International Journal of Mathematical Archive-3(2), 2012, Page: 362-372
Available online through www.ijma.info ISSN 2229 – 5046

A Stream Cipher Cryptosystem Based on Linear Feedback Shift Register

Sukalyan Som*[1], Saikat Ghosh[2]

[1]Department of Computer Science, Barrackpore Rastraguru Surendranath College, Kolkata, West Bengal, India
[2]Department of Information Technology, West Bengal University of Technology, Kolkata, West Bengal, India
E-mail: [1]sukalyan.s@gmail.com, [2]ghoshsaikat6@gmail.com

(Received on: 15-01-12; Accepted on: 06-02-12)

ABSTRACT

Cryptography is considered to be a disciple of science of achieving security by converting sensitive information to an un-interpretable form such that it cannot be interpreted by anyone except the transmitter and intended recipient. An innumerable set of cryptographic schemes persist in which each of it has its own affirmative and feeble characteristics. In this paper we present a Stream Cipher Cryptosystem based on Linear Feedback Shift Register where our aim is to encipher Plain texts depending on some parameters, which are continuously varied with every run-time so that the cipher text becomes different for similar entered texts every runtime.

Keywords-- *Stream Cipher, Linear Feedback Shift Register, Encryption, Decryption, Testing of Hypothesis, Correlation Immune.*

1. INTRODUCTION:

Cryptography, a word with Greek origin, means "Secret writing", defined to be a set of techniques and study of mathematics related to aspects of information security such as confidentiality, authenticity, integrity, non-repudiation.[1] Cryptanalysis refers to analyzing and breaking secret writings and Cryptology is the combination of both.[2] Cryptanalytic attacks can be cipher text only, known plaintext, chosen plaintext, chosen cipher text, adaptive chosen plaintext, brute force attack, key guessing attack etc [3].

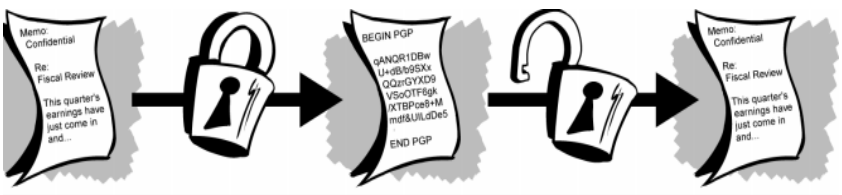
Plaintext or Cleartext signifies a message that can be understood by the sender, the recipient and also by anyone else who gets an access to that message.

Cipher refers to the algorithm(s) for transforming an intelligible message to unintelligible form.

When a plain text message is codified using any suitable scheme, the resulting message is known as Ciphertext.

A key is a number (or a set of numbers) that the cipher, as an algorithm, operates on.

The method of disguising plaintext in such a way as to hide its substance is called encryption. Decryption is the process of reverting ciphertext to its original plaintext.



Plaintext — Encryption (Encryption Cipher + Encryption Key) — Cipher text — Decryption (Decryption Cipher + Decryption Key) — Plaintext

Figure1: Encryption and Decryption

**Corresponding author: Sukalyan Som*, *E-mail: sukalyan.s@gmail.com*

As depicted in Figure 1, to encrypt a message, an encryption cipher, an encryption key and plaintext is needed which creates the ciphertext. To decrypt a message a decryption cipher, a decryption key and the cipher text is needed which reveals the original plaintext.

Innumerable algorithms have been developed over the years to provide security of information but each of them has some merits and demerits. No single algorithm is sufficient on its own for this purpose. As a result researchers are constantly devoting themselves in the field of cryptography to eliminate the deficiency and find a better solution.

In this paper an effort has been made to develop a stream cipher based cryptosystem. Our aim is to encipher English texts depending on some parameters, which are continuously varied with every run-time so that the cipher text becomes different for similar entered texts every runtime. The parameters for enciphering are as follows:

- The number of Linear Feedback Shift Registers;
- The length of Linear Feedback Shift Registers;
- The connection polynomials for each Linear Feedback Shift Registers;
- Initial condition for each Linear Feedback Shift Registers and
- Combining function of the Linear Feedback Shift Registers;

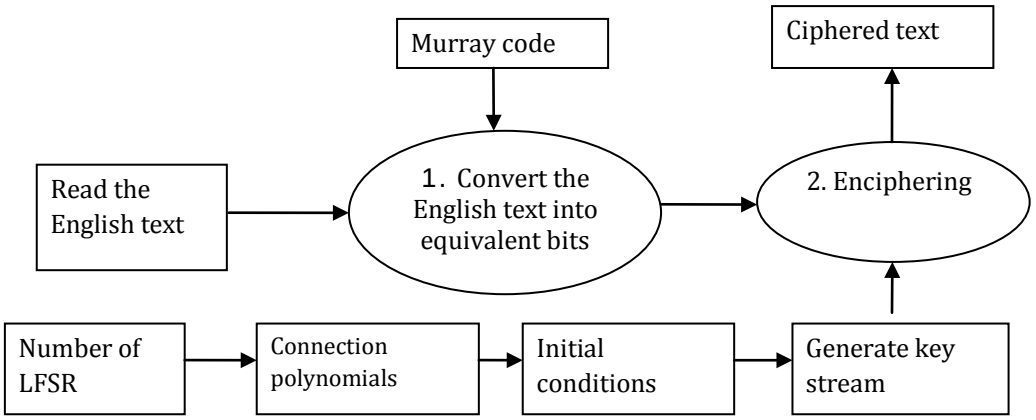


Figure 2: The proposed goal of the system

As depicted in Figure 1 if these parameters are varied for each runtime then the key stream generated every time will be different, the cipher text will become different. A bank of LFSRs is employed in the work, which is able to produce a pseudorandom sequence. The output of the individual LFSRs is combined by means of functions to break the linearity property of the LFSRs. The English text is first converted into a stream of bits for each character based on the Murray code pool. Then these generated bits are combined with key stream to generate the encrypted version of the entered English text. To get back entered English message the five parameters are again have to be entered correctly otherwise the output will be an erroneous one. The bank of LFSR employing the similar combining functions will produce a key stream and combined with the encoded message to produce the message bit stream. Now it is compared with the Murray code pool to produce the desired result.

