Jurnal Teknologi, 46(D) Jun 2007: 1–16

COMPARISON ANALYSIS OF STREAM CIPHER ALGORITHMS FOR DIGITAL COMMUNICATION

ABD RAHIM MAT SIDEK[1] & AHMAD ZURI SHA'AMERI[2]

Abstract. The broadcast nature of radio communication such as in the HF (High Frequency) spectrum exposes the transmitted information to unauthorized third parties. Confidentiality is ensured by employing cipher system. For bulk transmission of data, stream ciphers are ideal choices over block ciphers due to faster implementation speed and not introducing error propagation. The stream cipher algorithms evaluated are based on the linear feedback shift register (LFSR) with nonlinear combining function. By using a common key length and worst case conditions, the strength of several stream cipher algorithms are evaluated using statistical tests, correlation attack, linear complexity profile and nonstandard test. The best algorithm is the one that exceeds all of the tests.

Keywords: Confidential, LFSR, stream, block, correlation

Abstrak. Penghantaran maklumat dalam sistem komunikasi radio seperti frekuensi tinggi akan mendedahkan maklumat itu kepada pihak-pihak yang tidak berkaitan. Untuk memastikan maklumat tersebut selamat, ia haruslah dienkodkan terlebih dahulu sebelum dihantar. Bagi maklumat bersaiz besar, pengenkod jenis satu bit adalah lebih sesuai berbanding pengenkod jenis blok kerana ia lebih cepat dan tidak mempengaruhi bit bersebelahan jika berlakunya kesilapan semasa penghantaran. Pengenkod satu bit biasanya dihasilkan menggunakan kaedah anjakan balik secara linear dan juga penggabungan secara tidak linear. Dengan menggunakan panjang kunci yang sama untuk setiap pengenkod iaitu 64 bit, kekuatan pengenkod ditentukan dengan menggunakan beberapa jenis ujian piawaian. Pengenkod yang melepasi kesemua ujian adalah yang paling baik dan sesuai untuk digunakan dalam penghantaran maklumat digital.

Kata kunci: Penghantaran, pengenkod, komunikasi, selamat, linear

1.0 INTRODUCTION

The introduction of advanced techniques in communication and digital signal processing technology has improved the reliability of communication in the HF spectrum. Besides, voice and telegraphy, text, fax and images can be transmitted using HF modem [1, 2]. The broadcast nature of communication exposes the transmitted information to unauthorized third party. Confidentially can be ensured by employing cipher systems. Examples of the systems available in the market are Mils Electronic

[1&2] Digital Signal Processing Lab, Faculty of Electrical Engineering, Universiti Teknologi Malaysia
81310 UTM Skudai, Johor, Malaysia
Email : abdrahim30@lycos.com[1], ahmadzs@yahoo.com[2]

System 700 [3] and HC-6830 by Crypto AG [4]. Both systems are supported with stream cipher to keep information safe. This is because stream ciphers are ideal choices over block ciphers for bulk transmission of data such as text due to faster implementation and not introducing error propagation.

This paper presents results for the analysis of various stream cipher algorithms such as Linear Generator, Improved Geffe, Summation Register (2 and 3 Registers), Shrinking, Multiplexing and Variable - Memory Binary Generator (Memory Generator). All of these are based on the linear feedback shift registers with combining functions. For a common key length of 64 bits, the objective is to find the best algorithm using the statistical tests, linear complexity profile, correlation attack and guess and determine attack. If necessary, the security of the algorithm can be enhanced by increasing the key length to 128 bits or more.

2.0 CIPHER SYSTEMS

Cipher system is a system that encrypts and decrypts data. Generally, there are two types of cipher systems: symmetric and asymmetric. The symmetric or the secret key cipher system uses the same key in encryption and decryption while asymmetric or the public key cipher system uses one key to encrypt and the other corresponding key to decrypt. A cipher system comprises of 5 elements: the plaintext, cipher text, key, encryption and decryption functions [5].

In general, the symmetric cipher systems are designed to ensure that cipher text can only be decrypted by the corresponding secret key [6]. The core security factor for symmetric cipher system is the encryption algorithm. It must be strong enough so that it is impossible to decrypt a message based on the cipher-text alone. Secondly, the security is dependent on the secrecy of the key, and not on the algorithm. Thus, it is not required to keep algorithm secret.

There are two types of symmetric algorithms: stream ciphers and block ciphers. Examples of symmetric cipher system are Data Encryption Standard (DES), Advanced Encryption Standard (AES), IDEA, CAST, SEAL and RC4 [7]. Among these, RC4 and SEAL are the stream ciphers and the rest are block ciphers.

