IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 2, Ver. VIII (Mar-Apr. 2014), PP 72-78
www.iosrjournals.org

Dynamic Stream Ciphering Algorithm

[1]Mohammed Mobark Salih, [2]Mohammad G. S. Al-Safi, [3]Faiaz Hassan Ali
[1]Rafidain University College Computer Engineering Techniques
[2]Al-Esraa Uinversity College
[3]Al-Mustansiria University, College of Science Math. Dept.

Abstract *The main idea of any stream cipher algorithm is to generate stream cipher key base on the use set of LFSR with fix arrangement, all this LFSR are filling depending on the value of the basic key. In this paper we implement new technique base on dynamic stream cipher algorithm. In this algorithm we implemented dynamic stream cipher algorithm which base on idea of changing the structure of the LFSR with each change in BK and MK to get complex ciphering algorithm this is done by use a bank of LFSR store in file and we select random 10 register that is used in algorithm to generate the key. We implement Basic Efficient Criteria on Key Generator (KG) to test the result which is store in binary files. Three sample of key generation (KG) store in the binary file are test and all the sample is pass the test.*

I. INTRODUCTION

The security of GSM conversation is based on usage of the A5 family of stream ciphers. Many hundred million customers in Europe are protected from over-the-air piracy by the stronger version in this family, the A5/1 stream cipher. Other customers on other markets use the weaker version, A5/2. The approximate design of A5/1 leaked in 1994, and in 1999 the exact design of both A5/1 and A5/2 was discovered by Briceno [1]. A lot of investigations of the A5 stream ciphers followed.

In 1987, Ron Rivest from RSA Data Security, Inc. made a design of a byte oriented stream cipher called RC4 [2]. This cipher found its application in many Internet and security protocols. The design was kept secret up to 1994, when the alleged specification of RC4 was leaked for the first time [3]. Since that time many cryptanalysis attempts have been done on RC4 [4, 5, and 6].

The stream cipher Grain was developed by a group of researchers M. Hell, T. Johansson, and W. Meier, and was especially designed for being very small and fast in hardware implementation. It uses the key of length 80 bits and the IV is 64 bits, its internal state is of size 160 bits. Grain uses a nonlinear feedback shift register (NLFSR) and a linear feedback shift register (LFSR), and the idea to use NLFSR is quite new in modern cryptography [7].

Scream was developed by the IBM researchers Coppersmith, Halevi, and Jutla in 2002 [8]. It is a purely software-oriented stream cipher. The design is based on the ideas behind the SEAL stream cipher [19],

but considered to be more secure. The so-called “toy cipher” denoted Scream0 uses the AES S-box whereas the Scream stream cipher uses secret S-boxes, generated by the key.

At FSE 2004, a new stream cipher called VMPC [9] was proposed by Bartosz Zoltak, which appeared to be a modification of the RC4 stream cipher. In cryptanalysis, a linear distinguishing attack is one of the most common attacks on stream ciphers. In paper [10] it was claimed that VMPC is designed especially to resist distinguishing attacks.

At the same conference, FSE 2004, another cipher, RC4A [10], was proposed by Souradyuti Paul and Bart Preneel. This cipher is another modification of RC4.

Recently, a new European project eSTREAM [11] has started, and at the first stage of the project 35 new proposals were received by May 2005. Although many previous stream ciphers were broken, collected cryptanalysis experience allowed strengthening new proposals significantly, and there are many of them that are strong against different kinds of attacks. One such good proposal was the new stream cipher Grain.

Dragon is a word oriented stream cipher [12] submitted to the eSTREAM project, designed by a group of researchers, Ed Dawson et al. It is a word oriented stream cipher that operates on key sizes of 128 and 256 bits. The original idea of the design is to use a nonlinear feedback shift register (NLFSR) and a linear part (counter), combined by a filter function to generate a new state of the NLFSR and produce the keystream. The internal state of the cipher is 1088 bits, which is updated by a nonlinear function denoted by F. This function is also used as a filter function producing the keystream. The idea to use a NLFSR is quite modern, and there are not many cryptanalysis techniques on NLFSRs yet developed.

Ali F. H. [13] introduces the mathematical process to generate a sequence from two generators of The Multiplicative Cyclic Group (MCG). The two generators with some initial variables (keys) make a unit called MCG unit. A number of MCG units are combined with each other by a combining logical function to get MCG system.