In section 2 we have dealt with the fundamental theories and basic terminologies. Section 3 depicts the proposed technique. Experimental results are given in section 4. Testing and analysis has been done in section 5 and conclusions are drawn in section 6.

2. PROPOSED SCHEME:

Firstly, we fabricate the LFSRs algorithm. Secondly, the English text message to bits conversion algorithm is designed. Finally, the combining function for the bank of LFSRs and the function for the enciphering process are chosen and implemented. Then the encrypted cipher text is obtained. Similarly the decryption process can also be implemented.

Implementation of LFSR

The implementation of the bank of LFSRs is a common phase of designing for both encryption and decryption process. The LFSR is implemented by means of the following steps mentioned below

- In this phase, the program takes explicitly the number of LFSRs for making the bank so that the linearity property of the LFSRs is cracked.
- The length of each LFSR is allotted for each LFSR in every run time.

© 2012, IJMA. All Rights Reserved

- The connections polynomials are specify the tapping for each LFSR.
- Finally, the initial conditions are accepted.
- The combing function for the bank of LFSR is chosen as EX-OR.

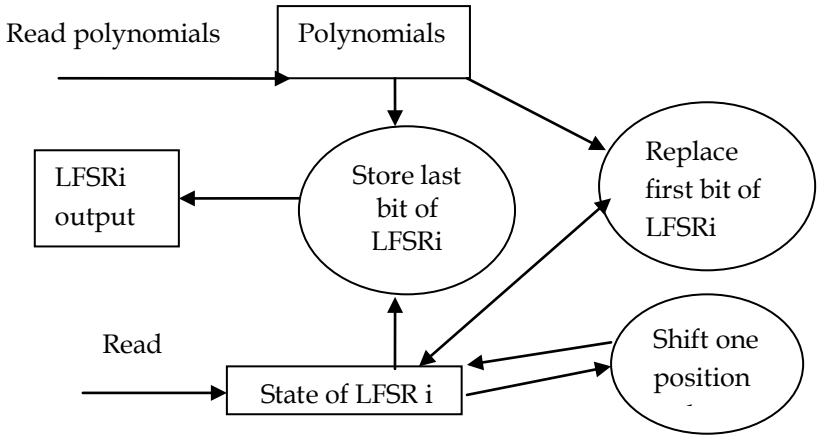


Figure 3: The behavior of ith LFSR

Encryption Process

Generation of Encoded Message

Module 1: Generation of encoded message from English text message

Input : English text in uppercase having no white space contained in file eng.txt. Characters with their Corresponding 5-bit codes supported by Murray code contained in code itself.

Output : Encoded message file **encode.txt**
Values of $P(M_i=0)$ and value of $P(M_i=M_{i-1})$
Values of $P(M_j=0)$ and value of $P(M_j=M_{j-2})$

Module 2: Generation of random bit sequence with certain value of $P(M_i=0)$

Input : Length of random bit sequence value of $P(M_i=0)$

Output : Random bit contained in sequence
Values of $P(M_i=0)$ and value of $P(M_i=M_{i-1})$
Values of $P(M_j=0)$ and value of $P(M_j=M_{j-2})$

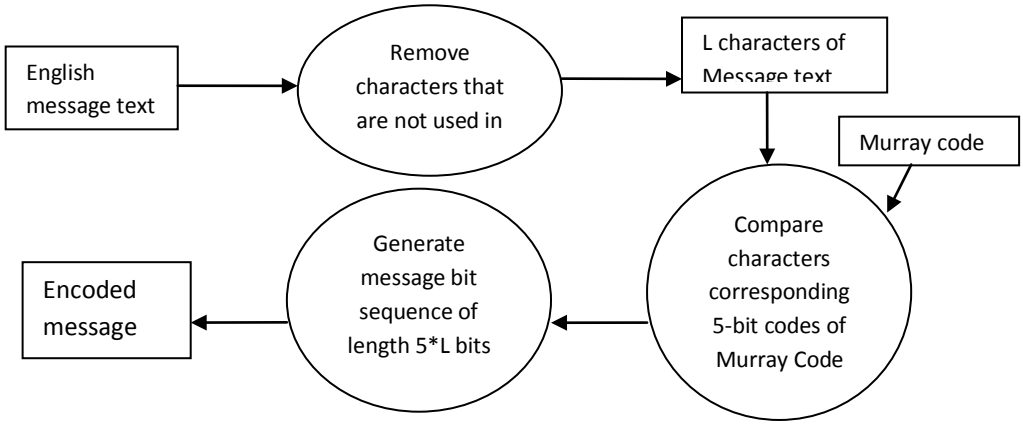


Figure 4: How message bits are generated from English text message

© 2012, IJMA. All Rights Reserved

Generation of Stream Cipher Text

The key stream generated from the bank of LFSRs and the encoded version of the English test is the major two ingredients of the stream cipher text. The encoded message is EX-OR-ed bit wise with the generated key stream to produce the cipher text.

