rules of cellular automata provides diffusion property of cryptography and can be used for encryption and decryption algorithms.

Theorem 2: XOR, PB CArule2 (or PB CA rule32), PB CA rule8 (or PB CA rule128) are all reversible and linear rules of cellular automata which can be used for encryption and decryption providing diffusion property of cryptography.

Theorem 3: Output Boolean functions are always balanced, non linear, high algebraic degree and high co-relation immune in case of S-Box of AES and DES providing confusion property of cryptography and hence used for encryption and decryption.

VI. STEPS IN EQUIVALENCE OF DES ALGORITHM WITH CELLULAR AUTOMATA

A. Initial permutation (IP)

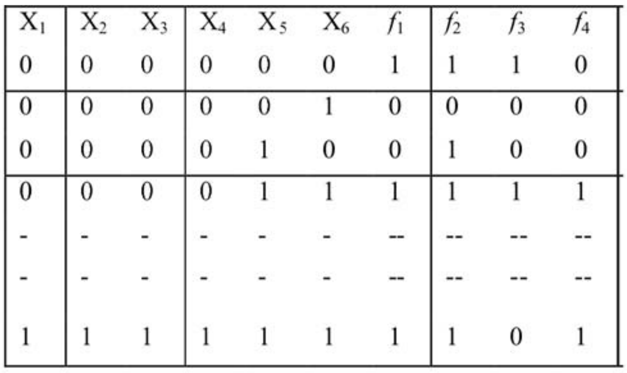
In initial permutation, 64 bits of plain text changes their bit position as per the initial permutation table of DES algorithm. So 64 Boolean functions ($f_{0,}$ $f_{1,}$ $f_{2,\_,}$ f_{63}) on 64 variables ($x_{0,}$ $x_{1,\_}$ $_{,}$ x_{63}) are involved, which are found to be linear according to the following rules of cellular automata. f ($x_{0,}$ $x_{1,\_,}$ x_{63}) = x_i , where i = 0,1,2,—,63 (By Theorem).Here the numbers of input Boolean functions is same as the output Boolean functions but only the positions are changed. But the output Boolean functions are balanced. Again the output Boolean functions are not high co-relation immunity. Those output Boolean functions are not high algebraic degree. Output Boolean functions also provide diffusion .One conclusion is drawn from the above observation is that every permutation function is linear.

B. Expansion Permutation (EP)

In this step, we obtain 48 bits of output Boolean functions from 32 bits of right plaintext, as per the expansion permutation table of DES algorithm. From the EP table it is clear that all the 32 bit input variables are present in the output, but some of the input bits are repeated in output Boolean functions. Hence 48 Boolean functions on 32 variables are involved satisfying the linear rule: f (x1,x2,x3,…..,x32)=xi, where i=1,2,3,………,32. Thus expansion permutation function is linear, providing diffusion property of cryptography.

C. S-Box (Substitution Box)

S-Box substitution is a process that accepts the 48-bit input, after the XOR operation performed between expansion permutation(EP) (or expanded right plain text(RPT) of 48 bits) and Permutation choice2(PC2)(or contracted key of 48 bits) and produces a 32 bits of output using the substitution technique provided by the data from 8 S-Boxes. The s-boxes are substitutions based on a table of 4 rows and 16 columns. Suppose that block B_j is the six bits $b_1b_2b_3b_4b_5b_6$. Bits b_1 and b_6, taken together, form a two-bit binary number b_1b_6, having a decimal value from 0 to 3. Call this value as r. Bits $b_2,$ $b_3,$ $b_4,$ b_5 taken together form a 4-bit binary number $b_2b_3b_4b_5,$ having a decimal value from 0 to 15. Call this value as c. The substitutions from the S-boxes transform each 6-bits of block B_i into the 4-bits of result shown in row r, column c (using S-box definitions). If we think it in terms of Boolean function then we can get all the four cryptographic criteria. If we take all possibilities of six-variable Boolean functions and pass it in S_1-box then we get four different functions. Table 2 shows the 6 inputs S-Box.



X_1	X_2	X_3	X_4	X_5	X_6	f_1	f_2	f_3	f_4
0	0	0	0	0	0	1	1	1	0
0	0	0	0	0	1	0	0	0	0
0	0	0	0	1	0	0	1	0	0
0	0	0	0	1	1	1	1	1	1
-	-	-	-	-	-	--	--	--	--
-	-	-	-	-	-	--	--	--	--
1	1	1	1	1	1	1	1	0	1

Table 2: S-Box

If we study these four functions f_1, f_2, f_3, f_4 we get all four cryptographic properties .Those four properties are **non-linear**, **balanced**, **high algebraic degree** and **co-relation immune**. But, in fact this non-linear transformation is the concatenation of 8 S-Boxes, which transform binary strings of 6 bits into 4 bit long ones.So,32 (that is,8*4) Boolean functions on 6 variables are involved.

Similarly if we pass the above six-variable Boolean functions in S_1, S_2, S_3, S_4, S_5, S_6, S_7 and S_8 then we get 32 different functions which are non-linear, balanced, high algebraic degree and co-relation immune.

A. Permutation Box(P-Box)

In this step, 32 bits of S-Box are permuted using a P-Box. Hence 32 Boolean functions on 32 variables are involved, which are linear. Because f(x1,x2,x3,……,x32)=xi, where i=1,2,3,….32.P-Box permutation provides diffusion property of cryptography.