The asymmetric or the public key cipher system makes use of various intractable mathematic problems that are normally easy to calculate from one way but not its inverse. These problems include discrete logarithm problem, the integer factorization problem and elliptic curve discrete logarithm problem. Asymmetric cipher system allows public keys to be simply distributed to the public. Data that is encrypted using the published public key can only be decrypted using the secure private key kept secretly by individuals. The computation of one key from the other key is infeasible. Examples of asymmetric cipher systems are RSA, El-Gamal and elliptic curve [8].

3.0 STREAM CIPHER

The stream cipher [9] is a symmetric cipher that is described in Figure 1. Unlike the block cipher, the stream cipher enciphers the plaintext one bit at a time and only utilizes the confusion characteristics, not diffusion of the cipher system. This is because each character of the cryptogram is independent of each other. The key component of the stream cipher is the keystream generator as shown in Figure 2. Various types of stream cipher can be generated based on the choice of the combining function and the linear feedback shift registers (LFSR).

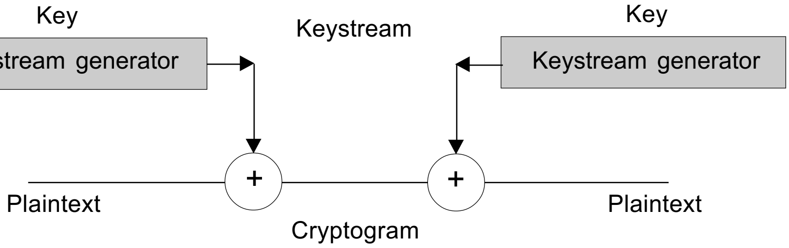

Figure 1 Block diagram of stream cipher

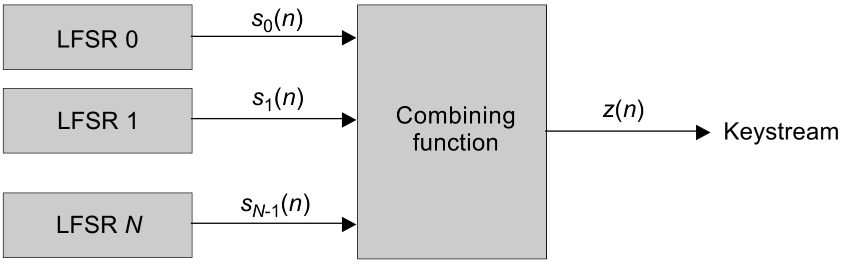

Figure 2 Block diagram of keystream generator

Each of the linear feedback shift registers (LFSR) that forms the keystream generators is shown in Figure 3. The combining function can be any logic functions either linear or nonlinear combination depending on algorithm.

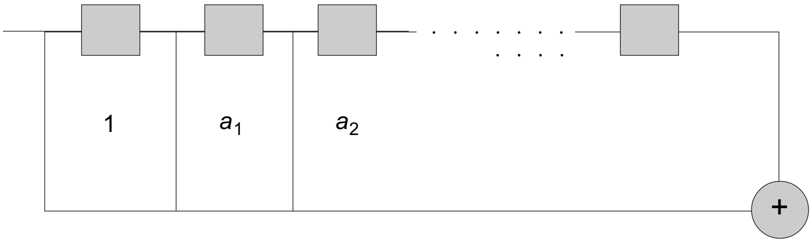

Figure 3 A general case *N*-length linear feedback shift register (LFSR)

3.1 Worst Case Condition

A cipher system is secured as long as the attacker does not know the algorithm, the keys and equivalent plaintext and cryptogram. The algorithm is compromised if any one of these factors is known. This leads to the worst case cryptographic conditions [9] which are:

(i) The algorithm can completely be described.
(ii) A considerable amount of cryptogram is known.
(iii) There is certain amount of cryptogram and its equivalent plaintext available.

For the first condition, the only security that can be provided by algorithm is in the key length. Attack based on an exhaustive keysearch of all the possible keys should not be possible within a reasonable time. Given the present computing technology, a 64 bit key is no longer considered safe compared to a 128 bit key. That is why a double or triple DES (Data Encryption Standard) block cipher is considered more secured compared to the basic DES algorithm. If a cryptogram is random, the second condition is not valid since it is not possible to derive the plaintext from the cryptogram based on the statistics. Condition (iii) is as the result of the known plaintext attack where it is possible to derive the keystream from the cryptogram and its equivalent plaintext. The strength of the keystream can be tested based on the correlation attack and the linear complexity profile. An algorithm that can be proven of its strength under the worst case cryptographic condition is considered ideal.

