ISSN: 0067-2904

Design of New Dynamic Cryptosystem with High Software Protection

Abdullah Ayad Ghazi[1*], Faez Hassan Ali[2]
[1]Department of Mathematics, College of Science, Baghdad University, Baghdad, Iraq
[2]Department of Mathematics, College of Science, Al-Mustansiryah University, Baghdad, Iraq

Abstract

In this paper a design of new robust and dynamic cryptosystem using stream key generator with good random statistical properties is introduced. The results of the statistical tests, initialization and the components of the proposed generator of the suggested cryptosystem are detailed in [1]. The proposed cryptosystem called **Robust and Efficient Dynamic Stream Cipher Cryptosystem (RDSCC)**. The dynamic here means the generator of RDSCC has unfixed construction; the design will be changed as the seed (initial) key changed. The generator will construct itself by barrows number of Linear Feedback Shift Registers (LFSR's), randomly, stored in protected bank file. The RDSCC is constructed to be flexible to encrypt (decrypt) different types of data; like message text and image data efficiently. Also we suggest using software protection for RDSCC which is represented by partitioned the cryptosystem file into two parts, each one saved in different memories to keep it in save from any attack. Lastly, the proposed generator and the cryptosystem are implemented practically and the main interfaces of the two programs are shown in this paper. These interfaces are programmed with GUI MATLAB2017.

تصميم نظام تشفير ديناميكي جديد مع حماية برمجية عالية الكفاءة

عبدالله اياد غازي[*1]، فائز حسن علي[2]
[1]قسم الرياضيات، كلية العلوم، جامعة بغداد، بغداد، العراق
[2]قسم الرياضيات، كلية العلوم، الجامعة المستنصرية، بغداد، العراق

خلاصة

هذا البحث يقدم تصميم نظام تشفير جديد رصين وديناميكي باستخدام مولد مفتاح انسيابي يتمتع بخواص إحصائية عشوائية جيدة. تم عرض النتائج التفصيلية للخواص الاحصائية وعملية الاملاء مع مكونات المولد المقترح للنظام في المصدر [1]. نظام التشفير المقترح سمي نظام التشفير الانسيابي الرصين والديناميكي الكفوء (**RDSCC**). الديناميكية هنا تعني ان بناء المولد الخاص بنظام RDSCC غير ثابت، وستتغير طبيعة التصميم للمولد بتغير مفتاح القيم الابتدائية للنظام، حيث سيقوم المولد ببناء نفسه من خلال استعارة عدد من المسجلات الزاحفة الخطية ذو التغذية الخلفية الراجعة وبشكل عشوائي من ملف بنك محمي خاص بالمسجلات الزاحفة. لقد تم بناء نظام RDSCC بشكل مرن لتشفير (حل) انواع مختلفة من البيانات، مثل رسائل نصية وبيانات صور وبشكل كفوء. لقد تم اقتراح توفير حماية برمجية للنظام RDSCC، حيث تمثل بتجزئة ملف النظام الى جزئين وحفظ كل جزء منهم في ذاكرة مختلفة لحفظه من اي نوع من انواع المهاجمة. واخيرا، فقد تم تنفيذ برنامجي المولد والنظام المقترح بشكل عملي وتم عرض الواجهات الرئيسية لهما في البحث. تلك الواجهات تم برمجتها باستخدام نظام GUI المتوفرة في لغة ماتلاب 2017.

*Email: faezhassan@uomustansiriyah.edu.iq

1. Introduction

A stream cipher is a type of symmetric cryptosystem in which the plaintext divided into small entities called characters and encrypt one character at time the plaintext is encoded on bit by bit. Stream Cipher systems, 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.