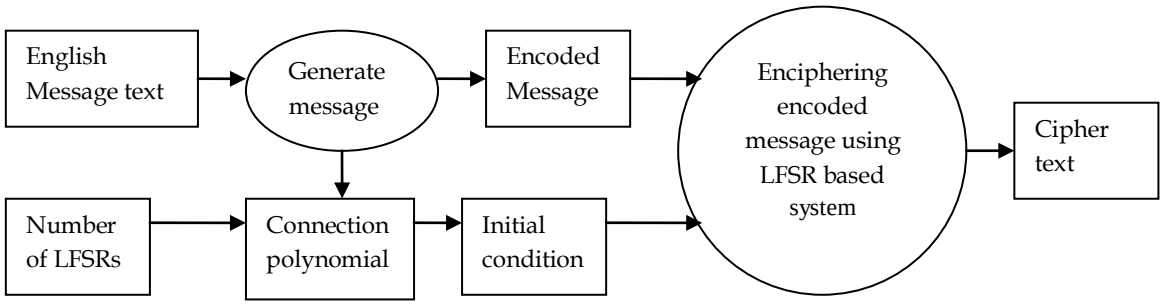


Figure 5: Generation of Cipher text

Decryption Process

Now in this module, we have wished to regenerate the entered English text message from the cipher text. This procedure is called the deciphering mechanism. Here we have used the same set of LFSR bank and the similar set of initial parameters to generate the same key stream. This key stream is then bit wise EX-OR-ed with the cipher text to regain the original message text.

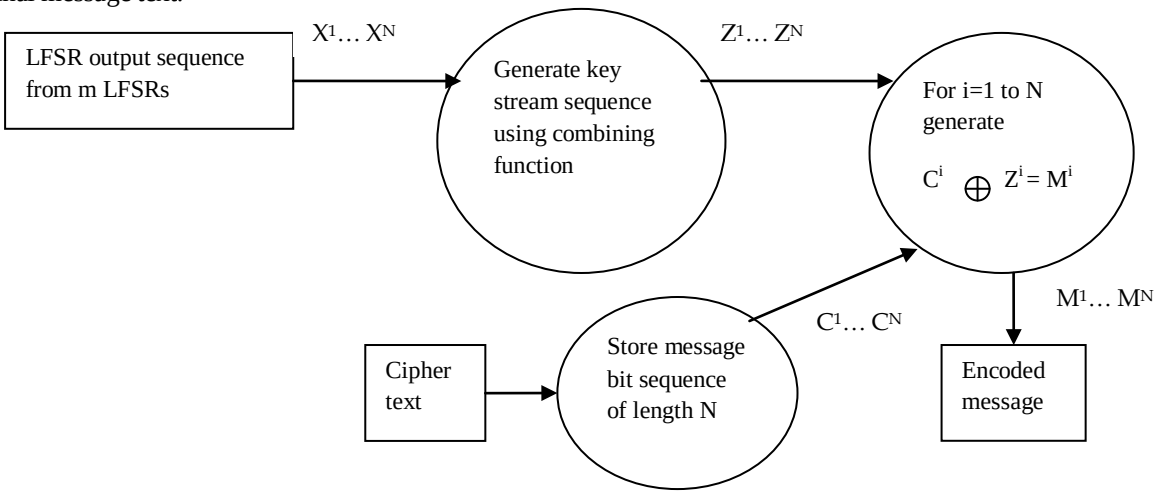


Figure 6: Decryption Process

3. EXPERIMENTAL WORK:

In this section we present an experimental run of the programs that have been developed for the above proposed scheme.

Implementation of variable length LFSR
Enter the no of LFSR: 5
Enter the size of the 1st LFSR: 4
Enter the initial condition of the 1st LFSR: 18
Enter the size of the 2nd LFSR: 5
Enter the initial condition of the 2nd LFSR: 25
Enter the size of the 3rd LFSR: 7
Enter the initial condition of the 3rd LFSR: 66
Enter the size of the 4th LFSR: 6
Enter the initial condition of the 4th LFSR: 62
Enter the size of the 5th LFSR: 10
Enter the initial condition of the 5th LFSR: 74
Creating the LFSR with their initial conditions
0010
11001
1000010
111110

© 2012, IJMA. All Rights Reserved

0001001010

Encryption:

Enter English text now (Strictly in Capital Case without Space)
SAIKATGHOSH
Enter the size of the 1st LFSR: 7

Enter the size of the 2nd LFSR: 5
Enter the initial condition of the 1st LFSR: 65
Enter the initial condition of the 2nd LFSR: 30
Enter the connection polynomial: 1 3
The encrypted data code for your entry is:
00101010011110101111010011000011010101001001000010110100

Decryption:

The encrypted data stream that we obtain:
00101010011110101111010011000011010101001001000010110100
Enter the size of the 1st LFSR: 7
Enter the size of the 2nd LFSR: 5
Enter the initial condition of the 1st LFSR: 65
Enter the initial condition of the 2nd LFSR: 30
Enter the connection polinomial: 1 3
The retrieved plain-text is:
SAIKATGHOSH

Output sequence of an LFSR: Consider the LFSR < 4; 1 + D + D4 > , if the initial state of the LFSR is [0, 0, 0, 0], the output sequence is the zero sequence. The following tables show the contents of the stages D3, D2, D1, D0 at the end of each unit of time t when the initial state is [0; 1; 1; 0].

t	D3	D2	D1	D0
1	0	1	1	0
2	0	0	1	1
3	1	0	0	1
4	0	1	0	0
5	0	0	1	0
6	0	0	0	1
7	1	0	0	0
8	1	1	0	0
9	1	1	1	0
10	1	1	1	1
11	0	1	1	1
12	1	0	1	1
13	0	1	0	1
14	1	0	1	0
15	1	1	0	1
16	0	1	1	0

Output sequence of the LFSR < 4; 1 + D + D4 > with initial condition [0; 1; 1; 0] is s = 0; 1; 1; 0; 0; 1; 0; 0; 0; 1; 1; 1; 1; 0; 1; ….. and is periodic with period 15.