4.0 KEYSTREAM GENERATOR

There are many different pseudorandom sequence generators applied to stream ciphering. This paper only discusses about stream cipher based on the structure of LFSR as described in the previous section. The LFSR provides a simple way to obtain sequences of vary high periods together with excellent statistics properties. The following subsections explain the stream cipher based on LFSR and non-linear combining function.

4.1 Linear Generator

A set of M LFSR's is combined such that the output sequence is

$$z(n) = s_0(n) \oplus s_1(n) \oplus s_2(n) \oplus \ldots \oplus s_{M-1}(n) \tag{1}$$

where $s_0(n)$, $s_1(n)$ ... $S_{M-1}(n)$ represent the outputs of LFSR 0 to M-1 [7]. If M is chosen as 3, then Equation (1) becomes

$$z(n) = s_0(n) \oplus s_1(n) \oplus s_2(n) \tag{2}$$

The LFSR used are defined by polynomials over GF(2) as shown in Equation (3), (4) and (5):

$$f(x) = 1 + x + x^2 + x^5 + x^{19} \tag{3}$$

$$f(x) = 1 + x^5 + x^{23} \tag{4}$$

$$f(x) = 1 + x^2 + x^{29} \tag{5}$$

4.2 Improved Geffe

This is a keystream generator based on a nonlinear combining function with three LFSRs [11]. The output of the generator is

$$z(n) = s_0(n).s_1(n) \oplus s_0(n).s_2(n) \oplus s_1(n).s_2(n) \tag{6}$$

where $s_0(n)$, $s_1(n)$ and $s_2(n)$ are the output of the LFSR 0 to 2. The LFSR's used are defined by polynomials as in Equation (3) to (5) in GF(2).

4.3 Summation Register

Unlike the Improved Geffe generator, this keystream has an internal memory in the combining function [11]. The summation register is

$$< z(n), c_{out}(n) >= \sum (s_0(n), s_1(n), c_{in}(n)) \tag{7}$$

where $s_0(n)$ and $s_1(n)$ are the output of LFSR 0 and 1, $c_{in}(n)$ is the carry-in, $c_{out}(n)$ is the carry-out and $z(n)$ is the arithmetic sum of the input. Its equivalent Boolean equations are

$$z_0(n) = s_0(n) \oplus s_1(n) \oplus c_{in}(n) \tag{8}$$

$$c_{out}(n) = s_0(n).s_1(n) \oplus s_0(n).c_{in}(n) \oplus s_1(n).c_{in}(n) \tag{9}$$

Possible output for the generator can be chosen as:

$$z(n) = z_0(n) \text{ if } c_{in}(n) = c_{out}(n-1) \tag{10}$$

$$z(n) = c_{out}(n) \text{ if } c_{in}(n) = z_0(n-1) \tag{11}$$

The configurations based on Equation (10) and (11) are known as summation register (z_0 output) and summation register (c_{out} output) respectively. The polynomials used in GF(2) are

$$f(x) = 1 + x^2 + x^{29} \tag{12}$$

$$f(x) = 1 + x^3 + x^{41} \tag{13}$$

Like linear generator, summation register is also expandable depending on numbers of LFSR used. Previous description uses 2 LFSR, $s_0(n)$ and $s_1(n)$ respectively. For 3 LFSR nonlinear combinations, Boolean equations are

$$z_o(n) = s_0(n) \oplus s_1(n) \oplus s_2(n) \oplus c_{in}(n) \tag{14}$$

$$\begin{aligned} c_{out}(n) = s_0(n).s_1(n) \oplus s_0(n).s_2(n) \oplus s_0(n).c_{in}(n) \oplus s_1(n).s_2(n) \oplus \\ s_1(n).c_{in}(n) \oplus s_2(n).c_{in}(n) \end{aligned} \tag{15}$$

where $s_0(n)$, $s_1(n)$ and $s_2(n)$ are the output of the LFSR 0 to 2. The LFSR's used are defined by polynomials as in Equation (3) to (5) in GF(2).

4.4 Multiplexer Register

This cipher uses a standard digital electronic component called multiplexer [9]. A multiplexer has many inputs but only one output. The set of inputs (often termed select lines) uses the state of LFSR 0, R_0 to provide an address for the multiplexer. The output of multiplexer is then merely the member of LFSR 1, R_1 specified by the address. Figure 4 shows the example of nonlinear combination for multiplexing generator. Polynomial used to provide R_0 and R_1 respectively defined in Equation (12) and (13) in GF(2).