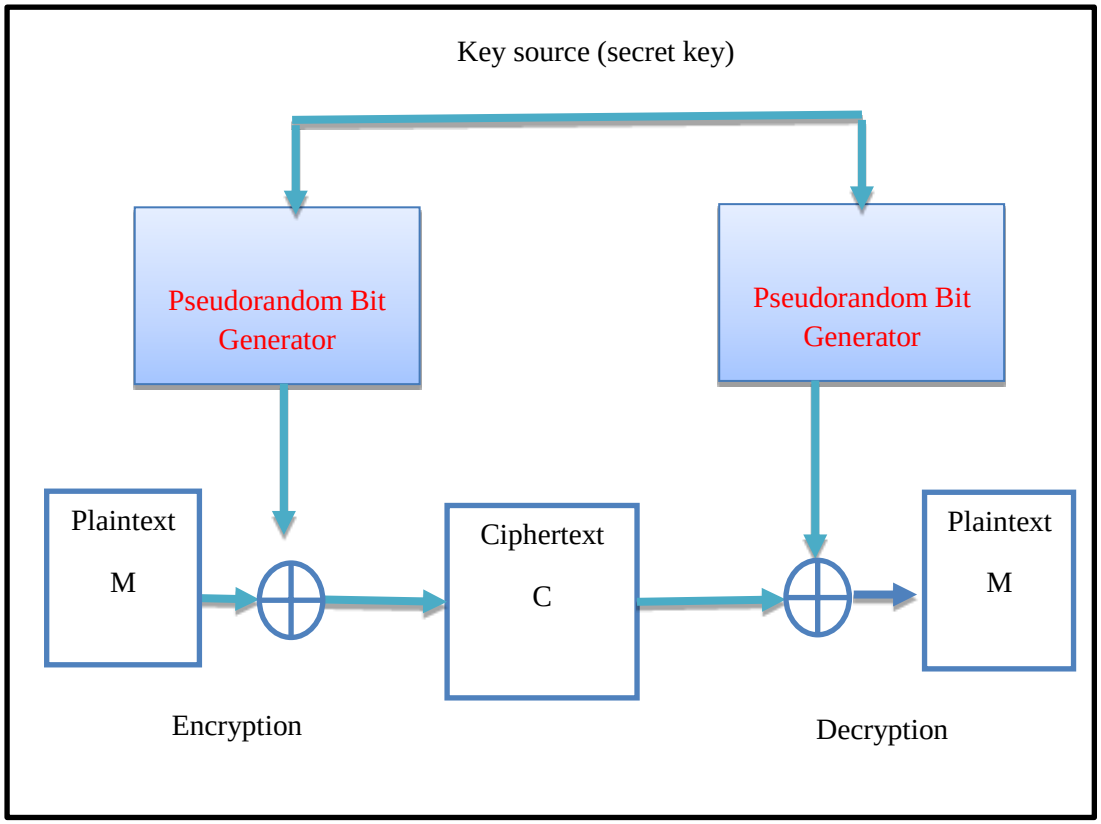

Figure1-Stream Cipher System

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 ciphertext stream, using the same random bit generator and seed.

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** [2].

In military cryptograph is that the cipher stream can be generated in a separate box that is subjected 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 [3].

A block cipher, it's an algorithm that depart an a plaintext to blocks, and then encrypted every block of plaintext to block of ciphertext of the same length. An n-bit block cipher divides the plaintext into blocks of n-bits and encrypts one block at a time with a fixed key dependent invertible transformations, in practice the block size n is often 64 or 128.

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 [4]

The stream cipher as usual depends on its structure in what we call Linear Feed Back Shift Register (LFSR). This component is combined of two parts: a shift register (SR) and a feedback polynomial (function). The SR considered a sequence of bits, (the length of a SR is figured in bits). In each time a single bit is needed, all of the bits in the SR are shifted 1 bit to the right[5]

In 1997, there are many variants of Clock Controlled Generators [6], but the basic idea is the following: one or more LFSR's control the clock of one or more other LFSR's. A simple variant is the Stop-and-Go generator. It consists of two registers, whereas one register clocks the other if the output is 1, otherwise the previous output is repeated. As in example, the first LFSR1 decides whether LFSR2 or LFSR3 should be clocked.

Salih M. M, Mohammad G. S. Al-Safi and Faez Hassan Ali (2014) [7], in their paper implement a new technique base on dynamic stream cipher algorithm, based on idea of changing the structure of the LFSR with each change in BK and MK to get complex ciphering algorithm such that selecting random 10 registers that is used in algorithm to generate the key. The Basic Efficient Criteria on Key Generator to test the results is implemented.

In 2017, Al-Zewary R. M., introduced a new stream cipher key generator called a High Efficiency Non-linear Shift Register Generator (HENSRG) which passes all basic efficiency criteria [8].

In this paper, we will construct a new cryptosystem depends on a dynamic key generator, this key generator is detailed in [1]. So, we shouldn't have to mention the design of this key generator as much as we need for constructing the suggested cryptosystem. In section 2, Constructing of the New Dynamic Cryptosystem will be described. In section 3, the Encryption and Decryption Process using RDSCC will be described. In section 4, RDSCC Protection will be described.

