Um-Salama Science Journal **Vol.5(2)2008**

Hybrid Cipher System using Neural Network

Asst. Lec. Mokhtar Mohammed Hasan,Asst. Lec. Noor Adnan Ibraheem**

Date of acceptance 12/3/2008

Abstract

The objective of this work is to design and implement a cryptography system that enables the sender to send message through any channel (even if this channel is insecure) and the receiver to decrypt the received message without allowing any intruder to break the system and extracting the secret information.

In this work, we implement an interaction between the feedforward neural network and the stream cipher, so the secret message will be encrypted by unsupervised neural network method in addition to the first encryption process which is performed by the stream cipher method.

The security of any cipher system depends on the security of the related keys (that are used by the encryption and the decryption processes) and their corresponding lengths.

In our work, there are two types of keys; the first type is the keystream that is adopted by the stream cipher stage with optimal length (length of the keystream greater or equal the message length); and the second key type is the final weights that are obtained from the learning process within the neural network stage, So we can represent our work as an update or development for using the neural network to enhance the security of stream cipher.

As a result for a powerful hybrid design, the resulted cipher system provides a high degree of security which satisfies the data confidentially which is the main goal of the most cryptography systems.

1. Introduction

Cryptography is used to protect information to which illegal access is possible. Thus it can be applied to protect communication channels and physical databases [6].Usually to describe any powerful cipher system it must presented from these two points of view:

- The sender point of view (this is called encryption process)
- The receiver point of view (this is called the decryption process)

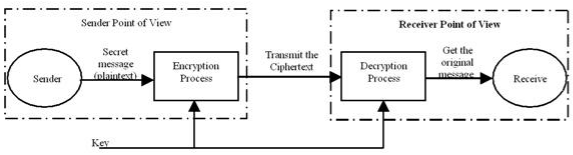


Figure 1: Framework of a cipher system

One of the Cryptographic techniques is symmetric-key technique, which means that the same key is used for both encryption and decryption [8]. The studying of symmetric key ciphers relates mainly to the study of stream ciphers and to their applications [9].

A stream cipher is a type of symmetric encryption algorithm. Stream ciphers designed to be exceptionally fast, much faster than any block cipher (type of symmetric encryption algorithm, which operates on large blocks of data), stream ciphers typically operate on smaller units of plaintext, usually bits. With a stream cipher, the transformation of these smaller plaintext units will vary, depending on when they are encountered during the encryption process.

Stream cipher generates what is called a keystream (a sequence of bits used as a key).

Encryption is accomplished by combining the keystream with the plaintext, usually by the bitwise Exclusive-OR (XOR) operation.

In our work the generation of the keystream is independent of the plaintext and ciphertext, yielding what is termed a synchronous stream cipher.

The security of such synchronous stream cipher depends on the randomness of

*University of Baghdad/ College of Science for Women/ Department of Computer Science

the keystream that is the subsequent keys- tream digits must no better predictable than by just randomly guessing them [7].
the most popular building block of stream ciphers is Linear Feedback Shift Register (LFSR) which is a mechanism for generating a sequence of binary bits. We used the LFSR as a basic component in our work to implement stream cipher.

Our method is hard to break because we used the one time pad key using the LFSR that generates the keys, the key length is 16-bits, so, this produces 2^{16} different combinations, which is 65536 key permutations, if we use each key to encipher two characters that is the key length=16-bits, each character is 8-bits, so we can use our LFSR to encipher 65536*2 characters, which are 131072 characters, if we assume each word is 7 characters as a mean, then we can write a message of 18725 words, this number is far enough to use to write a message and send it through the network, as we said, this is not every think, the output of this method is used to feed a neural network to produce a very difficult to break cipher text.

The next section presents an overview of hybrid cipher system that uses the concepts of neural network for adopting the stream cipher. Then, in the next section the Encryption stage is described. After that Decryption stage is illustrated. Then the results from an experiment are presented. Finally conclusions are expressed.

2. Hybrid Cipher System using Neural Network: An Overview

The suggested design explains that, at the sending end the encryption process will be performed in two consecutive modules.