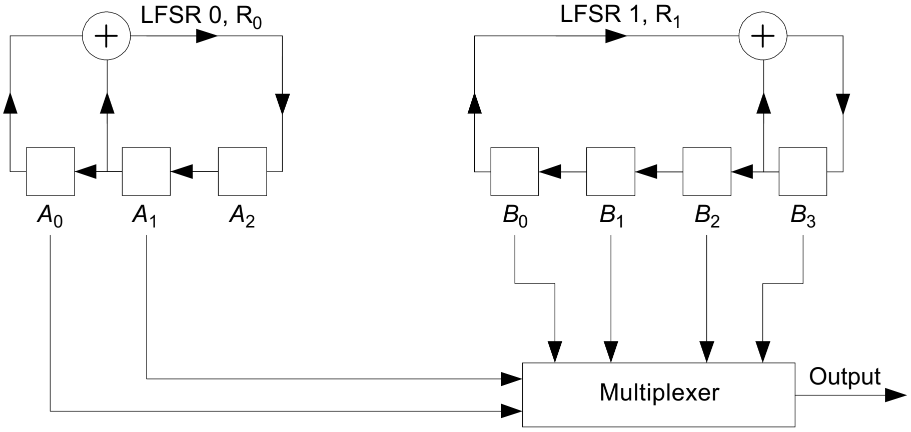

Figure 4 A 4 to 1 multiplexing register

4.5 Shrinking Register

Shrinking register is a promising candidate for high speed encryption applications [11]. This cipher uses 2 LFSR's in non-linear combinations where LFSR R_0 is used to select a portion of the output sequence of a second LFSR R_1. The keystream produced from the output sequence of R_1 is shown in Figure 5.

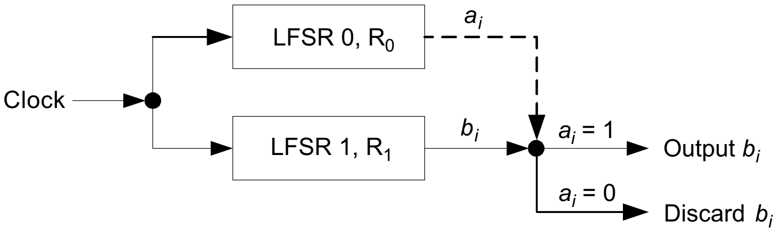

Figure 5 The shrinking register

Instead of using LFSR R_0 to select sequences of LFSR R_1, modifying the condition of selector is another method to implement this cipher. This modified cipher is called XOR-shrinking register as shown in Figure 6.

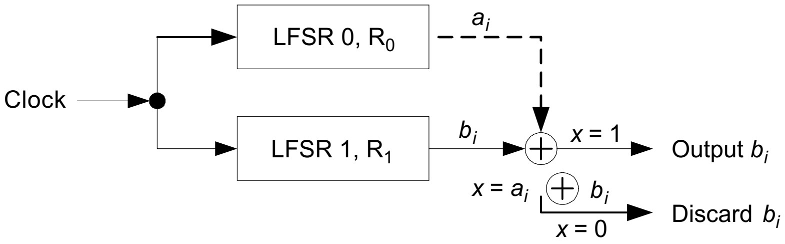

Figure 6 The XOR-shrinking generator

Both shrinking and XOR-shrinking use polynomials as of Equation (12) and (13) in GF(2) for LFSR R_0 and R_1.

Another type of shrinking cipher is called self-shrinking generator [7]. This cipher uses single LFSR only. A pair of bits from LFSR is taken and if the first bit in the pair is 1, the output of the generator is the second bit, or else, the pair will be discarded and the same process repeats for the next clock. The polynomials used in GF(2) is

$$f(x) = 1 + x^2 + x^3 + x^{64} \tag{16}$$

4.6 Variable-Memory Binary Generator (Memory Generator)

Variable-Memory Binary Generator which is also called memory generator needs three LFSRs, R_0, R_1, R_2 respectively and a memory to implement [12]. First, the output of

R_1 read out of the memory location addressed by the read address becomes a keystream sequences. Second, the output bit of R_0 is written into the memory location addressed by the write address R_2. Non-linear combination for this cipher is more like substitution as shown in Figure 7.

