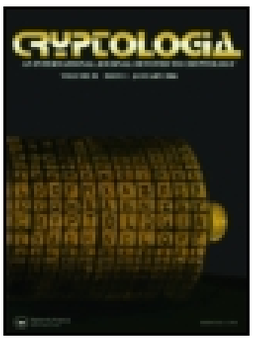


CRYPTOLOGIA

ISSN: 0161-1194 (Print) 1558-1586 (Online) Journal homepage: https://www.tandfonline.com/loi/ucry20

INFORMATION THEORETIC APPROACH TO SECURE LSFR CIPHERS

DAVID AUGUST

To cite this article: DAVID AUGUST (1985) INFORMATION THEORETIC APPROACH TO SECURE LSFR CIPHERS, CRYPTOLOGIA, 9:4, 351-359, DOI: 10.1080/0161-118591860111

To link to this article: https://doi.org/10.1080/0161-118591860111

Published online: 04 Jun 2010.

INFORMATION THEORETIC APPROACH TO SECURE LSFR CIPHERS

DAVID AUGUST

[This paper is the winning paper in the Cryptologia Fourth Annual Undergraduate Paper Competition in Cryptology.]

ABSTRACT: To break a normal LFSR cipher, a cryptanalyst needs only 2n bits of corresponding plain and ciphertext, where n is the number of stages of the shift register.[1] In this paper, a method of substituting completely random characters into the ciphertext and therefore preventing the encipherment of a full 2n-length sequence (under its proper key) will be discussed. Due to the high redundancy of English, a cipher containing several completely random characters will still be readable.

KEY WORDS: Linear feedback, shift register, random substitution, plaintext redundancy, hardware efficiency.

Meyer and Tuchman in [1] demonstrate a method of breaking an n-stage linear feedback shift register (LFSR) given 2n bits of known plaintext. The basic method is to set up the matrix equation

$$\begin{bmatrix} Y_{n+1} + X_{n+1} \\ \vdots \\ Y_{2n} + X_{2n} \end{bmatrix} = \begin{bmatrix} X_n & X_{n-1} & \cdots & X_1 \\ 0 & X_n & X_{n-1} & \cdots & X_2 \\ 0 & 0 & \ddots & & \\ \vdots & & & \ddots & \\ 0 & 0 & & & X_n \end{bmatrix} \begin{bmatrix} S_1 \\ S_3 \\ S_3 \\ \vdots \\ S_n \end{bmatrix}$$

or $K = XS$ where X is the matrix of successive shifts of the first n bits of plaintext, Y is the ciphertext, and S is the unknown matrix of switch states. $Y + X = K$, the key matrix. (Note: In modulo 2 addition, since $X + K = Y$, then $X + Y = K$.) Thus, the switch values can be obtained by inverting X and solving $S = X^{-1}K$.

The matrix method, however, requires 2n consecutive bits of known plaintext, and herein lies its chief weakness. If getting 2n consecutive bits $x_1 \ldots x_{2n}$ is never possible, then the equation provides no solution for S. This, then, is the cryptographer's problem. The trivial answer, of course, limits all messages to length less than 2n. This solution would require either very short messages or enormous shift registers, neither of which are very practical.

A more beneficial answer is found in an information theoretic approach. The English language is quite redundant. For example,

SNINCS WIHT VWLS R STLL FRLY SY T RD.

A message of length k may still be read if it lacks some quantity D(k) of letters. Alternately, the message remains fairly easy to read if D(k) random letters are substituted for characters. Consequently, one may still "read" a 2n-length sequence of characters containing D(2n) random substitutions (eg. bits from a noisy diode), yet never have a "complete" 2n sequence. With a new algorithm that

1) Includes some random bits in every set of 2n message characters

2) Discards the unused part of the 2n sequence of key bits after a random insertion has been made. This prevents a normal known plaintext attack (as in [1]) on the reconstructed plaintext

3) Repeats this process for a message of length k

we can theoretically obtain a modified, secure LFSR cipher. mod (LFSR) may be described functionally using a composite form:

$$\begin{aligned} E_1(X) &= X + K \\ E_2(X) &= cR + (1-c)X \\ E(X) &= E_2(E_1(x)) \end{aligned} \qquad (1)$$

where c is a binary flag, R is some random byte, and K is the key. c is on or off depending on whether or not the key byte corresponds to a certain preset configuration. R is a byte rather than a bit because, to implement the new configuration algorithm, it is necessary to change the normal LFSR cipher from a stream cipher to a block cipher, with a convenient block size of eight.