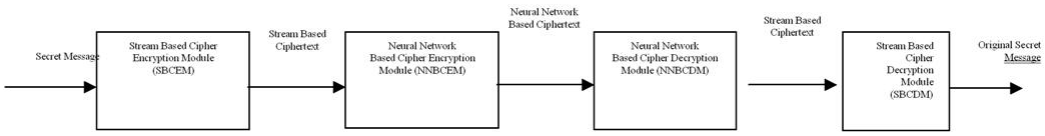


Figure 3: Block diagram of the suggested design

The first module concerned with encrypting the plaintext using stream cipher method to obtain the ciphertext (we called the ciphertext in this stage as stream based ciphertext).We named this module as Stream Based Cipher Encryption_Module (SBCEM).

The second module concerned with encrypting the ciphertext (stream based ciphertext) resulted from the stream based cipher encryption module (the first module) using feedforward neural network to generate the final ciphertext (we called the ciphertext at this stage as neural network based ciphertext) which is ready for transmitting by the sender via secure or maybe insecure channel. We named this module as Neural Network Based Cipher Encryption Module (NNBCEM).

At the receiving end, the decryption process will be done in two consecutive modules as in the encryption process but in the reverse order.

The first decryption module is called Neural Network Based Cipher Decryption Module (NNBCDM) that concerned with decrypting the received ciphertext (neural network based ciphertext) using feedforward neural network to obtain the stream based ciphertext.

The second decrypting module is called Stream Based Cipher Decryption Module (SBCDM) deals with decrypting the stream based ciphertext (resulted from NNBCDM) to obtain the original secret message as a equivalent to the sent message by the sender.

Figure 3 shows the block diagram of our suggested design which comprises the encryption and decryption modules with their corresponding inputs and outputs

3. Encryption Stage

As we mentioned in the last section that the encryption process will be performed in two consecutive modules. Here we will explain each module separately.

3.1 Stream Based Cipher Encryption Module (SBCEM).

SBCEM is the encryption module that converts the secret message from the plaintext to the ciphertext using the stream cipher technique. The ciphertext which is obtained from this module is called stream based ciphertext that will be forwarded to the second encryption module (NNBCEM) to complete the encryption process.
The following figure represents the block diagram of the Stream Based Cipher Encryption Module (SBCEM).

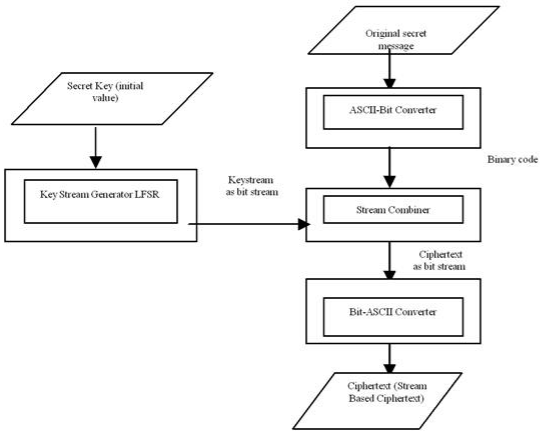


Figure 4: Block diagram of SBCEM

The main functions of this module which must be performed in order steps as follow:

1. Converting the plaintext from the ASCII code to the binary code.
2. Generating a keystream using Linear Feedback Shift Register (LFSR) as a keystream generator. In the next subsection we will explain its mechanism.
3. Combining bit-by-bit the binary code (output of step one) with the bit sequence of the keystream (obtained from step two) to obtain the ciphertext as bit representation,by using the EXCLUSIVE-OR (XOR) as a simple function, therefore it will perform "XORing" bit-by-bit between the binary codes of the secret message and the bit sequence of the keystream to obtain new bit streams
4. Converting the binary code of the ciphertext (obtained from step three) to the ASCII code.

3.1.1 Linear Feedback Shift Register (LFSR)

LFSR is a mechanism for generating a sequence of binary bits. It used in many of the keystream generators where, LFSRs are well-suited to hardware and software implementation, they can produce sequences of large period, and they can produce sequences with good statistical properties. LFSR has two parts:

- A shift register: it is a sequence of bits (if it is *n*-bit long then it is called *n*-bit shift register). Whenever a bit is needed, all of bits in the shift register are shifted 1 bit to the right.
- A feedback function: the new left-most bit is computed as a function (XOR) of all other bits in a register.

The bit positions selected for using in the feedback function are called *taps,* and the list of the bits positions that affect the next state in LFSR is called the *tap stages*.