2. Constructing of the New Dynamic Cryptosystem

Before we describe the main steps of the proposed cryptosystem which we called it Robust and Dynamic Stream Cipher Cryptosystem (RDSCC) we have to show some relevant details in this constructing.

2.1 Key Management of RDSCC

The proposed system can work in two key issues, when there is a center with many (n) stations:

1. **General Information Key issue (GIK)**: this issue includes a general information want to be sent from center to its related stations in one way sending. This means the center use its key for encryption only while the stations use it for decryption only, such that $GIK_1=GIK_2=\ldots=GIK_n$. As usual, the information which is sent in one way is represented by instruction, orders,…, etc.
2. **Special Information Key issue (SIK)**: This issue includes transmitting the information between the center and each station related to the center in two ways sending. Of course, the message transmitted from center and station i is encrypted in key different from the key used to encrypt message from station i to the center, $1\leq i\leq n$ and the $SIK^s_i\neq SIK^r_i$ (send (s) or receive (r)).

Figure-2 shows the key management of RDSCC for n stations, notice that the number of keys for GIK is n, while for SIK is 2n.

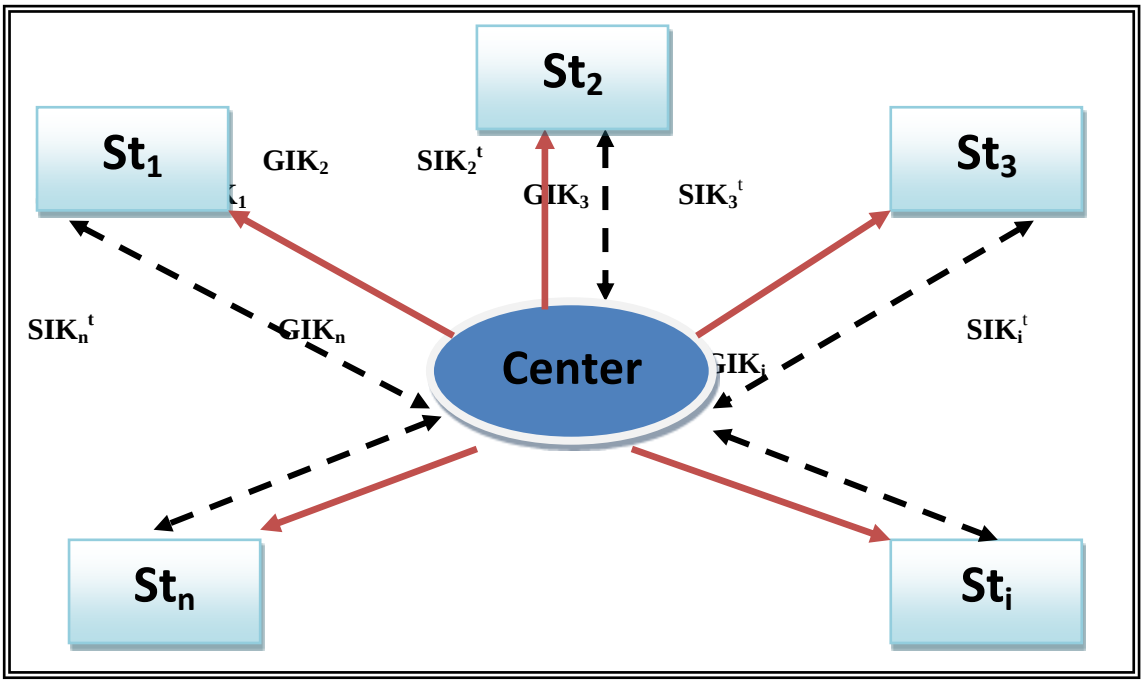

Figure 2- Key Management of RDSCC for n stations.

In order to fill the system with the seed key, we suggest using two types of keys that are designed as an initial key for LFSRs of the RDSCC, these keys are:

1. **Message Key (MK)**: This key is non-secret which consists of (10) **ASCII CODE (8bits)** characters. A new MK can be used with every new message to guarantee that no two messages have the same key [1].
2. **Basic Key (BK):** This key consists of (20) **ASCII CODE (8bits)** characters and it's changed daily (or with each message) [1].