Ali. F. H. and Mahmoud A. G. [14], in there paper develop the MCG unit to gain random digital sequences with high efficiency to be used in stream cipher systems.

The main goal of this paper is to construct a new stream cipher system generator, generates good statistical properties digital sequences could be used in cryptography. The proposed generator considered as a stream cipher system which is depends on linear feedback Shift Registers (LFSR's). The LFSR unit considered a basic unit which the stream cipher systems depend on.

The four basic efficiency criteria, periodicity, linear complexity, randomness and correlation immunity are applied to measure the efficiency of the pseudo random sequences which are generated from the proposed generator.

II. STREAM CIPHERS SYSTEMS

Stream ciphers are an important class of encryption algorithms. They encrypt individual characters (usually binary digits) of a plaintext message one at a time, using an encryption transformation which varies with time [15]. The main properties of stream ciphers separating them from block ciphers are that the encryption function works on individual symbols (letters) of the underlying alphabet and that the encryption function is time-varying [16].

Stream ciphers have extensive applications; many of them are in the area of wireless communication. As an example, they are part of the security framework in GSM networks, Bluetooth or WLANs [17].

In stream ciphers, the message units are bits, and the key is usual produced by a random bit generator (see figure (1)). The plaintext is encrypted on a bit-by-bit basis.

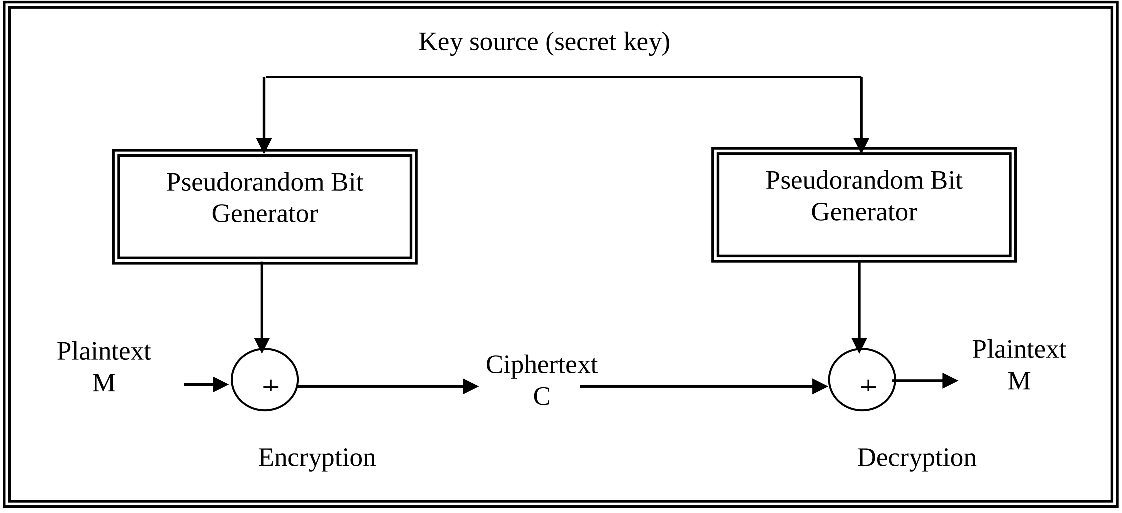


Figure (1) Stream Cipher System.

The key is fed into random bit generator to create a long sequence of binary signals. This “key-stream” k is then mixed with plaintext m, usually by a bit wise XOR (Exclusive-OR modulo 2 addition) to produce the cipher text stream, using the same random bit generator and seed.

Stream ciphers are generally faster than block ciphers in hardware, and have less complex hardware circuitry. They are also more appropriate, and in some cases mandatory (e.g., in some telecommunications applications), when buffering is limited or when characters must be individually processed as they are received. Because they have limited or no error propagation, stream ciphers may also be advantageous in situations where transmission errors are highly probable.

The security of stream cipher is thus always measured relative to the complexity of exhaustive searching for the correct key. If the complexity of an attack is less than that of the exhaustive search, the cipher is said to be **broken** [16].

Another advantage of stream ciphers in military cryptograph is that the cipher stream can be generated in a separate box that is subject to strict security measures and fed to other devices, e.g. a radio set, which will perform the XOR operation as part of their function. The latter device can then be designed and used in less stringent environment [18].