Before discussing the practicality of the new cipher, we must first review some probability and information theory. The probability of x successes in n independent trials of a random experiment is

$$P(x) = \binom{n}{x} p^x q^{n-x}$$

where p = probability of success of each trial, and q = 1 − p. This is called the "Binomial distribution". Thus, $P(0) = \binom{n}{0} p^0 q^{n-0} = q^n$. For the probability of success to be 99%, an equivalent condition is that the probability of failure P(0) be less than 1%. Thus,

$$\begin{aligned} 1 - P(0) &> .99 \\ 1 - q^n &> .99 \\ .01 &> q^n \\ n &> \ln(.01)/\ln(q) \end{aligned} \qquad (2)$$

provides the number of trials, n, necessary such that the probability of failure is 1% or less.

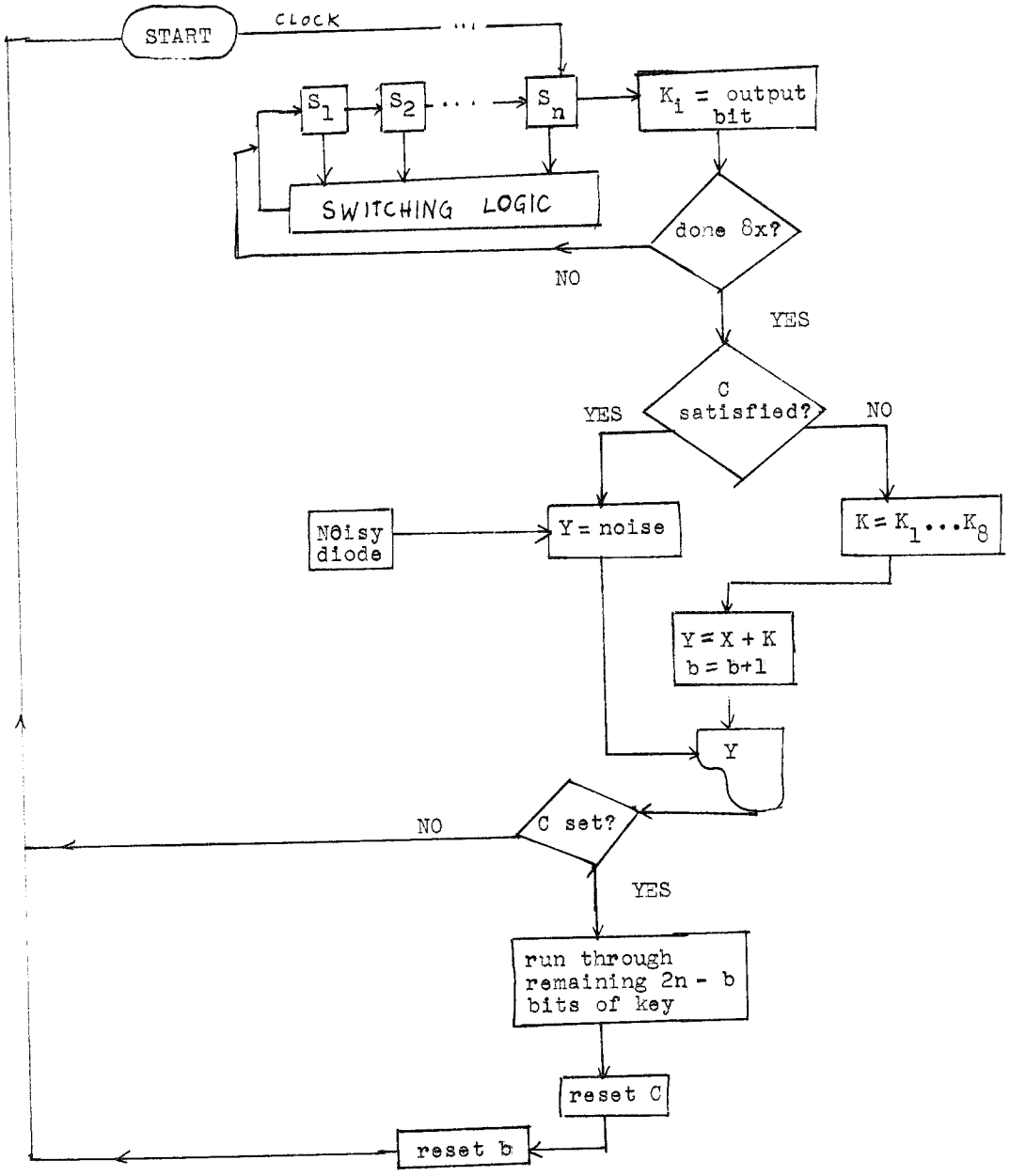
In order to include some random bytes in every 2n set of cipher bits, we need some sort of a "flag" -- a condition available to both parties contained within the key. If the flag is set, we substitute a random byte for ciphertext; if the flag is clear, enciphering proceeds normally as in (1).