For analysis purpose, two types of memory are used that are 16 and 64 addresses. Polynomials involved for all LFSRs are defined in Equation (3) to (5) for GF(2).

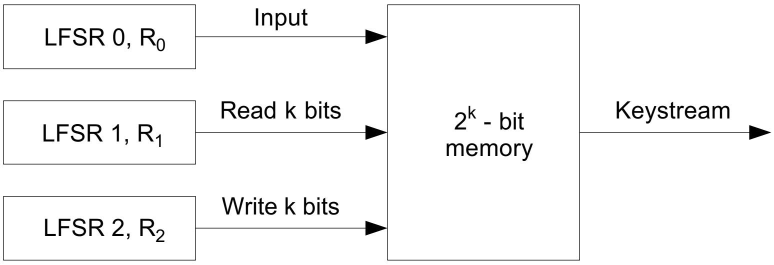

Figure 7 The variable-memory binary generator

5.0 TESTS FOR STREAM CIPHER

There are several tests that can be used to quantify the strength of stream ciphers. Standard tests that are independent of the algorithm are statistical tests, correlation attack and linear complexity profile. Nonstandard test specific to the algorithm based on the guess-and-determine attack is also used.

5.1 Statistical Tests

A binary sequence is said to be random if there is no obvious relationship between the individual bits of the sequence. Since the sequence generated by the LFSR is periodic with a period p, then it is not considered a true random sequence but is referred as a pseudorandom sequence or a pn (pseudonoise) sequence. For this class of sequences, the randomness postulates by Golomb [13] is applicable. If the period p is large, it is of interest to evaluate the randomness of the sequence within an observation interval that is referred as the local randomness. The statistical tests are derived from hypothesis testing and the standard statistical tables utilized are found in [14]. A set of statistical tests applicable are frequency test, serial test, poker test, autocorrelation test and runs test.

5.1.1 Frequency Test

This test calculates the number of ones and zeroes of the binary sequence and checks if there is no large differences.

5.1.2 Serial Test

The transition characteristics of a sequence such as the number 00, 01, 10 and 11 are evaluated. Ideally, it should be uniformly distributed within the sequence.

5.1.3 Poker Test

A N length sequence is segmented into blocks of M bits and the total number of segments is N/M. Within each segment, the integer value can vary from 0 to $m = 2^{M\text{-}1}$. The objective of this test is to count the frequency of occurrence of each M length segment. Ideally, all the frequency of occurrences should be equal.

5.1.4 Autocorrelation Test

The test performs the autocorrelation of the sequence and compares the value of the maximum peak with the value in the origin, the principal-secondary lobe. The worst result of this test is when there is a large peak because many of bits shifted will reflect the same behaviors as the originals. It is preferably having many reasonable middle peaks than the few high peaks, also due to the correlation immune attack. The outcome of the rest will reflect the relationship principal-secondary lobe in dB.

5.1.5 Runs Test

A sequence is divided into contiguous stream of 1's that is referred as blocks and contiguous stream of 0's that is referred as gaps. If r_{0i} is the number of gaps of length i, then half of the gaps will have length 1 bit, a quarter with length 2 bits, and an eighth with length 3 bits. If r_{1i} is the number of blocks of length I, then the distribution of blocks is similar to the number of gaps.

5.2 Correlation Attack

A pseudorandom sequence $z(n)$ is generated by a combination of N LFSR where $s_0(n)$, $s_1(n)$ ... $s_N(n)$ and p_0, p_1 ... p_N are their input sequences and periods respectively as shown in Figure 8. There is a possibility that any one of the input sequences will leak into the output $z(n)$. A sequence is said to be N-th order correlation immuned if it is not possible to correlate any combination of n input sequence to the output [15].

For a pseudorandom sequence that is proven statistically random, it is desired to find a sequence from an external register that is correlated to this sequence. The following block diagram of Figure 8 explains how this is done. The setting of the attacking register to anyone of the input register will result in the resulting sequence $z_0(n)$ to be nonrandom. The randomness of $z_A(n)$ is quantified by performing a statistical test such as a frequency test on the sequence.