III. IMPLEMENT OF DYNAMIC STREAM CIPHER KEY GENERATION

The key generation system consist of

1: Basic key (BK) of length 20 byte

2: Massage key (MK) of length 10 byte
3: Sysfill (160) it is sequence of bits of length 160 bits
4: Sysfill(4) it consist four LFSR of length 39 bits ,41 bits ,53 bits and 27 bits.
5: SysregA it consist of four register of variable length
6: SysregB it consist of four register of variable length
7: SysregC it consist of two register of variable length

The software implement in two part first part system fill register part and the second is key generation part.

The system fill register consist four model

First model is used to fill (MK) and (BK) in sequence of bits of length 160 bits . first we use 20 byte for BK from text key file and use 10 byte for MK it is randomly generate and it expanded to 20 byte. BM XORing with MK bits by bits to fill 160 sequence bits call sysfill(160 bits).

In Second model the sysfill (160 bits) used to fill four LFSR of length 39,41,53,27 this four LFSR call sysreg (4).

In Third model Sysreg (4) move many time to get 10 different number of mod 16 this number is used to select the random LFSR from the bank of register . Four of this register is used as sysreg A and the next four register is used as sysregB the last two register is used as sysreg C. After we select the register length and taping we move sysreg (4) number of time depended on the length of sysreg A, sysreg B and sysreg C and fill all register in sysreg A , sysreg B and sysreg C .

The last model used sysreg A and sysreg B to fill the RAM (256 byte) with unique data

In key generation part sysregA and systergB move one time we combined the output of LFSR to get byte used as address for RAM 256 byte and get byte from its call ROB (RAM Out Byte).

SysregC move four time and get byte from its four bits for each register call Cbyte the final key generation (KG) is

KG=ROB xor Cbyte

The result is store in binary file with different length to test its later.

IV. RESULT

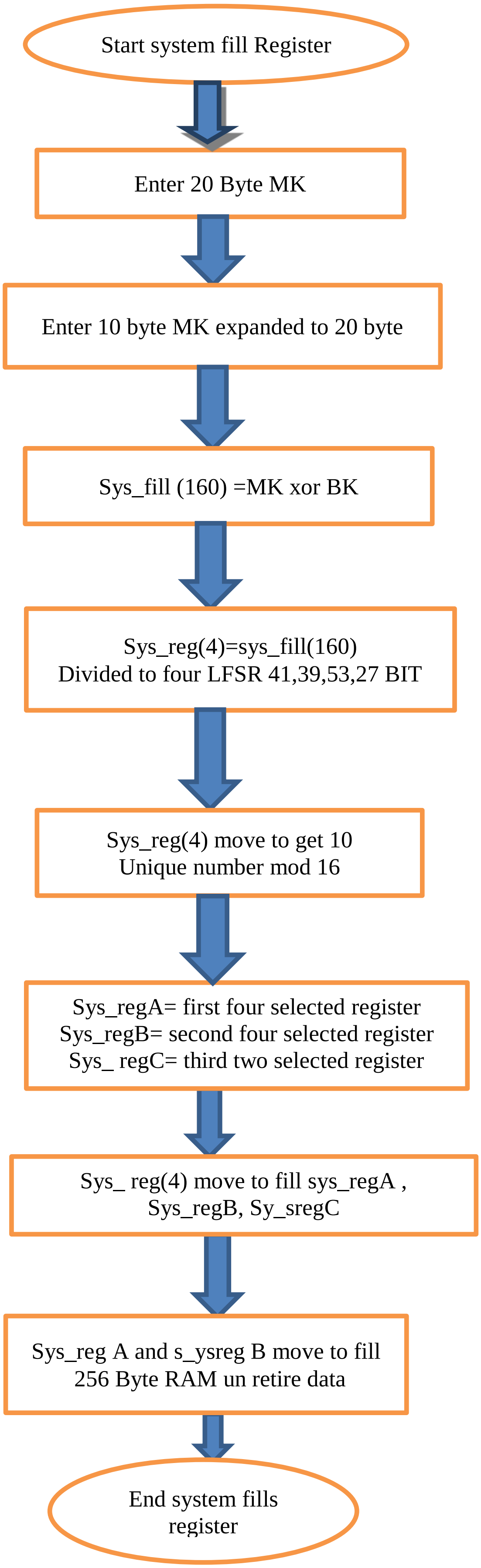
In this paper we run the program which is design in visual basic and store the result in three files of type binary with variable length and we use packet software to test the result this is done by Implement of Basic Efficient Criteria on Key Generator (KG)

1. Randomness Criterion (R)

This criterion depends on lengths LFSR's and CF. we expect that the output keys of this KG will be real random. We have to test the randomness of the KG three samples (test1, test2 and test3) each of L=20000, 25000 and 30000 bytes respectively.