4. TESTING AND ANALYSIS:

The statistical approach is the only way to determine the reliability of the immunity of the coding from the various kinds of attacks.

Statistical analysis when attempt is to determine initial condition:
Consider the following:

i) $C_t = Z_t$ if $M_t = 0$
$C_t \neq Z_t$ if $M_t = 1$

© 2012, IJMA. All Rights Reserved

ii) If $M_t = 0$ then
$C_t = X_{ti}$ if $Z_t = X_t$
$C_t \neq X_{ti}$ if $Z_t \neq X_t$

iii) If $M_t \neq 0$ then
$C_t = X_{ti}$ if $Z_t \neq X_t$
$C_t \neq X_{ti}$ if $Z_t = X_t$

Let $X_i = (X_{i1} , X_{i2} ,….., X_{iN})$ be the output sequence from i-th LFSR and di be the length $Z = (Z_1 , Z_2 ,……, Z_N)$ and $C= (C_1 , C_2 ,…, C_N)$ which have been said earlier. From i), ii) and iii) , $P (C = X_i) = P (Z = X_i) P (M = 0) + P (Z \neq X_t)$

$P(M \neq 0)$. For all practical coding schemes, $P (M = 0) \neq 0.5$. When the correct initial condition is used to generate the LFSR output sequence X_i from LFSRi for which $P (Z = X_i) \neq 0.5$, then $P (C = X_i) \neq 0.5$. In contrast, for a wrong initial condition, the sequence X_i is random and uncorrelated with C which implies that $P (C = X_i) = 0.5$. Since initial conditions of all LFSR are unknown, for each LFSR we have to consider all possible nonzero initial conditions of which only one is correct. Therefore for i-th LFSR we may have to consider the initial conditions from 1 to $2d_i – 1$. $2d_i – 1$ possible Xi sequences may have to be generated. Thus from an input sequence Xi and the cipher text C, we can determine whether the corresponding initial condition is correct by testing following hypotheses:

H_1: $P (C = X_i) = 0.5$ H_2: $P (C = X_i) \neq 0.$

The statistic used for testing H1against the alternative H2 is $(Z_1, Z_2… Z_N)$. For each nonzero initial condition of i-th LFSR, we check the value of $P (C = X_i)$. The initial condition, for which it is away from 0.5, is suspected to be a feasible initial condition for the i-th LFSR. If $P (C = X_i) = 0.5$, then the variables C and Xi are independent and the corresponding initial condition will not be feasible. What we need to consider is the dependence between C and Xi. We have

$$P (C = X_i) = \delta = P (Z = X_i) P (M = 0) + P (Z \neq X_t) P (M \neq 0)$$

Given the value of $P (Z = X_i)$ and $P (M = 0)$, it is possible to calculate δ as follows:
Say, $P (M = 0) = p$ and $P (Z = X_i) = q$. Then $\delta = q*p + (1-q) * (1-p)$ if $p \neq 0.5$ or $q \neq 0.5$ then $\delta \neq 0.5$ $P (C = X_i) \neq 0.5$. But if p=0.5 then $\delta = 0.5(q+1-p) = 0.5$. Same as if q = 0.5 then $\delta = 0.5(p+1-p) = 0.5$. Then $P (C = X_i)$ should be 0.5. So we see if q=0.5 for all LFSR, this method fails because $P (C = X_i)$ is not always from 0.5 for all LFSRs. That is the method fails due to the independence of the key stream X_i and the cipher text C. It would depend on the nature of the combining function and cipher length.

Attempt to determine the initial condition of LFSR

The attempts to determine the initial condition of the LFSRs are further subdivided into three modules for three types pf system i) Memory less, ii) One-memory, iii) Two-memories. These modules are shown in the program structure given below:

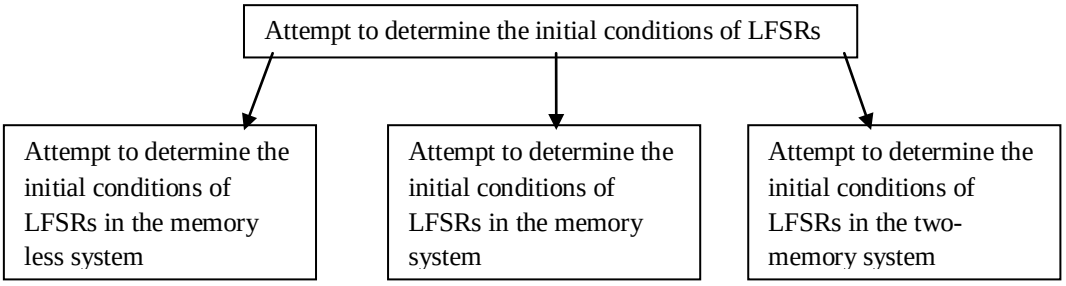


Figure 7: structure for design steps of attempt to determine initial condition of LFSRs in three types of system

For each of above modules, we try to find out that for which initial condition, output of an LFSR are correlated with given cipher text that is not 0.5 and reasonably away from 0.5 which is described earlier. To do this, at first we introduce a term “Correlation Immunity of a Boolean function”. The definition of “Correlation Immunity of a Boolean function”, a Boolean function f of m input variables X_1, $X_{2…}X_m$ with $Z = f(X_1, X_2, ..X_m)$ is defined as l-th order ($1 \leq l \leq m-1$) correlation immune if $P(Z = X_i | X_{i1}= cond_{i1}, X_{i2}= cond_{i2} ,….., X_{i1-1}= cond_{i1-1}) = 0.5$

For all possible choices of m distinct variables $i_1, i_2… i_{l-1}$ is distinct set of indices.
$X_{i1}, X_{i2}… X_{i1-1}, X_{il} € \{ X_1, X_2, ..X_m\}$ $condi_1, condi_2… condi_{1-1} € \{0, 1\}$.