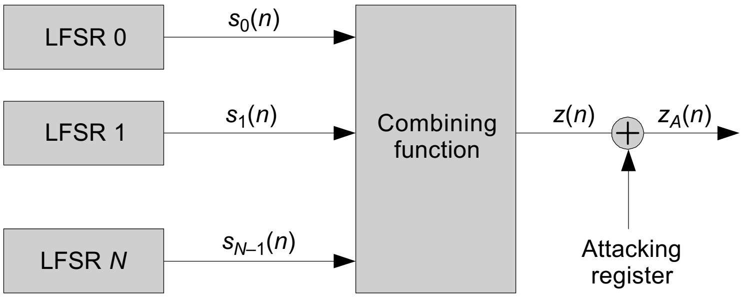

Figure 8 Methodology for performing correlation attack on a keystream generator

5.3 Linear Complexity Profile

If N length sequence is

$$s(n) = [s(0)\ s(1)\ s(2)\ \dots\ s(N)] \tag{17}$$

then the linear complexity of the sequence, denoted by $L(N)$, is the length of the shortest LFSR that will generate the first N terms of $s(n)$. The properties of the linear complexity for random sequence $s(n)$ and $r(n)$ are:

(i) For any $N \geq 1$, the linear complexity of a sequence $s(n)$ satisfies $0 \leq L(N) \leq N$.
(ii) $L(N) = 0$ if and only if $s(n)$ a zero sequence of length N.
(iii) $L(N) = N$ if and only if $s(n) = [0,0,0,0 \dots 1]$.
(iv) If $s(n)$ is periodic with period N, then $L(N) \leq N$.
(v) If the linear complexity of $s(n)$ and $r(n)$ are $L(N)$ and $L(M)$ respectively, the linear complexity of a sequence

$$z(n) = s(n) + r(n)$$

Its relation to L(N) and L(M) is

$$L(P) \leq L(N) + L(M)$$

If $s(n)$ sequence is defined as in sequence (17), then the linear complexity profile L_N of the sequence up to N is

$$L_1, L_2, \dots L_{\text{N}} \tag{18}$$

Ideally, the linear complexity profile of a random sequence will increase linearly with the n. If the linear complexity approaches a constant value L, the resulting L-th length LFSR can be used to regenerate the random sequence. For this analysis, Berlekamp-Massey algorithms are used as Linear Complexity Profile Test and details of the algorithm can be found in [16].

5.4 Guess and Determine (GD) Attacks

Guess and Determine (GD) is a non-standard and effective method in analyzing stream ciphers. By considering the worst case condition, this attack will exploit the relationships between internal values (such as the recurrence relationship in a shift register) and the relationship used to construct the keystream values from the internal values [17]. A GD attack guesses some internal values and then exploits the relationships to determine other internal values. The cipher is broken when a complete internal state has been determined from the guessed values. However, this attack limits on generators that use more than one LFSR and their non-linear function is not complex where register value and keystream that is produced correlates either forward or backward processes. Here is the example of GD attacks on Multiplexing Generator. First, consider all of these following assumptions:

(i) LFSR 1 becomes a selector and polynomial of LFSR 1 is GF(2^3):

$$f(x) = 1 + x^2 + x^3$$

(ii) LFSR 2 as input and polynomial of LFSR 2 is GF(2^4) :

$$f(x) = 1 + x^3 + x^4$$

(iii) Amount of keystream is known. For example: $K(n) = 1,1,1,0,0,0,0,1$

(iv) Complete knowledge of algorithm. In this case, the algorithm is using 4 to 1 multiplexer and X^3 and X^2 states as selectors.

Then, guess the initial condition of LFSR 1. In this case, total guessing equal to 2^3. For each trial, the attackers will have the value of multiplexer selector and from this value, the binary bit for each state of LFSR 2 will be determined. To make it easy, build a truth table of LFSR 1 and LFSR 2 as shown in Figure 9. Then, put the keystream bit by bit to each stage of LFSR 2 depending on the selector. Figure 9 shows that every row of truth table filled with the keystream. Finally, fill the truth table for LFSR2. From the truth table of LFSR 2, another secret or private key will appear and the algorithm is not secure anymore. As a conclusion, the real strength of multiplexer algorithm depends on LFSR 1 and in this case, instead of 2^7, the real strength of multiplexer algorithm is 2^3. With reference to common key 64 bit, the strength of Multiplexing Generator is 2^{29}, not 2^{64} as claimed.

X^3	X^2	X^1
1	1	1
1	1	0
1	0	0
0	0	1
0	1	0
1	0	1
0	1	1
1	1	1

(a) LFSR 1

X^4	X^3	X^2	X^1
1			
1			
	1		
			0
		0	
	0		
		0	
1			

(b) LFSR 2

Figure 9 Truth table for both LFSRs

6.0 RESULTS

Basically, the strength of stream cipher correlates with the size of key used. Although the algorithms is not good but with the large key used and excellent key management scheme, it could increase their performance. In practice, long messages are not transmitted to avoid Possibility of Intercept (POI) and on average, data format for military standard is around 4000 bits [18]. Therefore, with the constant key length which is 64 bits, and 10 000 bits sequence, the strength comparison of stream cipher was made. For statistical test, 1000 different initial conditions were chosen randomly and all the results are shown in Table 1.