CRYPT-X [19] package used to test the output results of KG tested by using group of tests. Table (1) shows the Frequency test, 1st and 2nd Binary Derivative test, While Change Point test shown in table (2), Subblock (Poker) shown in table (3) and lastly, Run tests described in table (4).

Fill system register


Start system fill Register
Enter 20 Byte MK
Enter 10 byte MK expanded to 20 byte
Sys_fill (160) =MK xor BK
Sys_reg(4)=sys_fill(160)
Divided to four LFSR 41,39,53,27 BIT
Sys_reg(4) move to get 10
Unique number mod 16
Sys_regA= first four selected register
Sys_regB= second four selected register
Sys_ regC= third two selected register
Sys_ reg(4) move to fill sys_regA ,
Sys_regB, Sy_sregC
Sys_reg A and s_ysreg B move to fill
256 Byte RAM un retire data
End system fills
register

Flow chart of key generation

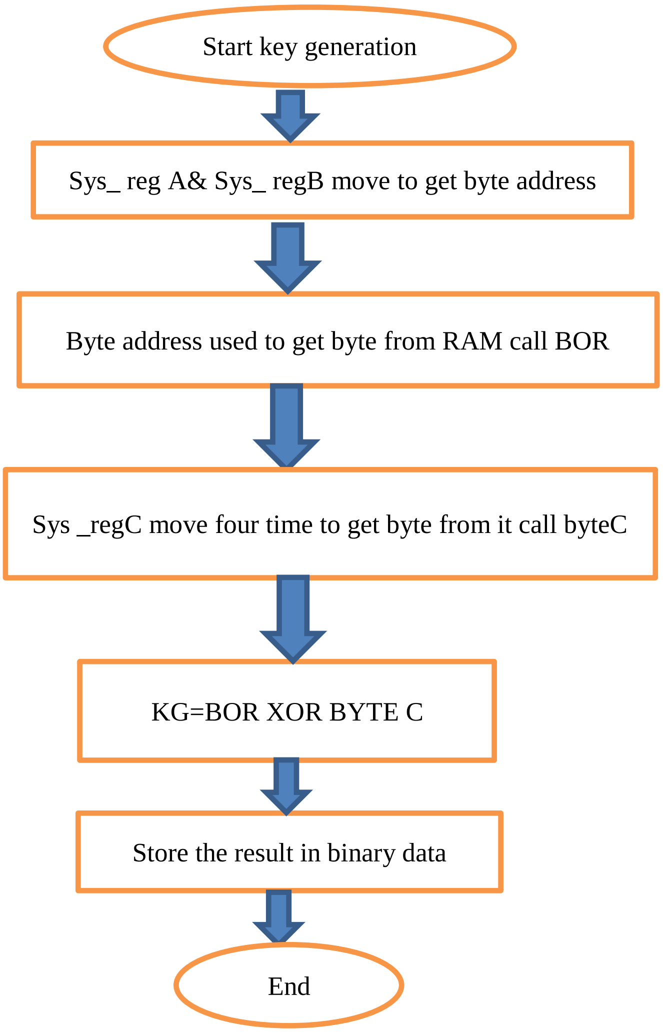

Start key generation
Sys_ reg A& Sys_ regB move to get byte address
Byte address used to get byte from RAM call BOR
Sys _regC move four time to get byte from it call byteC
KG=BOR XOR BYTE C
Store the result in binary data
End

Table (1) KG tested by frequency and binary derivative tests.

Test	Ex.	Length (bits)	n_1	mean(1)	prop(1)	α
Frequency	test1	160000	80185	8000.0	0.5012	0.3550
	test2	200000	100015	100000.0	0.5001	0.9465
	test3	240000	119587	120000.0	0.4983	0.0918
1st Binary Derivative	test1	159999	80103	79999.5	0.5006	0.6048
	test2	199999	99987	99999.5	0.4999	0.9554
	test3	239999	120067	119999.5	0.5003	0.5003
2nd Binary Derivative	test1	159998	80033	79999.0	0.5002	0.8650
	test2	199998	99745	99999.0	0.4987	0.2560
	test3	239998	120448	119999.0	0.5019	0.0668

The decision is that the outputs of the three examples with corresponding lengths pass the Frequency and the 1st and the 2nd Binary Derivative tests.