© 2012, IJMA. All Rights Reserved

Our approach to determine initial condition follows the above definition of correlation immunity. Therefore, the probability for correlation immunity can be modified as: A combining function f of m input variables X1, X2…Xm with $Z = f(X1, X2...Xm)$ $p_1 = P(Z = X_{ik+1} \mid X_{i1}=0, X_{i2}=0 \ldots X_{ik}=0) \neq 0.5$ for some indices $i_1, i_2 \ldots i_{k+1}$.

$X_{i1}, X_{i2} \ldots X_{ik}$ are the outputs from k LFSRs at any clock instant i. Here, conditioning on outputs of k LFSRs $X_{i1}, X_{i2} \ldots X_{ik}$, we will try to determine whether the probability is away from half and if so, we can say the combining function is k-th order correlation immune. We can say,

$p_1 = P(Z = X_{ik+1} \mid X_{i1}=0, X_{i2}=0 \ldots X_{ik}=0) \neq 0.5$ for correct initial condition
$= 0.5$ for wrong initial condition

In this case, all the possible number of combinations may be checked to find the correct initial condition for the ik+1-th LFSR corresponding to the input X_{ik+1} invoking condition on other k LFSRs. The immunity of the combining function f is the minimum of the immunity of all its inputs based on this idea, we adopt the following approach determining the initial conditions when the combining is unknown and correlation immune of unknown order. The initial conditions of the LFSR are the initial parameter of the program, which is the main center of the attention of the attackers.

Attempt to determine Non-correlation immunity

At first, we try to find out that for which input sequence of i-th LFSR $P(C=Xi) \neq 0.5$. Here for all nonzero initial conditions of i-th LFSR, we will generate input sequence Xi and compute P(C=Xi) where $1 \leq i \leq m$ and m is the number of LFSRs used. For initial condition of i-th LFSR for which P(C=Xi) is reasonably away from 0.5, we suspect this initial condition is the correct initial condition for i-th LFSR. When this method fails for all LFSRs or some LFSRs, we will go to the next step.

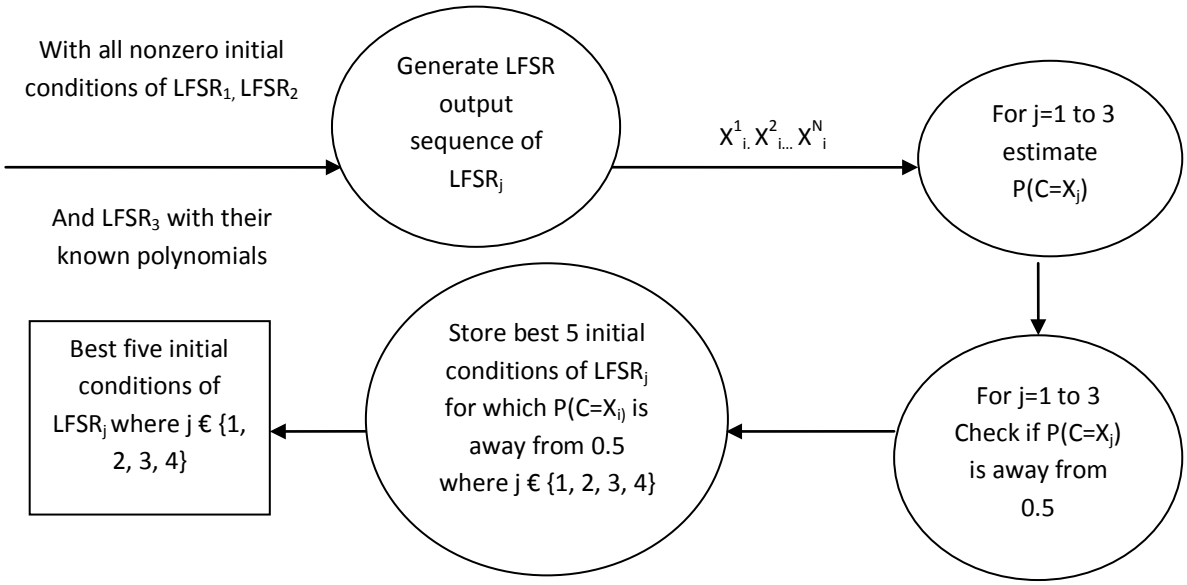


Figure 8: Initial condition determination by checking if the function is non-correlation immune

Algorithm: The following algorithm is developed for checking $P(C=X_j)$ $\forall$ $LFSR_i$ and $LFSR_j$ pairs.

1. Select the polynomial and the length of any $LFSR_i$.
2. Take a particular initial condition, possible for $LFSR_i$.
3. Generate LFSR output sequence for that initial condition say $X_{1i}, X_{2i} \ldots X_{Ni}$ $\forall$ LFSRi.
4. For this initial condition, count fi the number of times C_t of Cipher text C is equal to X_{ti} where t = 1, 2… N.
5. Repeat steps 1 to 4 for all initial conditions possible for $LFSR_i$. The initial conditions for which $|f_i - 0.5|$ is reasonably large as well as sufficiently separated from next (lower) value of |fi .5|, it can be deduced that the corresponding initial condition for $LFSR_i$ is required initial condition.
6. Try steps 1 to 6 for all LFSRi s where i € {1, 2, 3, 4}.