The condition that the last two bits of a byte are "11" has, for instance, probability P = 1/4 = (1/2) (1/2). In the corresponding Binomial distribution, q = 1 − p = 3/4, and from (2), we get n > 16. Interpreting this result, we say that for one byte to have the form $x_7x_6x_5x_4x_3x_2$ 11 with 99% probability, we need at least 16 randomly chosen bytes — such as those from a normal LFSR.

Intrinsic to the security of mod(LFSR) is the fact that every message has many completely random "undecipherable" characters, which will be reconstructed in proper plaintext context. This may seem improper at first, but a brief glimpse into the redundancy rate of the English language alleviates any doubts as to the benefit of these "undecipherables."

The redundancy rate of a language D_n/n is defined in [2] as $4.700 - R_n/n$ where R, the rate of the language may be approximated by the entropy of the first n letters of plaintext $H(X_0, X_1 \ldots X_{n-1})$. The limiting value is

$$\lim_{n \longrightarrow \infty} \{h(X_0, \ldots X_{n-1})/n\} = 3.2.$$



Flowchart of Encryption Scheme

Shannon in [5] provides a somewhat clearer definition. The redundancy is one minus the ratio of the plaintext source entropy to the maximum value it could have. Letting Z_m be an alphabet of m symbols,

$$D_n(Z) = (1 - H_n(Z))/n\log Z$$

and

$$H_n(Z) = -\sum_{t \varepsilon Z_m} p(t)\log(p(t))$$

are the redundancy and entropy of n-grams in Z_m. The redundancy of the language is the limiting value

$$D(\text{language}) = \lim_{n \longrightarrow \infty} D_n(Z)$$

and has been assigned many values as Jurgenson [4] points out, ranging from 50% to 78% (Konheim [2] calculates 57%).

In information theory, redundancy has a very specific technical definition, but it can also mean the percentage of characters that may be lost while still leaving the message readable. If the redundancy is 50%, one can reconstruct a message with only half of its total letters. Alternately, a message with 50% random characters should still be readable. For example, since

Y CN RD THS SNTNC WTHT VWLS

then surely,

YOX CANTREID FHISASENTINCE EVKN

THONGHCITAHASKRANDVM CHALAXTERS

Apparently, a substitution of random characters about 25% of the time should not alter readability or reconstructibility. This is the basis for mod (LFSR).