Table (2) KG tested by Change point test.

Ex.	Length	Change point (Cp)	n_1 before Cp	prop(1) before Cp	prop(1) after Cp	α
test1	160000	77873	39144	0.5027	0.4997	0.5017
test2	200000	55720	27743	0.4979	0.5009	0.5558
test3	240000	37352	18752	0.5020	0.4976	0.5190

The decision is that the outputs of the three examples with corresponding lengths pass the Change point test.

Table (3) KG tested by subblock tests.

Test	Ex.	Length	Size	χ^2-value	DF	α
Subblock	test1	160000	2	3.2809	3	0.3503
			4	9.4936	15	0.8503
			8	237.5680	255	0.7774
	test2	200000	2	0.0463	3	0.9974
			4	13.7683	15	0.5432
			8	216.0205	255	0.9621
	test3	240000	2	5.3815	3	0.1459
			4	13.2080	15	0.5862
			8	252.3221	255	0.5385

The decision is that the outputs of the three examples with corresponding lengths pass the Subblock test.

Table (4) KG tested by Run tests.

Test	Ex.	Length	No. of Runs	No. of Blocks	No. of Gaps	χ^2-value	DF	α
Run	test1	160000	80104	40052	40052	9.7396	26	0.9984
	test2	200000	99988	49994	49994	25.4291	28	0.6044
	test3	240000	120068	60034	60034	23.4546	28	0.7100

The decision is that the outputs of the three examples with corresponding lengths pass the Run test.

Where $\mathbf{n_1}$ denotes the number of ones, **mean(1)** is the mean of ones, **prop(1)** is the proportion of ones, χ^2**-value** is the chi square value, **DF** is the degree of freedom and lastly, α is the tail-area probability or can be called significance probability, **.**

2. Periodicity Criterion (Per)

As known that this criterion depends on LFSR length only, since we have (10) LFSRs, let ri be the length of LFSR i, 1≤i≤10, so:

= lcm (LFSR1,LFSR2,…,LFSR10)

Per(KG)=lcm (2^{r_1} -1, 2^{r_2} -1,…, $2^{r_{10}}$ -1) $\approx 2^{\sum r_i}$ …(1)

This mean that the output key sequence of KG semi impossible to repeat itself.

3. Complexity Criterion (C)

This criterion depends on the length of combined LFSR's and the CF of the generator. This criterion divided into two parts, the first part represented by the General complexity (GC) and the second is the linear complexity (LC) of KG.

For General Complexity of KG

GC(KG) = 2160 …(2)

Where 160(=8*20) represent the size of key space in bits

Now BerliCamp-Massey algorithm [2] will applied to estimate the LC of KG, Table (5) shows LC(KG) for various sequences length from the two outputs.

From equation (2) we can judge that if the cryptanalyst used brute force (exhaustive search) attack he will need for hundreds of years to extract the plaintext.

Table (5) LC(KG) of various sequences length.

Ex.	Length in bits	LC(KG)
1	4000	1993
2	10000	5003
3	20000	9983

The results of LC(KG) in table (5), shows that the length of the minimum equivalent LFSR which generates the output sequence ≈ (1/2) length of tested sequence.

4. Correlation Immunity Criterion (CI)

This criterion depends on CF only. If we consider that the RAM is high nonlinear function, we expect to gain very weak correlation, especially when adding the output byte of the balance subsystem to the output byte of RAM.

Naturally, the same results can be obtained by using the real time run of the generator. Table (6) shows these results from various comparison sequences lengths.

Table (6) CP and CI results for various sequences lengths.

Ex.	Length in bits	CP								CI
		x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	
1	8000	0.494	0.497	0.500	0.498	0.499	0.513	0.509	0.491	8
2	16000	0.499	0.499	0.502	0.498	0.499	0.489	0.503	0.483	8
3	24000	0.522	0.483	0.495	0.506	0.513	0.488	0.521	0.490	8

From table (6), we conclude the output variables of all ten LFSRs are immune against correlation or fast correlation attack [3].

V. CONCLUSION

New technique is implemented base on dynamic stream cipher algorithm. We implemented dynamic stream cipher algorithm which base on idea of changing the structure of the LFSR with each change in BK and MK to get complex ciphering algorithm. This is done by use a bank of LFSR store in file and we select random 10 register that is used in algorithm to generate the key. We implement Basic Efficient Criteria on Key Generator (KG) to test the result which is store in binary files.as we saw in our work in this paper we can enhancement the key generation by increasing the number of the LFSR and also we can modify this generator the initial LFSR is not fix structure and it reconstruction from BK and MK.