Checking First Order Correlation Immunity

In this case while we check dependency between output of an LFSR and cipher text, we also check if the output of the LFSR is dependent on output of another LFSR. We try to find out for which input sequence of i-th LFSR and which input sequence of j-th LFSR $P(C=X_i \mid X_j=0) \neq 0.5$ where $i \neq j$.

© 2012, IJMA. All Rights Reserved

Here for all possible nonzero condition pairs of i-th LFSR and j-th LFSR, we will generate LFSR output sequence Xi and Xj and compute P($C=X_i \mid X_j=0$) where $\neq j$, $1 \leq i, j \leq m$ and m is the number of LFSRs used. It should be remembered that if any LFSR/ LFSRs is/are is detected in step 1, we don't need to check all initial conditions of the detected LFSR/ LFSRs. For those output sequences X_i and X_j , P($C=X_i \mid X_j=0$) is reasonably away from 0.5, it can be deduced that the corresponding initial conditions are the required initial conditions for i-th LFSR and j-th LFSR respectively . When this method fails for all LFSRs or some LFSRs, we will go next step.

Attempt by checking if function is 1st order correlation Immune for the memory less system

Given bellow is a program structure, which is designed to generate further modules or process for memory less system. Figure 9 represents determining the initial condition of the LFSR. In this case, in spite of taking any higher order of correlation we have started our investigation regarding the determination of the correlation immunity of the LFSRs from the very basic levels. Here we are describing the first order correlation immunity and its fundamentals.

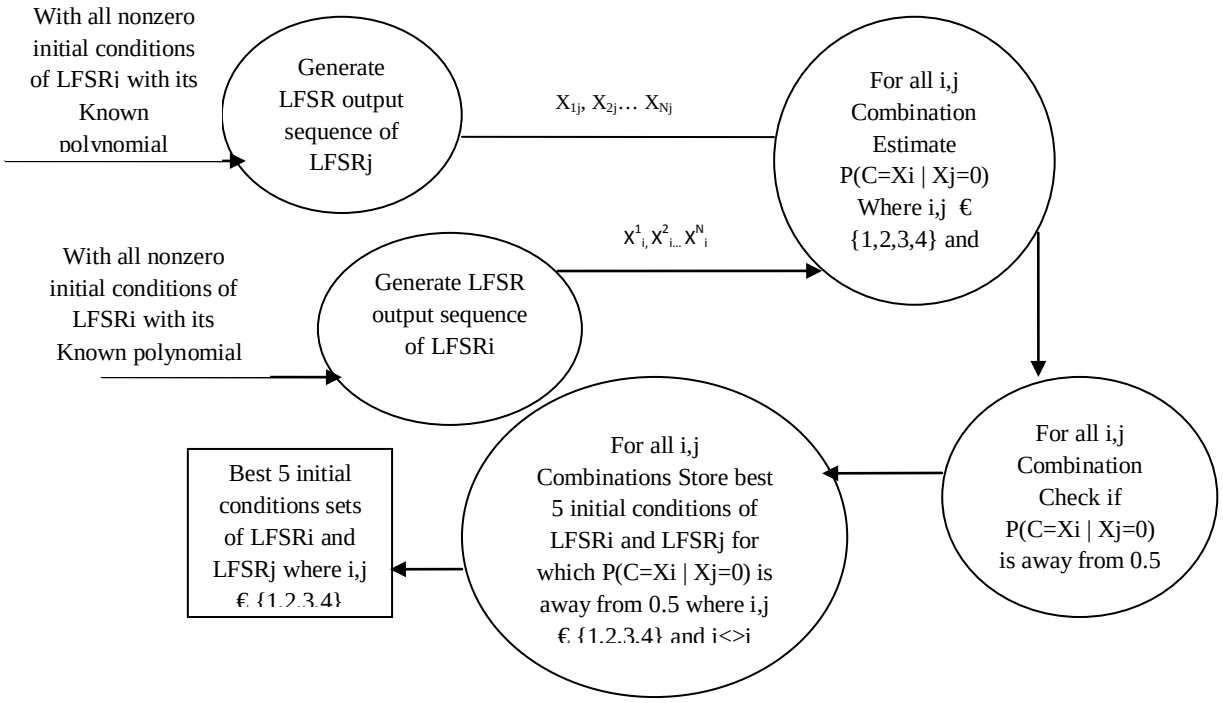


Figure 9: Initial condition determination by checking if the function is first Order Correlation Immune

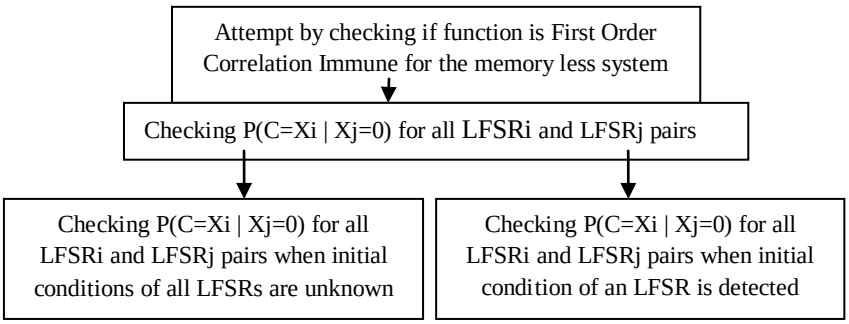


Figure 10: Design steps for Attempt by checking if function is first Order Correlation Immune for the memory less system

Checking P($C=X_i \mid X_j=0$) for all LFSRi and LFSRj pairs is further divided into

- Checking P($C=X_i \mid X_j=0$) for all $LFSR_i$ and $LFSR_j$ pairs when initial conditions of all LFSRs are unknown.
- Checking P($C=X_i \mid X_j=0$) for all $LFSR_i$ and $LFSR_j$ pairs when initial condition of a LFSRs is detected.