Table 1 presents the number of keys that passed for each statistical test. Based on 95% significant level, a good generator should passes at least 950 over 1000 initial conditions or keys. This is true only for self-shrinking generator. Thus, the probability of choosing a set of keys that generate a statistically random sequence is higher for this generator as compared to the linear, gaffe, multiplexing, both summation registers (Z_0 and c_{out} output), XOR-shrinking, and both memory (16 and 64 addresses) generator. Even though self-shrinking has good statistical properties but on the average, the sequence length of LFSR to produce the same amount of keystream with other generators is double. This is because the characteristic of self-shrinking itself that needs 2 bits or more to produce 1 bit of keystream. Therefore, the processing time to produce keystream also increases.

For memory generators, the size of memory obviously influences their randomness. A large memory size will present a better keystream compared with smaller memory due to statistical characteristics. From the table, both summation 2 and 3 registers (Z_0 output) give a good statistical result as compared to both summation 2 and 3 registers (C_{out} output). Summation 3 register (Z_0 output) presents a better statistical characteristic as compared to summation 2 registers (Z_0 output). From this observation, the number of register used in this generator influences their randomness.

Table 1 Results for each test of keystream generators

No.	**Generator**	**Freq. test**	**Serial test**	**Poker test**	**Auto. test**	**Runs test**	**Corr. attacks**	**LCP (Max)**	**GD. attack**
1	Linear	830/1000	941/1000	1000/1000	18522/19000	920/1000	56/60 immune	71	2^{42}
2	Geffe	956/1000	818/1000	732/1000	17962/19000	800/1000	0/60 immune	1665	2^{48}
3	Sum 2 reg.(z_0 output)	986/1000	975/1000	979/1000	18002/19000	863/1000	25/40 immune	5000	2^{29}
4	Sum 2 reg. (C_{out} output)	689/1000	760/1000	728/1000	18231/19000	981/1000	3/40 immune	2378	2^{29}
5	Sum 3 reg.(z_0 output)	1000/1000	984/1000	1000/1000	16748/19000	915/1000	54/60 immune	5000	2^{42}
6	Sum 3 reg (C_{out} output)	0/1000	0/1000	0/1000	1000/19000	691/1000	5/60 immune	2556	2^{42}
7	Multiplexing	963/1000	0/1000	0/1000	15562/19000	865/1000	19/40 immune	5000	2^{29}
8	Shrinking	992/1000	739/1000	981/1000	17316/19000	933/1000	38/40 immune	5000	2^{29}
9	XOR-Shrinking	987/1000	992/1000	663/1000	17726/19000	961/1000	38/40 immune	5000	2^{64}
10	**Self-Shrinking**	**998/1000**	**989/1000**	**993/1000**	**18578/19000**	**974/1000**	**19/20 immune**	**5000**	$\mathbf{2^{64}}$
11	Memory gen. (16 add)	842/1000	0/1000	0/1000	2082/19000	579/1000	33/60 immune	5000	2^{64}
12	Memory gen. (64 add)	882/1000	636/1000	994/1000	12360/19000	937/1000	50/60 immune	5000	2^{64}

The XOR-Shrinking generator, which was modified by the researcher also presented a better statistical results compared with the original Shrinking generator. However, it still did not passed 95% significant level.

For correlation attacks, linear generator, summations 3 registers (Z_0 output), shrinking, XOR-shrinking, self-shrinking and memory generators (64 addresses) present good performance where 80% of attacks are immune. For multiplexing, memory registers (16 addresses) and summation 2 registers (Z_0 output), half of the possible attacks were successful. Others generators were absolutely not immune and can be easily broken by this attack. Notice that the strength of generator will reduce to 2^N trials with N as the length of LFSR that leaks into keystream.

