INTERNET: a Cryptosystem for Internet security

Sri Sunil Karforma, Dr. Sripati Mukhopadhyay, A. M. Midda

Abstract

An Algorithm for Cryptography using Linear Feedback Shift Register (LFSR) Polynomial for a Communication System, say, Computer Network, has been developed and implemented. A Cryptography is concerned with keeping, communication private, two men can communicate over an insecure channel, such as Internet in such a way that a third person cannot understand what is being communicated. Such an ideal Crypto System has been developed using L.F.S.R. polynomial.

INTRODUCTION

With the advancement of Computer and Communication technology many publishers are floating journals in the Internet, a tool for library and information service, that requires data security and copyright. Only authorized contributors of that journal should have the access. Not only journals but many secret library information may be needed to communicate between two persons through Internet. Cryptography is concerned with keeping communication private. Network security can be incorporated using cryptographic techniques, making the information meaningless to the interlopers.

The process of Cryptography can be represented graphically as follows: (figure-1)

Figure 1

- Plaintext – Message in original form
- Ciphertext – Encrypted output
- Encryption – Converting Plain text to Cipher text using an algorithm (Mathematical Process)
- Decryption – Reconverting cipher text to plaint text.

ENCRYPTION

Encryption is done by stream cipher. Stream Ciphers operate on bits. A random sequence of bits equal to length of the message is generated. This forms the key stream which is exclusively Xored (XORed) with the message producing bit by bit encryption. Stream ciphers operate on small units of plaintext, usually bits. Stream ciphers can be designed to be exceptionally fast and hence popularly used in cryptographic applications. The chief problem, however, is the generation of a keystream which is truly random in order that the system may be perfectly secure. In practice, a pseudorandom sequence generator is used at both the sender and receiver ends, set up with the same initial conditions. Such a generator is commonly implemented using Linear Feedback Shift Register (LFSR)

An LFSR is a connected series of registers, each of which can store a binary value. The connections are made in such a way that the resulting recurrence relation can be represented by a polynomial. This polynomial is the connection polynomial of the LFSR. When this polynomial is chosen to be primitive, the period of the pseudorandom sequence generated is of maximal length , i.e. 2^d-1 where d stands for the length of the LFSR or number of shift registers. An LFSR with connection polynomial $1+X+X^4$ is shown in Figure

2. The initial condition of the primitive polynomial is set, to begin with in order to compute LFSR's outputs. At each clock-pulse the values of each register are shifted to the left-hand one. Thus the values of the leftmost register are shifted out. This is the output of the register for given clock-pulse. The next outputs of for each clock-pulse are determined by XORing the bits, corresponding to the bit-position of the primitive polynomial and transferring the value into the right-most register simultaneously. The pseudorandom sequence thus generated has random occurrence of a zero and one.

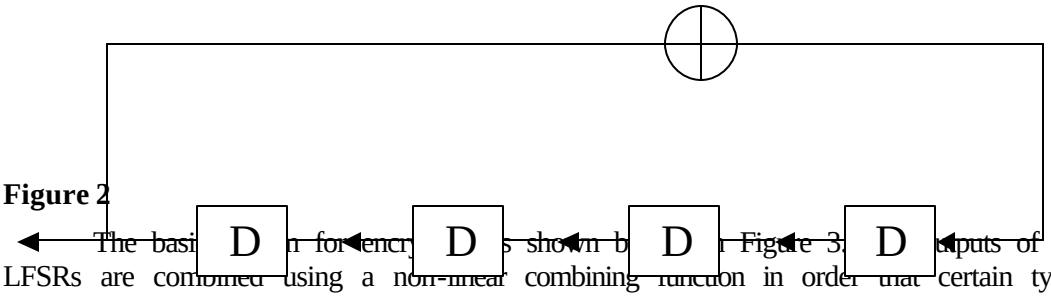

Figure 2

The basic system for encryption is shown below in Figure 3. The outputs of several LFSRs are combined using a non-linear combining function in order that certain types of attacks on the system (utilizing the linearity of the system) can be avoided. The output of the combining function is the keystream which is bit-wise XORed with the coded plaintext (for example, using the Murray code) to form the ciphertext. It can be observed that in order to recover the coded plaintext we need to only XOR the keystream with the ciphertext.