The following algorithm is developed for

- Checking P(C=Xi | Xj=0) for all LFSRi and LFSRj pairs when initial conditions of all LFSRs are unknown.

© 2012, IJMA. All Rights Reserved

- LFSRi and LFSRj pairs when initial conditions of all LFSRs are unknown module when initial condition of an LFSR is known.

Algorithm for checking $P(C=X_i \mid X_j=0)$ for all LFSRi and LFSRj pairs when initial conditions of all LFSRs are nknown:

1. Select the polynomial and the length of any two LFSRs say $LFSR_i$ and $LFSR_j$.
2. Take a particular initial condition pair, one taken for LFSRi and other for LFSRj.
3. Generate LFSR output sequence for each initial condition for this pair. Say X_{1i}, X_{2i}… X_{Ni} for $LFSR_i$ and X_{1j}, X_{2j}… X_{Nj} for $LFSR_j$.
4. For this initial condition pair, count f_{ij} the number of times C_t of Cipher text C is equal to X_{ti} when $X_{tj}=0$ where t = 1, 2… N.
5. Repeat steps 1 to 4 for all initial condition pair possible for $LFSR_i$ and $LFSR_j$.
6. The initial condition pair for which $|f_{ij} -0.5|$ is reasonably large as well as sufficiently separated from next (lower) value of $|f_{ij} -0.5|$, it can be deduced that the corresponding initial condition for $LFSR_i$ and $LFSR_j$ are the required initial condition.

7. Try steps 1 to 6 for all $LFSR_i$ and $LFSR_j$ pairs where i,j € {1,2,3,4} and i<>j .

Algorithm for $LFSR_i$ and $LFSR_j$ pairs when initial conditions of all LFSRs are unknown module when initial condition of an LFSR is known.

The algorithm is same as checking $P(C=X_i \mid X_j=0)$ for all $LFSR_i$ and $LFSR_j$ pairs when initial conditions of all LFSRs are unknown is same but except step2 and step6. The known LFSR may be any of $LFSR_i$ and $LFSR_j$.
If the known LFSR is $LFSR_j$, step2 and step6 are changed as follows respectively:

- Take a particular initial condition pair, where one is known initial condition of LFSRj and other is taken from all possible nonzero initial conditions of $LFSR_i$.
- The initial condition pair for which $|f_{ij} -0.5|$ is reasonably large as well as sufficiently separated from next (lower) value of $|f_{ij} -0.5|$, it can be deduced that the corresponding initial condition for LFSRi is required initial condition.

If the known LFSR is $LFSR_i$, step2 and step6 are changed as follows:

- Take a particular initial condition pair, where one is known initial condition of LFSRi and other is taken from all possible nonzero initial conditions of LFSRj.
- The initial condition pair for which |fij -0.5| is reasonably large as well as sufficiently separated from next (lower) value of |fij -0.5|, it can be deduced that the corresponding initial condition for LFSRj is required initial condition.

Attempt by checking if function is 1st order correlation Immune for one memory system

In one memory stream cipher system, we assume $LFSR_1$ is replaced with memory. Figure 11 depicts a program structure for attempt by checking if function is First Order Correlation Immune for the one memory system.

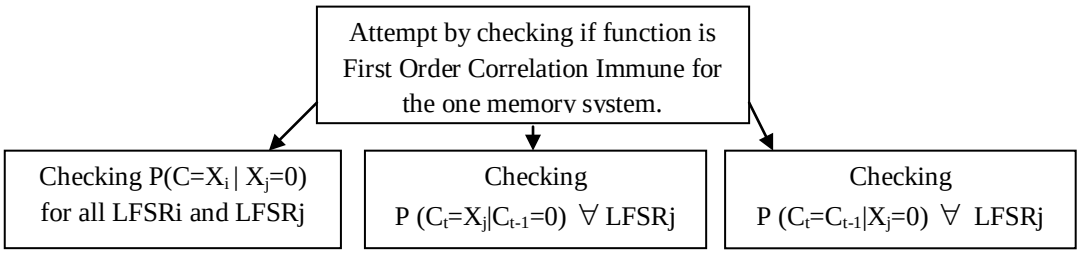


Figure 11: Attempt by checking if function is First Order Correlation Immune for the one memory system.

Sub algorithm for Checking $P(C=X_i \mid X_j=0)$ for all $LFSR_i$ and $LFSR_j$ pairs are same as memory less system.

While devising the algorithm for Checking $P(C_t=X_i \mid C_{t-1}=0)$ for all $LFSR_i$ s and algorithm for checking $P(C_t=C_{t-1}|X_j=0)$ for all $LFSR_j$ are two possible cases where we assume one input to the function is the function output delayed by one clock instant.

© 2012, IJMA. All Rights Reserved

Algorithm for Checking P (Ct=Xj|Ct-1=0) for all LFSRj s is as follows:

1. Select the polynomial and the length of any $LFSR_i$.
2. Take a particular initial condition from all possible nonzero initial conditions of $LFSR_i$.
3. Generate LFSR output sequence for that initial condition say X_{1i}, X_{2i}… X_{Ni} for LFSRi.
4. For this initial condition, count fi the number of times C_t of Cipher text C is equal to X_{ti} when $C_{t-1}=0$ where t = 1, 2… N.
5. Repeat steps 1 to 4 for all initial conditions possible for $LFSR_i$.
6. The initial conditions for which |fi -0.5| is reasonably large as well as sufficiently separated from next (lower) value of |fi -0.5|, it can be deduced that the corresponding initial condition for LFSRi is required initial condition.
7. Try steps 1 to 6 for all LFSRi s where i € {1, 2, 3, 4}.