2.2 Bank of LFSR's

This bank considered as an information file includes 16 different LFSR's with different connection polynomials and different lengths. Supply the proposed key generator by the required numbers of LFSR's for MS and BS systems. This file is stored in special memory and protected by a secret key which is part of the Basic key.

The basic component, Initialization, Moving and the main algorithm of RDSCC are mentioned in [1].

RDSCC Algorithm

Step (1): input BK (20 chrs), MK (10 chrs), length L bytes, text_input (plain/cipher);
Step (2): BK and MK mixed to initialize IS.
Step (3): Fill CSR system by IS to specify the shift registers of MS and BS from LFSRs bank.
Step (4): Move IS to fill MS, BS, and VMU.
Step (5): For i=1:L
AD=MS(x_j); j=1,..,8.
Byte1=VMU(AD);
ASR=BS(y_j,p_j); j=1,..,4.
Byte2=Move(ASR);
Key(i)=Byte1⊕Byte2;
Text_output(i)=Key(i)⊕text_input(i);
End {for i}
Step (6): Output text (cipher/plain);
End
The block diagram of RDSCC algorithm is described in Figure-4.

3. The Encryption and Decryption Process using RDSCC

The GUI of the BEC program for all output key tests are described in Figure-3.

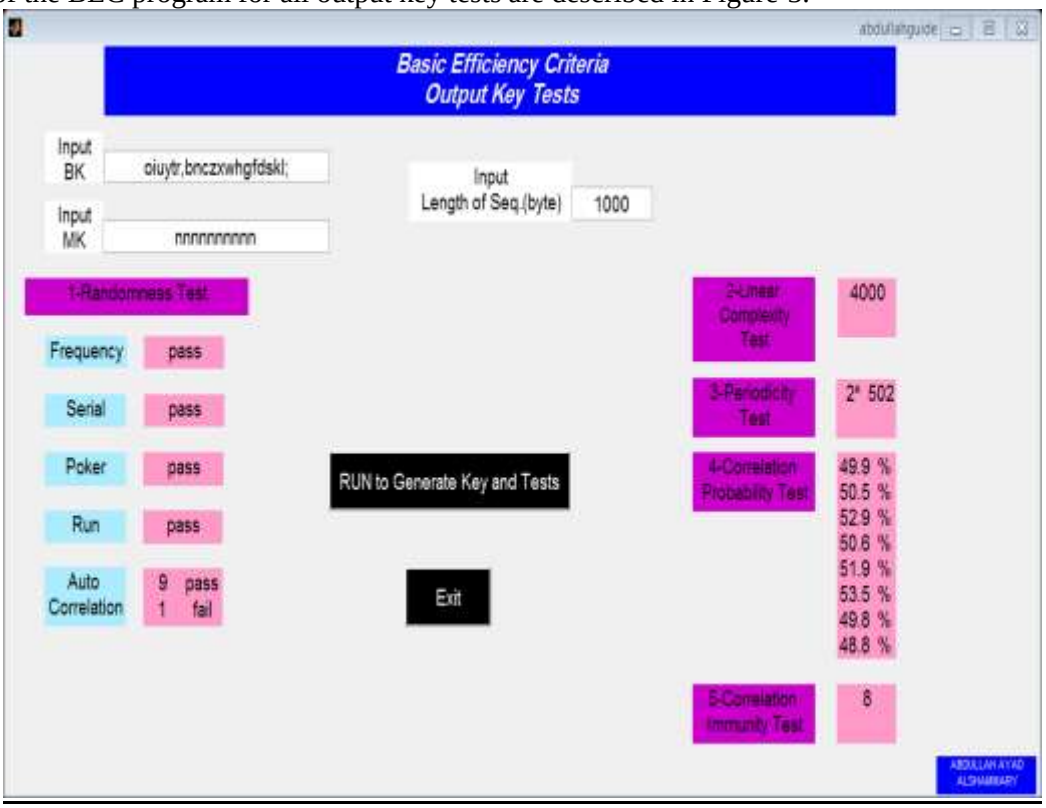

Figure 3- the block diagram for (BEC) output key tests.

A. The Encryption and Decryption process using RDSCC

In this section, we will discuss and show how to use the generated key from RDSCC algorithm to encrypt and decrypt messages or images? Firstly, after execute the RDSCC program, the main window of RDSCC program appear as in Figure-4. Secondly, input the plaintext or image we want to encrypt.