Although it is possible to "read" a 100 letter message with only 75 true characters, it is certainly easier given seven-eighths or 88 characters. In this case, the probability of the "randomizing flag" would only be p = 1/8. The tradeoff is in complexity. Since n = ln(.01)/lnq, as q increases from 3/4 to 7/8, n must increase from 16 to 35, producing a corresponding increase in

the number of stages in the shift register. In a sense, the cryptographer is involved in a mathematical game with himself. He seeks to maximize readability and minimize shift register length in order to obtain a maximum "payoff" in terms of security. A summary of tradeoffs between n and p appears in Table 1.

Case (2) is highly secure and easy to read, but rather complex to implement. Case (1) is just the opposite. Case (3) yields higher security and better readability, while not being overly complex to implement. The probability p = .1875 is obtained by setting the randomizing condition to be P(C) = P(A) or P(B) or P(A and B), where

CASE	p	Confidence level	n	length of sequence never obtainable	# of stages of LFSR
(1)	1/4	90%	8	64	32
		99%	16	128	64
(2)	1/8	90%	18	144	72
		99%	35	280	140
(3)	.1875	90%	12	96	48
		95%	15	120	60

Table 1.

$A = (x_7x_6x_5x_4x_3111)$ bit configuration

$B = (x_7x_6x_5x_4111x_0)$ bit configuration

Thus, P(C) = (1/4) (3/4) = .1875. This is a more ideal condition: The message will be easier to read since randomizing condition p is not as large as 1/4, but on the other hand, the shift register will not have to be as large as with p = 1/8. Additionally, the cipher will have a 95% security level. While on the subject of comparisons, an important aside is that for the same randomness of key, it is still easier to implement even a fairly large shift register than, say, the Natural Bureau of Standards Data Encryption Standard. Returning to equation (1), we let c = 1 if C is satisfied and c = 0 otherwise.

The first part of the algorithm has been discussed. However, the opponent may still perform a known plaintext attack as in [1]. Step 1 of the algorithm assures only that a random block will occur in every 2n bit set, not where in the 2n set it will be. A portion of this must be thrown out after each random substitution. Consider a 16 stage shift register. Then 2n = 32. If the block size is 8 bits, a sequence of 2n may still be obtained even though every set of 32 bits has a random character (see Illust. 1). If, on the other hand, the modified enciphering algorithm automatically "runs through" — calculates, but does not use — A bits of key after each random substitution, we do not have this problem of ciphertext/reconstructed plaintext to key correspondence. For convenience, we let A = 2n − b, where b is the number of bits between the current random substitution and the previous one (see Illust. 2). In this case $X_i = Y_i$ will not equal K_i, and therefore $K \neq X^{-1}S$. In these Illustrations, $K_a = K_{7a}K_{7a+1} \ldots K_{7a+7}$, and the same notation is used for X_a and Y_a.

In summary, to break a normal LFSR of n = 32 stages requires only 64 bits of known plaintext. Examples of these 64-bit sequences abound; "BUSINESS", "DEAR*SIR", "PROGRAM*", and "THAT*IS*" are only a few that come to mind. However, using a mod(LFSR) algorithm, it is possible to get a cipher 90% secure using a 32 stage shift register — if one is willing to have only 3/4 of the characters come out decipherable. A cipher 95% secure with even fewer "undecipherables" (between 1/8 and 1/5) is possible with an addition of only 16 stages to the shift register. Further,

length 2n			length > 2n
$Y_1\ Y_2\ R\ Y_4$	$Y_5\ R\ Y_7\ Y_8$	$R\ Y_{10}\ Y_{11}\ Y_{12}$	$Y_{13}\ Y_{14}\ Y_{15}\ R$
$X_1\ X_2\ X_3\ X_4$	$X_5\ X_6\ X_7\ X_8$	$X_9\ X_{10}\ X_{11}\ X_{12}$	$X_{13}\ X_{14}\ X_{15}\ X_{16}$
$K_1\ K_2\ -\ K_4$	$K_5\ -\ K_7\ K_8$	$-_9\ K_{10}\ K_{11}\ K_{12}$	$K_{13}\ K_{14}\ K_{15}$
			corresponding key