Algorithm for checking P ($C_t=C_{t-1}|X_j=0$) for all $LFSR_j$ s:

1. Select the polynomial and the length of any LFSRj
2. Take a particular initial condition from all possible nonzero initial conditions of LFSRj.
3. Generate LFSR output sequence this initial condition says X1j, X2j… XNj for LFSRj.
4. For this initial condition, count fj, the number of times Ct of cipher text C is equal to Ct-1 when Xtj = 0 where t = 1, 2… N.
5. Repeat steps 1 to 4 for all possible initial conditions for LFSRj.
6. The initial condition for which |fj -0.5| is reasonably large as well as sufficient separated from next (lower) value of |fj -0.5|, it can be deduced that the corresponding initial condition of LFSRi is the required initial condition.
7. Try steps 1 to 6 for all LFSRj s where j € {1, 2, 3}.

Attempt by checking if function is 1st order correlation Immune for two memory system

In two memory stream cipher system, we also assume LFSR1 is replaced with memory. Figure 12 depicts the program structure for attempt by checking if function is First Order Correlation Immune for the two memory system.

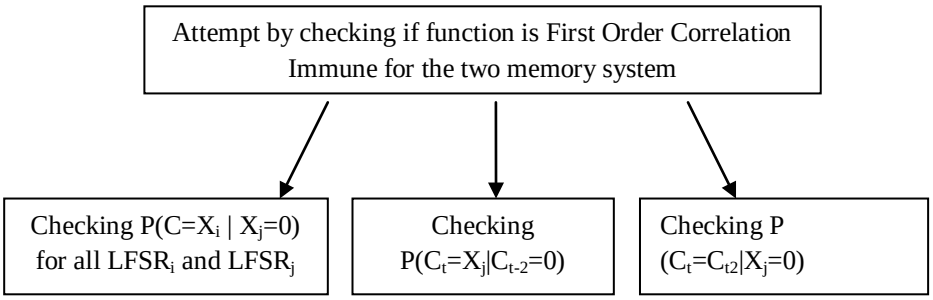


Figure 12: Attempt by checking if function is First Order Correlation Immune for the two memory system

Sub algorithms of Checking P($C=X_i \mid X_j=0$) for all $LFSR_i$ and $LFSR_j$ pairs are same as memory less system. Checking P ($C_t=X_i \mid C_{t-1}=0$) $\forall$ $LFSR_i$ s and checking P ($C_t=C_{t-2}|X_j=0$) $\forall$ $LFSR_j$, we condition on C_{t-2} as we know one input to the function is the function output delayed by two clock instants. The algorithm for Checking P($C_t=X_i \mid C_{t-1}=0$) $\forall$ LFSRi s is same as we deduce for one memory system except step 4.For this initial condition, count f_i the number of times C_t of cipher text C is equal to X_{ti}. When $C_{t-1}=0$ where t = 1,2,…,N. The algorithm for module: checking P (Ct=Ct-2|Xj=0) for all LFSRj is same as we deduce for one memory system except step 4. For this initial condition, count f_i the number of times C_t of cipher text C is equal to $C_{t-2}=0$ when $X_{tj}=0$ where t = 1, 2… N.

5. CONCLUSION:

On the basis of the observed experimental results, tests and analysis performed above it can be inferred that 'NPSBC' is an efficient and a sufficiently strong cryptosystem providing a superior level of security. To conclude the proposed algorithm is a simple, straightforward and compact approach to develop a cryptosystem using the essence of elementary operations. It provides the same or sometimes even better level of security using minimal time complexity. A comparative study and security level will be verified in future and the proposed algorithm may extended to work on other digital media encryption like images.

© 2012, IJMA. All Rights Reserved

6. REFERENCES:

[1] Bement A. L. et. al. (2004), Standards for Security Categorization of Federal Information and Information Systems, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, MD 20899-8900.

[2] Ayushi, (2010), A Symmetric Key Cryptographic Algorithm, International Journal of Computer Applications (0975 - 8887) Volume 1. No. 15.

[3] Atul Kahate, (2008) Cryptography and Network Security, Tata McGraw-Hill Education, pg. 47.

[4] Ijaz Ali Shoukat, Kamalrulnizam Abu Bakar and Mohsin Iftikhar, "A Survey about the Latest Trends and Research Issues of Cryptographic Elements", p 141, International Journal of Computer Science Issues, Vol. 8, Issue 3, No. 2, May 2011, ISSN 1694 0814.

[5] Sukalyan Som, Saikat Ghosh, "A Survey of Traditional or Character Oriented Symmetric Key Cryptography", International Journal of Advanced Research in Computer Science, Vol. 2, No. 4, July-August 2011.

[6] R. Venkateswaran, Dr. V. Sundaram, "Information Security: Text Encryption and Decryption with Poly Substitution Method and Combining the Features of Cryptography", p28-30, International Journal of Computer Applications, Vol. 3, No. 7, June 2010.

[7] T. Siegenthalar, "Correlation-immunity of Non-linear combining functions for cryptographic application", IEEE Transactions on Information Theory, Vol. 30, No 5. September 1984, pp 776-780.

[8] R. Rueppel, "Analysis ans Design of Stream Ciphers", Springer Verlag, Berlin, 1986.

[9] S. Palit and B. K. Roy, "Cryptanalysis of LFSR encrypted codes with unknown combining function", Advances in Cryptology – ASIACRYPT '99, LNCS 1716, 1999, pp. 396-320.

****

© 2012, IJMA. All Rights Reserved