Figure 4-the main window of RDSCC program.

1. Encryption Process

If we have a text or an image that wants to be encrypted we press "Encryption" button to move to the "Encryption or Decryption" window as in Figure-5.

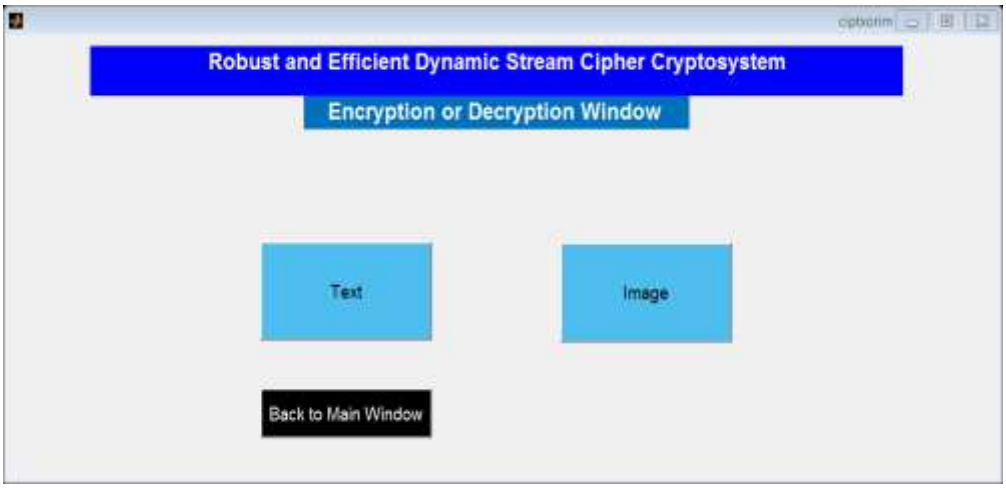

Figure 5- the Encryption window RDSCC program.

a) **Text Encryption**: If we have a plaintext message that wants to encrypt, first, input the BK and MK characters. Secondly, browse the plaintext file by pressing "Select File" button to show the text in the "Encryption text window" as in Figure-6. In order to start the encryption process, we must press on the "Start Encryption" button to obtain the encrypted text which is shown in "Encryption text window". Lastly, to return back to the main window we press the "Back to Main Window" button (see Figure in (7)).

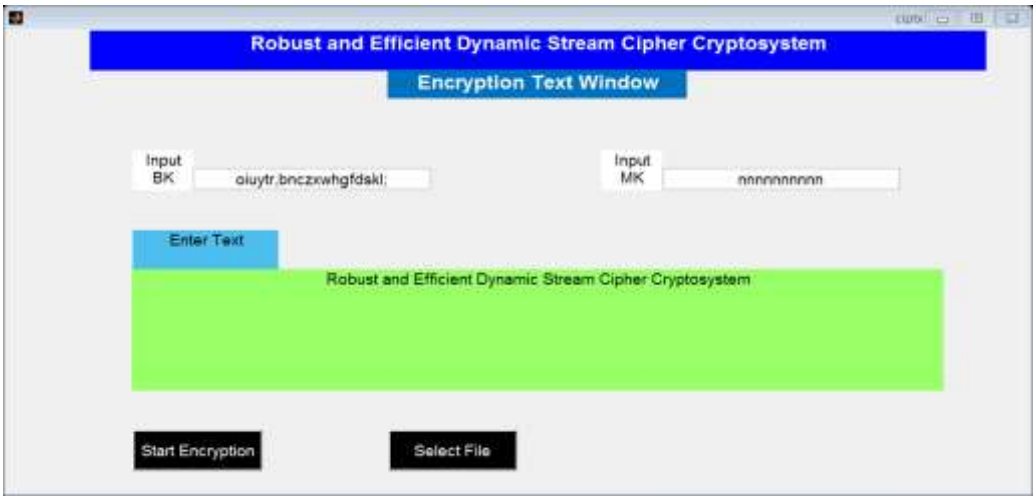

Figure 6-Encryption text widow of RDSCC program.