Illustration 1.

		length > 2n	
$Y_1\ Y_2\ R\ Y_4$	$Y_5\ R\ Y_7\ Y_8$	$R\ Y_{10}\ Y_{11}\ Y_{12}$	$Y_{13}\ Y_{14}\ Y_{15}\ R$
$X_1\ X_2\ X_3\ X_4$	$X_5\ X_6\ X_7\ X_8$	$X_9\ X_{10}\ X_{11}\ X_{12}$	$X_{13}\ X_{14}\ X_{15}\ X_{16}$
$K_1\ K_2\ -\ K_5$	$K_6\ -\ K_9K_{10}$	$-\ K_{13}\ K_{14}\ K_{15}$	$K_{16}\ K_{17}\ K_{18}\ -$
			different key

Illustration 2.

obtaining higher security necessitates a smaller proportional change in shift register length than without the modifying algorithm.

Calculating maximal length irreducible polynomials of high degree is a fairly easy process given one of a lower degree (cf. [3], [4]).

Very high security without a large shift register is only marginally approached using a normal LFSR to encipher. Consider this final comparison: To encipher a 500 letter message without using mod(LFSR) requires a shift register of minimum length (1/2) * (8 * 500) = 2000 stages for near perfect security. Using mod(LFSR), one can get a 99% secure cipher with a much smaller shift register -- about 1/20 the size. And, much longer messages can be enciphered with the same smaller shift register using mod(LFSR) encipherment.

For those messages where it is vital that every single letter be correctly reconstructed in deciphered context, use this procedure: Encipher the message, decipher it, give it to someone else, and if that person cannot properly reconstruct the deciphered text, re-encipher with different keys until reconstructability is possible.

The statistical information theoretic approach to more secure encryption may have other applications as well. One can foil an "index of coincidence" attack on a Vigenere cipher. Normally, to determine the period, one finds the number of coincidences between successive shifts of the ciphertext Y_s against itself, Y. Whenever IC(Y, Y_s) is a local maximum, s is some multiple of the period. To foil this attack, place random letter wherever coincidences occur for shift number r, (r is the period) until IC(Y,Y_r) is within the range of the other values, {IC(Y,Y_1), . . . IC(Y,Y_{r-1})}. True, this procedure will change the other IC's, but relative values should remain the same, since the cipher has a fairly flat distribution to begin with.

Consider even a simple substitution cipher, $S:X \rightarrow Y$. The weakness of simple substitution is that the rearranged frequency curve resembles an exp(-x) form rather than a flat distribution. If the cryptographer were to substitute S(E) with S(Z), say, the message would still be readable, yet at the same time, the frequency of "E" would be lowered, and the frequency of "Z" would be raised. Perhaps a similar substitution of S(J) for S(T) and S(Q) for S(A) would make frequency based attack more difficult.

In reality, there are other methods (eg. the phi test) for determining the period of a Vigenere, and simple substitution is a hopeless case no matter what is done to strengthen it. However, substituting completely random

characters on the basis that messages, due to their high redundancy, will still be readable is surely a method deserving more research.

REFERENCES

1. Meyer and Tuchman. 1972. Pseudorandom Codes can be Cracked. Electronic Design. 23:74 - 76.

2. Konheim, A. 1981. Cyptography: A Primer. New York: John Wiley and Sons.

3. Golomb, S. 1967. Shift Register Sequences. San Fransico: Holden-Day. (pp.73-74).

4. Jurgensen, H. 1983. Language Redundancy and the Unicity Point. Cryptologia. 7:37-48.

5. Shannon and Weaver. 1949. The Mathematical Theory of Communication. Urbana IL: Univ. of Illinois Press.

For tables of shift register polynomials, see

6. Peterson, W. W. 1961. Error Correcting Codes. New York: John Wiley and Sons.