B. Key Generation in DES

In key generation process, we take 64 bits of key and represent these bits in a 8X8 matrix. Then we discard the 8th column of this 8X8 matrix so that each bit remains the same excepting the 8th column and then we get a matrix of order 8X7 of size 56 bits. . Since the positions of bits are not changed in this 8X7 matrix, we conclude that this operation is equivalent to rule1 of cellular automata. Next we apply the operation of permutation choice1(PC1), which involves the shifting of positions of bits in different cells of the matrix and we call it as the linear cellular automata rule .Now the key of size 56 bits is divided into two equal halves (left half and right half) of 28 bits each . We apply 16-rounds to each of left and right halves. Each 28-bit are left-shifted by one-unit in round no 1, 2, 9, and 16. Then we can call it as rule-2of cellular automata. In all other rounds, both the left and right plain texts are left shifted by 2-bits, which can be also obtained by applying rule-2 of CA twice. We know rule-2 is linear. Then both 28 bits of each halves are passed through the operation of permutation choice2 (PC2), so that from 56 bits 8 bits are discarded and now we get the key of size 48 bits. In a similar way as that of key discarding and PC1 we can argue that PC2 is the application of linear cellular automata rule. Hence we conclude that the operations involved in key generation process in each of 16-rounds are the applications of linear cellular automata rules providing the permutation or diffusion property of cryptography. Thus the operations involved in key generation process are equivalent to the operations of linear rules of cellular automata.

C. Final permutation (FP)

At the end of the 16 rounds, the final permutation (FP) is performed according to the FP table. This operation is a transposition (permutation) which is equivalent to the linear rule of cellular automata providing diffusion property of cryptography.

D. Decryption in DES algorithm

The steps of algorithm for decryption are same as that of steps for encryption. The values of the various tables and the operations as well as their sequence are so carefully chosen that the algorithm is reversible. As we know PB CA rule2 and PB CA rule32 are reversible to each other if one rule is taken for encryption other will be taken for decryption. Similarly PB CA rule8 and PB CArule128 are reversible to each other and can be used for encryption and decryption alternatively. These are few equivalent rules in cellular automata used for decryption.

VII. STEPS IN EQUIVALENCE OF AES ALGORITHM WITH CELLULAR AUTOMATA

A. Add round key (XOR of plain text with key)

A simple bitwise XOR of the current block with a portion of the expanded key (after key generation) is performed. XOR operation is equivalent to linear rule of cellular automata providing diffusion property of cryptography.

B. Byte Substitution (S-box in AES)

AES defines a 16 x 16 matrix of byte values, called an S-box that contains a permutation of all possible 256 8-bit values. Each individual byte of state is mapped into a new byte in the following way. The leftmost 4 bits of the byte are used as a row value and the rightmost 4-bits are used as column value. These row and column values serve as indexes into the S-box to select a unique 8-bits output value. If we take 8-variables of all possible Boolean functions and pass into the S-box then we get 256 no of different 8-bit values.That means we get 8-different Boolean functions.i.e. $f_{1,} f_{2,,} f_{3,} f_{4,} f_{5,} f_{6,} f_{7}$ and $f_{8..}$ If we study these Boolean functions we get all the four cryptographic properties like non-linearity, balanced ness, high algebraic degree and good co-relation immunity. S-Box provides the confusion property of cryptography which is equivalent to non linear cellular automata rule.

C. Shift rows

In shift rows, the first rows remain unchanged. So the output bits are same as input bits, providing the CA rule1 of cellular automata, which is linear providing linear Boolean functions of 32 bits. The second, third, and fourth rows are circularly left shifted by 1, 2, 3bits respectively, producing the input as the corresponding outputs. Only the positions are changed. Hence the outputs of 2nd, 3rd, and 4th rows are linear Boolean functions. These shift operation is same as rule-2 in cellular automata. That means PB CA Rule-2 is applied in row-2,PB CA rule-2 is applied two times in row3 and PB CA Rule-2 is applied three times in row4, providing diffusion property of cryptography. Hence the shift row operation of AES algorithm is equivalent to CA rule2 of cellular automata.

D. Mix Column

In this step, the 4-bytes of every column are mixed in linear function, performing linear transformation. Since we multiply a constant matrix, an affine function is obtained from it, which are equivalent to non linear Boolean functions of cellular automata providing confusion property of cryptography.

E. Decryption in AES algorithm

Since XOR, Shift rows, mix columns and S-Box are invertible we can easily decrypt the cipher text using XOR, inverse substitution bytes, inverse shift rows. But inverse shift rows can be done through PB CA rule32 which is the inverse of PB CA rule2. Thus PB CA rule 32 is equivalent to linear cellular automata rule corresponding shift row operations of AES.

CONCLUSION AND FUTURE WORK

Our paper deals with equivalence of operations involved in DES and AES algorithm with operations of cellular automata. Since already we have proved that the operations of DES and AES are equivalent to the operations of Cellular automata and Cellular automata can process the texts in parallel so that the encryption and decryption can be done at very high speed in the order of nano seconds, we conclude that we can apply the linear and non linear cellular automata rules to any block cipher for encryption and decryption. Besides, due to availability of chip level design of Cellular Automata Machine (CAM), we can encrypt and decrypt the text at very high speed in the order of nano seconds.

In future we have planned to categorize group of all the linear as well as non linear CA rules that are reversible to each other, which will be helpful for us in cryptography for encryption and decryption. Then only we can develop a new encryption and decryption algorithms by using only linear and non linear cellular automata rules, which may be better than DES or AES.

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.