References

[1]. **Briceno**, M., Goldberg, I., and Wagner, D., "***A Pedagogical Implementation of A5/1***", Available at http://jya.com/a51-pi.htm (accessed August 18, 2003), 1999.

[2]. **Smart**, N., "***Cryptography: An Introduction***", McGraw-Hill Education, 2003. ISBN 0-077-09987-7.

[3]. **Schneier**, B., "***Applied Cryptography: Protocols, Algorithms, and Source Code in C***", John Wiley & Sons, New York, 2nd edition, 1996.

[4]. **Fluhrer**, S. R. and McGrew, D. A., "***Statistical Analysis of the Alleged RC4 Keystream Generator***", In B. Schneier, editor, Fast Software Encryption 2000, volume 1978 of Lecture Notes in Computer Science, pages 19–30. Springer-Verlag, 2000.

[5]. **Mantin**, I. and Shamir, A., "***Practical Attack on Broadcast RC4***", In M. Matsui, editor, Fast Software Encryption 2001, volume 2355 of Lecture Notes in Computer Science, pages 152–164. Springer- Verlag, 2001.

[6]. **Paul**, S. and Prenel, B., "***Analysis of non-fortuitous predictive States of the RC4 Keystream Generator***", In T. Johansson and S. Maitra, editors, Progress in Cryptology—INDOCRYPT 2003, volume 2904 of Lecture Notes in Computer Science, pages 52–67. Springer-Verlag, 2003.

[7]. **Johansson**, T. and Jönsson, F., "***On the Complexity of Some Cryptographic Problems Based on the General Decoding Problem***", IEEE Transactions on Information Theory, 48(10): 2669–2678, 2002.

[8]. **Halevi**, S., Coppersmith, D., and Jutla, C. S., "***Scream: A Software Efficient Stream Cipher***", In J. Daemen and V. Rijmen, editors, Fast Software Encryption 2002, volume 2365 of Lecture Notes in Computer Science, pages 195–209. Springer-Verlag, 2002.

[9]. **Zoltak**, B., "***VMPC one-way function and stream cipher***", In B. Roy and W. Meier, editors, Fast Software Encryption 2004, volume 3017 of Lecture Notes in Computer Science, pages 210–225. Springer-Verlag, 2004.

[10]. **Paul**, S. and Prenel, B., "***A New Weakness in the RC4 Keystream Generator and an Approach to Improve the Security of the Cipher***", In B. Roy and W. Meier, editors, Fast Software Encryption 2004, volume 3017 of Lecture Notes in Computer Science, pages 245–259. Springer-Verlag, 2004.

[11]. **eSTREAM:** ECRYPT Stream Cipher Project, IST-2002-507932. Available at http://www.ecrypt.eu.org/stream/ (accessed September 29, 2005), 2005.

[12]. **Chen**, K. and et al, "***Dragon: A Fast Word Based Stream Cipher***", eSTREAM, ECRYPT Stream Cipher Project, Report 2005/006 (2005-04-29), 2005.

[13]. Ali, F. H., "***Use the Multiplicative Cyclic Group to Generate Pseudo Random Digital Sequences***", Journal of Al-Rafidain University College for Sciences, Vol.20, pp.122-135, 2006.

[14]. Ali, F. H. and Mahmoud A. G., "***High Efficient Sequences Generate from Developed MCG Generator***", the 10th Scientific Conference, Al-Rafidian University College No#. 25, pp.169-182, 21, Nov., 2009.

[15]. Menezes, A. P. van Oorschot, P. and Vanstone, S., "***Handbook of Applied Cryptography*"**, CRC Press, 1996.

[16]. Ekdhal, P., "*On LFSR based Stream Ciphers Analysis and Design*", Ph.D. Thesis, November 21, 2003.

[17]. Rechberger, C., "***Side Channel Analysis of Stream Ciphers***", Master's Thesis, Institute for Applied Information Processing and Communications (IAIK) Graz University of Technology, Graz, Austria, 2004.

[18]. Matt, J. B., "***Stream Ciphers Technical Report TR-701***", RSA Laboratories, 1995

[19]. Phillip Rogaway and Don Coppersmith , SEAL (Software-Optimized Encryption Algorithm) 1994.