Linear Complexity Profile (LCP) is another standard test for measurement of strength and for this analysis, 10 000 bits keystream for each generator are used. Ideally, the LCP for each generator should be half of sequence length, which is 5000. From Table 2, linear generator, improved Geffe and summations register (c_{out} output) are not secure against LCP attack. Others are immune that have large LCP, which increase linearly with sequence length until 5000. Which means that, 10 000 bit keystream is not enough to find their equivalent LFSR. From the LCP test, a relation between LCP and key length can be derived but limited for certain generators such as linear generator and summation 2 registers (c_{out} output). Their relations are as follows:

(i) Linear generator: $L_{max} = N_0+N_1+N_2 = 19+23+29 = 71$

(ii) Summation 2 registers (c_{out} output): $L_{max} = 2N_0N_1 = 2(29)(41) = 2378$

A Guess-and-Determine attack is called nonstandard test because this performed attack depends on generator. Different generators will produce different GD attacks. As the key length is fixed to 64 bits, the best result for this attack is 2^{64}. There are 4 generators that are immune on this attack where their strength after the attack are 2^{64}, same as they claimed. They are XOR-shrinking, self-shrinking and both of memory generators. While for others, the strength of generators were reduced to 2^{29}, 2^{42} and 2^{48} respectively.

Self-shrinking generator that succeeded standard tests is immune of GD attacks. It is really robust and satisfies the properties of good stream cipher. However, it does not mean that it will be unbreakable. It could be broken with another nonstandard or mathematical attack, which is not included in this study.

7.0 CONCLUSIONS

The stream cipher is a type of cipher system where the process of enciphering and deciphering is performed one bit at a time. This type of cipher is useful in noisy channel conditions such as in radio communication environment for bulk data transmission. The basic theory for LFSR and the criteria for randomness are explained. The comparison of generator is performed on seven types of generator which are linear, improved Geffe, summation register, multiplexing, shrinking and memory generator. The analyses of their performance are made based on the statistical test, correlation attack, linear complexity profile and guess and determine attacks.

After the analyses, it is found that the only generator that succeeded all tests is self-shrinking. With the present performance of computer technology in terms of speed and memory space, this generator can be implemented easily on application such as messaging system, e-mail or any radio communication system. Due to its characteristic, this generator will guarantee the digital information exchange through radio communication is confidential. Other generators were shown to have their strength and weaknesses. However, it does not mean that they could not be used because the implementation choice depends on application and the information to protect.

REFERENCES

[1] Sailmail: Email Services for Yachts via Marine HF SSB Radio. 2004. http://www.sailmail.com/ (Accessed on 14/05/2005)

[2] CruiseEmail.com. 2004. http://www.cruiseemail.com/ (Accessed on 29/07/2005)

[3] Mils: The Secret of Unconditional Security. 2004. http://www.mils.com/ (Accessed on 06/06/2006)

[4] Crypto AG: For Communication & Information Security. 2004. http://www.crypto.ch/(Accessed on 14/09/2006)

[5] Nichols, K. and C. Lekkas. 2002. *Wireless Security: Models, Threats and Solution*. Canada:The McGraw-Hill Companies.

[6] Lail, M. 2002. *Broadband Network & Device Security*. Canada: The McGraw-Hill Companies.

[7] Schneier, B. 1996. *Applied Cryptography*. Minneapolis: John Wiley & Sons, Inc.

[8] Stallings, W. 2003. *Cryptography and Network Security*. New Jersey: Prentice Hall.

[9] Beker, H. and F. Piper. 1982. *Cipher Systems: Protection of Communication*. London: Northwood Publication.

[10] Press, W. and S. Teukolsky. 1992. *Numerical Recipes in C*. London: Cambridge University Press.

[11] Menezes, A. and P. Van oorshot. 1996. *Handbook of Applied Cryptography*. California: CRC Press.

[12] Golic, J. and M. Mihaljevic. 1990. Minimal Linear Equivalent Analysis of a Variable-Memory Binary Sequence Generator. *IEEE Transaction on Information Theory*. 36(1): 190-192.

[13] Golomb, S. 1967. *Shift Register Sequences*. UK: Holden Day.

[14] Johnson, R. 1992. *Elementary Statistics*. Boston: PWS-Kent.

[15] Siegenthaler, T. 1984. Correlation-immunity of Nonlinear Combining Functions for Cryptographic Applications. *IEEE Transactions on Information Theory*. 30: 776-780.

[16] Massey, J. L. 1969. Shift Register Synthesis and BCH Coding Decoding. *IEEE Transaction Information Theory*. 15: 122-127.

[17] Hawkes, P. and G. G. Rose. 2002. *Guess and Determine Attacks on SNOW*. Sydney: Qualcomm Australia.

[18] ACP 127(G). 1988. *Communication Instructions Tape Relay Procedures*. Allied Communication Publication.