Figure 7-Encrypted text window of RDSCC program.

b) **Image Encryption**: If we have an image that wants to encrypt, first, input the BK and MK characters. Secondly, browse the image file by pressing "Select Image" button to show the image in the "Encryption Image Window". In order to start encryption process, we must press on "Start Encryption" button to obtain the encrypted image which is shown in "Encryption Image Window". After that click on "Back to Main Window" button to go back to the main window (see Figure-8)).
The cipher text (image) file will be sent to the other party.

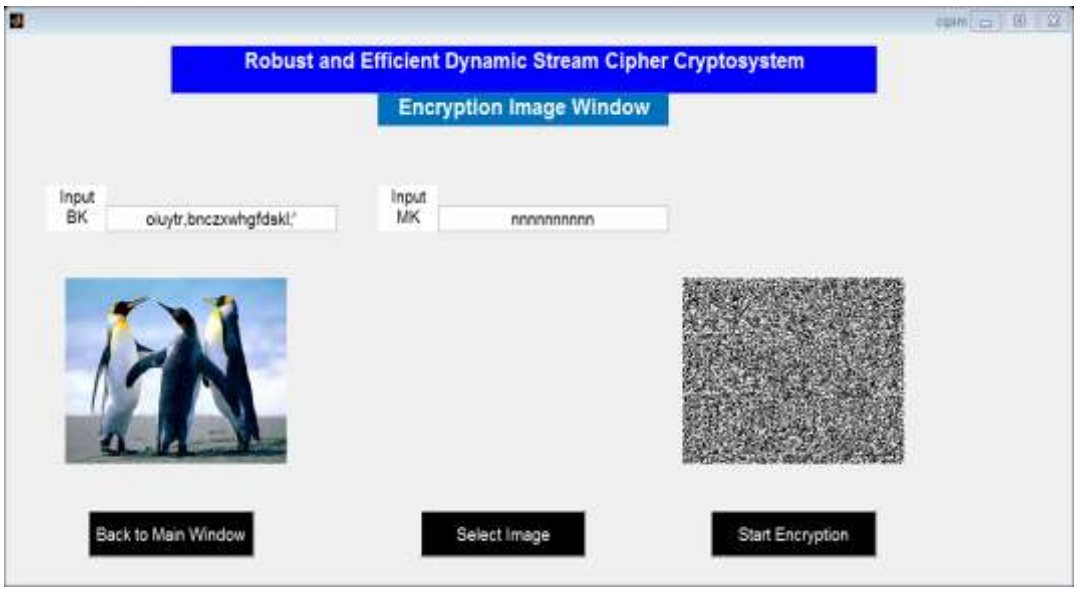

Figure 8-Encryption image window of RDSCC program.

2. Decryption Process

In the main window, we press the "Decryption process" button, to move to the "Encryption or Decryption window", as in Figure-5.

a) Text Decryption: If we have a cipher text message that wants to decrypt, first, input the BK and MK characters, secondly, browse the cipher text file by press the "Select and Start Decryption" start the decryption process, to obtain the decryption text which is shown in "Decrypted text window", (see Figure-9)).

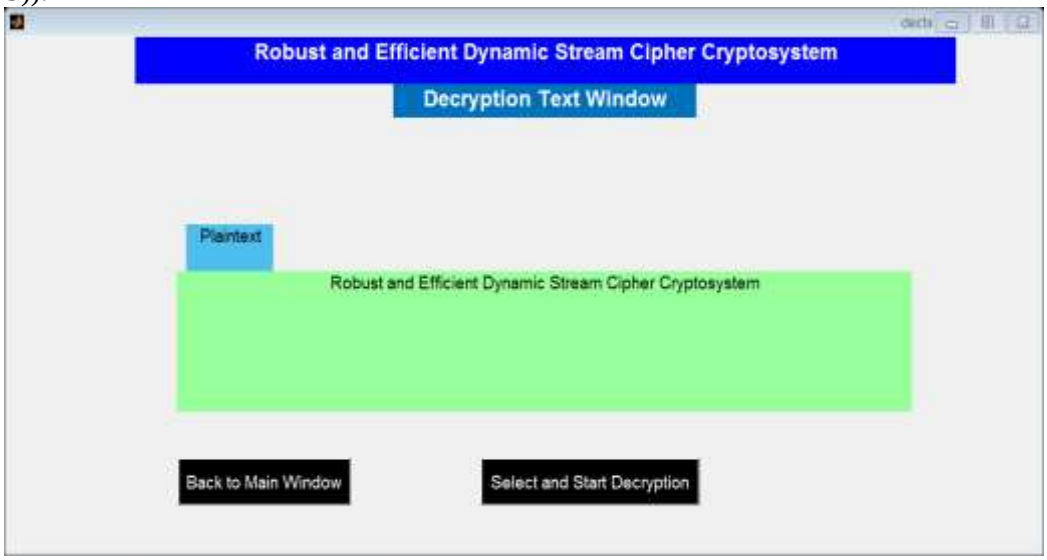

Figure 9-the block diagram of selecting Decryption Text or Decryption Image of RDSCC algorithm.