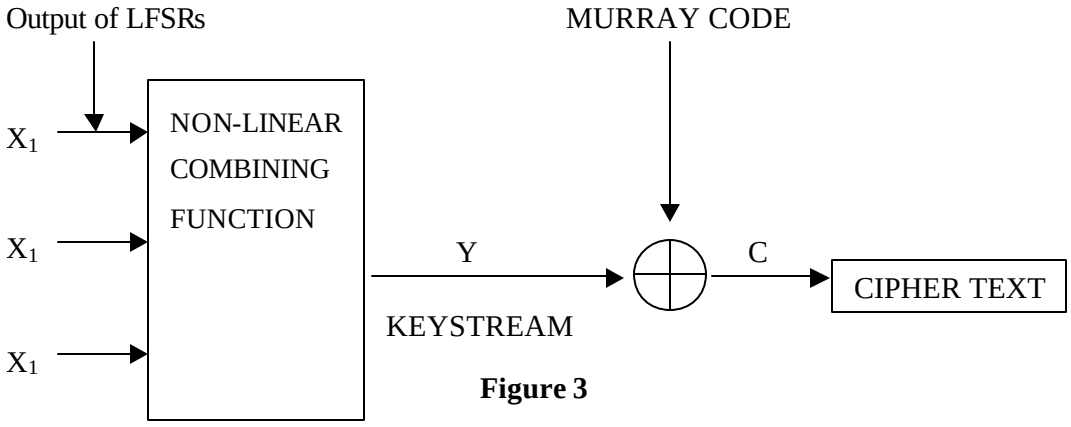

Figure 3

Implementation:

1: LFSR stream generation

Inputs : The length of the polynomial n, the connection polynomial e.g. the polynomial $1+x+x^4$ is represented as 1 0 0 1, the initial condition (of length equalling the degree of the polynomial, d) the desired length m of the output sequence and the no. of taps t.

Output : A pseudorandom sequence with period 2^d -1.

2. The LFSR-based encryption system:

Three LFSRs are combined using a multiplexer i.e.

$Y = X_1X_3 + X_2X_3$

to form the keystream Y. This bit wise XORed with the plaintext (a message in English coded into binary using the Murray code) to produce the ciphertext.

Inputs : A message in English and the LFSR polynomials.

Output : The ciphertext.

DECRYPTION

Attacks on cryptosystems can be a various kinds *viz*. Ciphertext only, Know Plaintext and Chosen Plaintext. In this piece of work, we shall implement a ciphertext only attack. As the name indicates, the ciphertext alone is available to the 'adversary' in this type of attack. The attack considered in the present work, hings on the existence of some correlation between the generated LFSR sequences and the received ciphertext and is hence, termed as a Correlation attack. For the stream cipher architecture outlined in the last section, we shall assume that the LFSR polynomials and the combining function, are both known. The initial conditions form the secret key. Once these are determined, the keystream can be generated and the ciphertext decrypted.

When he ciphertext is correlated with each of the LFSR output sequences, a 'divide and conquer' approach may be adopted and the initial conditions of each LFSR determined separately. If M_i represents the number of initial conditions of ith LFSR, the total number of initial conditions for the composite keystream is $?_{i?1}^{n} M_i$. Using the divide and conquer approach, this reduces to $?_{i?1}^{n} M_i$. Note that $M_I = 2^{di} - 1$, where d_i and this results in an enormous amount of savings in computation if all initial conditions are to be tested. In the algorithm in considered here, we consider only one LFSR without loss of generality (since each LFSR is attacked separately). Further, since we assume that a certain correlation exists between the ciphertext and the LFSR output sequence, say Prob (ciphertext = LFSR output sequence) = p, we generate a ciphertext satisfying this requirement and test the algorithm on this (details cab be found in the Implementation described later).

Algorithm :

1. Let us consider the unknown initial state of the target LFSR, denoted

   $u = (u_1, u_2, \ldots\ldots\ldots\ldots\ldots, u_d)$

   We can express each u_i as some known linear combination of the initial state u, i.e.,

   $$u_i = \sum_{j=1}^{d} w_{ij} u_j \quad ? \; i \; ? \; 1$$

   where $w_{ij}, i ? 1, 1 ? j ? d$ are known constants.

   Define the initial state polynomial, denoted U(x), to be

   $U(x) = U(x_1, x_2, \ldots\ldots\ldots\ldots\ldots., x_d) = u_1x_1 + u_2x_2 + \ldots\ldots\ldots\ldots\ldots\ldots + u_dx_d.$

   The correlation between u_i and z_i can be described by introducing a noise vector e as

   $e = (e_1, e_2, \ldots\ldots\ldots\ldots, e_N)$, Then we model the correlation by writing $z = u + e$, giving

   $z = (U(x_1) + e_1, U(x_2) + e_2, \ldots\ldots\ldots\ldots.., U(x_N) + e_N),$

   where x_i are known.