b) Image Decryption: If we have encrypted image want to decrypt, first, input the BK and MK characters. Secondly, browse the encrypted image file by pressing "Start Decryption" button. To start

the decryption process, to obtain the decrypted image which is shown in "Decryption Image Window", (see Figure-10).

Figure 10- decryption image window of RDSCC program.

4. RDSCC Protection

We know that the cryptosystem can implemented software or hardware, as usual if the implementation is hardware so the protection for the cryptosystem can be physical protection, while if the cryptosystem is software so it can be protected physically or by using some tools of software.

In this article, we suggest to use software protection to keep the algorithm or the program of the RDSCC secure form intruders or spices. The idea is in general, dived the program file of RDSCC into two parts; Part-A and Part-B. The Part-A saved in the same computer while the Part-B saved in any external unit, like USB flash, which it can be saved in safe lock at the responsibility of the etherized person to keep the responsibility cycle small.

The main step to satisfy the protection of is as in RDSCC Protection algorithm.

RDSCC Protection algorithm

Step(1): Input the RDSCC file (f) with length L, open new files Part-A(f1) and Part-B (f2), let N= L/2.
Step(2): FOR i = 1 : N
Read(f, byte (i*2-1), byte (i*2));
Save (f1, byte (i*2-1));
Save (f2, byte (i*2));
End {for i}
Step(3): Output Part-A and Part-B;
Step(4): Keep Part-A in computer; Save Part-B in external unit.
End.

In the same manner, the algorithm of rising protection is as follows:

RDSCC Rising Protection algorithm

Step(1): input files Part-A(f1) and Part-B (f2) with length N, open the RDSCC file (f),
Step(2): FOR i = 1 : N
Read(f1, byte1 (i));
Read(f2, byte 2(i));
Save (f, byte 1(i), byte 2(i));
End {for i}
Step(3): Output RDSCC;
Step(4): Keep RDSCC file in computer.
End.

5. Conclusion

1. The proposed cryptosystem can be used to encrypt not only texts or images, it may be used to encrypt Audio, videos, or any media files, because of its high security and speed.
2. In the protection process, we guarantee that the RDSCC is secure against any attack and its can be portable to work in any computer.
3. The protection process can be extended to include all related files for the RDSCC, like Keys files, LFSR Bank, …., etc.
4. To improve the security of the protection Process, we can propose a secure cryptosystem to encrypt the important files of RDSCC before applying the dividing part.

References

1. Abdullah Ayad Ghazi and Faez Hassan Ali, **2018. "**Robust and Efficient Dynamic Stream Cipher Cryptosystem", *Iraqi Journal of Science*, **59**(2C): 1105-1114
2. Ekdhal, P. **2003.** "On LFSR based Stream Ciphers Analysis and Design", Ph.D. Thesis, November 21.
3. Matt, J. B. **1995.** "*Stream Ciphers Technical Report TR-701*", RSA Laboratories.
4. Menezes, A. P. van Oorschot, P. and Vanstone, S. **1996.** "*Handbook of Applied Cryptography*", CRC Press, 1996.
5. Golomb, S.W. **1982.** "*Shift Register Sequences" San Francisco: Holden Day 1967*,(Reprinted by Aegean Park Press in 1982).
6. Rivest, R. L. **1997.** "*Hand Book of Applied Cryptography*", John Wiley & Sons, 1997.
7. Mohammed Mobark Salih, Mohammad G. S. Al-Safi and Faez Hassan Ali, **2014**. "Dynamic Stream Ciphering Algorithm", *IOSR Journal of Computer Engineering* (IOSR-JCE), 16(2): 72-78.
8. Al-Zewary, R. M. **2017. "**Design of High Security Stream Cipher Generator using Non-linear Combining Function", M. Sc. Thesis, University of Baghdad, Iraq.