2. Since U(x) is linear polynomial, the sum of these two noisy observations will give rise to an even more noisy observation in the point $x_i + x_j$, since

   $$\begin{aligned} P(z_i + z_j = U(x_i + x_j)) &= P(z_i + z_j = U(x_i) + U(x_j)) \\ &= P(z_i = U(x_i))\, P(z_j = U(x_j)) + P(z_i = U(x_i))\, P(z_j \; ? \; U(x_j)) \\ &= (1/2 + \; ?)^2 + (1/2 - \; ?)^2 \\ &= 1/2 + 2?^2 \end{aligned}$$

   Now we want to check whether the hypothesized value $(\hat{u}_1, \ldots\ldots, \hat{u}_k)$ of $(u_1, \ldots.., u_k)$ is correct or not. This is done by first selecting a certain (d-k) tuple s_i, and then by finding all linear combinations of t (here t = 2) vectors in $\{x_1, x_2, \ldots\ldots.., x_n\}$,

   $$\hat{x}(i) = \sum_{j=1}^{t} x_{aj}$$

   We get the form

   $\hat{x}(i) = (\hat{x}_1, \ldots\ldots\ldots\ldots, \hat{x}_k, s_{k+1}, \ldots\ldots., s_d)$

   for all values of $\hat{x}_1, \ldots\ldots\ldots, \hat{x}_k$ (not all zero). Let S_i be the number of times the tuple s_i can be formed in this way.

   From out previous arguments, we can get the relation between U(x(i)) and $\hat{z}(i)$ in the form.

   $U(\hat{x}(i)) = \hat{z}(i) + e$

   Where e is a noise vector

   It is equivalent expressed as

   $$\sum_{j=1}^{k} u_j \hat{x}_j \; ? \; \sum_{j=k+1}^{d} u_j s_j \quad ? \quad \hat{z}_1 ? ? \; e$$

   This can be rewritten as

$$\sum_{j=1}^{k} (u_j \oplus \hat{u}_j) \hat{x}_j \oplus \sum_{j=k+1}^{d} u_j s_j \oplus e \quad \oplus \quad \sum_{j=1}^{k} \hat{u}_j \hat{x}_j \oplus \hat{z}(i)$$

where $W = \sum_{j=k+1}^{d} u_j s_j$.

Suppose $\hat{u}_j$ is correct. Then

$$\sum_{j=1}^{k} (u_j \oplus \hat{u}_j) \hat{x}_j = 0$$

and $P(W+e) = 0 = P\left(\sum_{j=1}^{k} \hat{u}_j \hat{x}_j \oplus \hat{z}(i)\right)$

Let $T_i = 1$, whenever $\sum_{j=1}^{k} \hat{u}_j \hat{x}_j \oplus \hat{z}(i)$ and z(i) and num = $\oplus T_i$

If W = 0, $P(W+e) = 0 = 1/2 - 2^{-2}$

It W = 1, $P(W+e) = 0 = 1/2 - 2^{-2}$

where $1/2+2^{-2}$ and $1/2-2^{-2}$ are denoted by P_w.

$\oplus$ num $\oplus$ Bin (S_i, p_w)

where nub has a binomial distribution Bin (S_i, p_w), with p_w being one of the two probabilities. If $\hat{u}_j$ is wrong,

num $\oplus$ Bin (S_i, 1/2)

3. In order to separate between the two, we calculate

$$dist = \sum_{j=1}^{2^{d-k}} (S \oplus 2 * num)^2$$

for all combinations of (S_{k+1}, ……., S_d) for the two candidate values of u_j. Finally, we select u_j for the highest dist value.

4. In order that above algorithm can be executed sequentially, the values of k are varied from 1 to d-1 and u_k calculated. Note that each step, the u_j values calculated in the earlier ones, are used, so that a choice between 0 and 1 only, has to be made.

5. In order to determine u_d, we consider both the candidate values and generate the corresponding LFSR output sequences. We then decide in favour of the one that gives rise to the sequence which has the greater number of matches with the ciphertext.

Implementation :

1. LFSR stream and ciphertext generation.

   Inputs : Degree of the polynomial, the coefficients of polynomial and initial conditions. The value p = Prob (Ciphertext = LFSR output sequence) is kept fixed at 0.6.

   Outputs : LFSR output sequence and Ciphertext.

2. Expressing every bit of LFSR sequence in terms of the initial conditions.

Inputs : Same as above.

Outputs : A matrix with m rows and d columns. The ith row corresponds to the ith bit of the LFSR output sequence and a '1' in the jth column of that row indicates that the bit is dependent on the jth initial condition or u_j.

3. Implementation of the fast correla tion attack.

Inputs : Same as above.

Outputs : The initial conditions u_1, ……., u_l identified to be the correct ones.

CONCLUSIONS

The algorithm was found to work well for a system using an LFSR polynomial $1+x+x^4$ and p = 0.6. Various sets of initial conditions were used and the algorithm was able to identify them correctly in each case. For Internet security efforts are being made to implement DNA cryptography and when it is fully implemented important information in the Internet will remain as a mystery to the third party.

REFERENCES

1. Rabi Chandra Rao, I.K : "The Internet: Regulatory Issues, Organizing resources, Retrieval Engines and its Impact", Bangalore, DRTC workshop, 1998, 1.S.1., 1998.

2. "Fast Correlation attacks through reconstruction of Linear Polynomials"-Thomas Johnson and Fredrik Johansson, Crypto-2000.

3. Computer Express, November, 5, 2001.