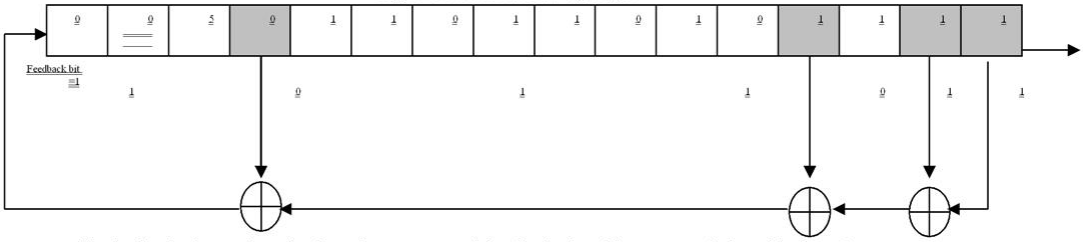
The register consists of a series of cells that are set by an initialization vector. At each instant the contents of the cells of the register are shifted right by one position, and the XOR ( ) of a subset of the cell contents is placed in the leftmost cell. One bit of output is usually derived during this update procedure [6].

The period of a shift register is the length of the output sequence before it starts repeating its initial value, thus for non-singular LFSR of length L bits, the period is at most 2L -1 [6], [7].

Our Suggested LFSR has register length equal 16 bits which guarantee that the period of the stream sequence equal 2^{16} - 1=65535 which means long keystream that will be used in conjunction with binary code of the plaintext.

An example on our suggested design of LFSR is showed in figure 5 with tap number equal four in sequences "0, 1, 3, and 12", the shadowed square means a tap stage.

Figure 5: LFSR with four tap stages



The simplest implementation of such synchronous stream cipher that is adopted in our system is showed in figure 6.

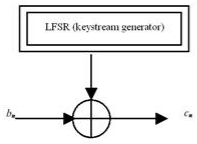


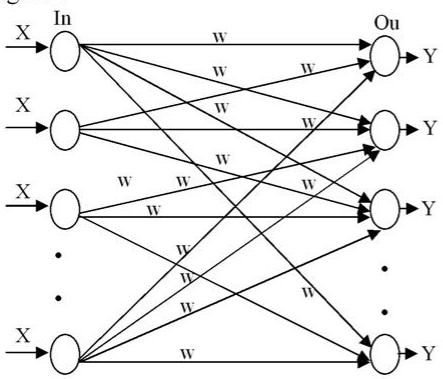
Figure 6: Synchronous Stream Cipher So the keystream (k_n) is "XORed" with a stream of plaintext (b_n) to produce the stream of ciphertext (c_n).

3.2 Neural Network Based Cipher Encryption Module (NNBCEM)

This module is the kernel of our research which represents a new technique that adopted by our design.

The purpose of NNBCEM is to improve the security of the first encryption module (SBCEM) and then improve the security of the overall system. Instead of sending ciphertext (stream based ciphertext) encrypted by SBCEM, the ciphertext will encrypt again by NNBCEM to obtain neural network based ciphertext before sending it to the receiver.

Our suggested design adopts the hebbian neural network which will be learned by an unsupervised approach as showed in figure 7



In our Neural Network The vector $X_1, X_2, \ldots, X_m$ is the input file which represents the stream based ciphertext (output of SBCEM), while $Y_1, Y_2, , Y_m$ is the vector of the output file that represent the final ciphertext (neural network based ciphertext) that it is ready for transmitting to the receiver, and the weight represents the connection strength between the (i^{th}) input node and the (j^{th}) output node.

From figure 7 it is clearly that number of the nodes in the input and output layer equal the number of the characters in the input file (stream based ciphertext), therefore we need (m^2) initial weights that must be prepared before the execution of the NNBCEM.

The suggested hebbian network has the following properties

- it is a single layer (does not contain any hidden layer), this topology provides an ability of simple implementation. And fast learning speed if it is compared with another networks that contain hidden layers.
- It is feedforward network which means that the propagation of the signals in the network will be in one direction only, therefore each neuron will depend on the directed input signals only.
- There is no activation function as in the conventional hebbian neural network. So our design overcomes the problem of selecting which is one of the activation functions that gives the better results in the learning process.

The block diagram of NNBCEM is viewed in figure 8.

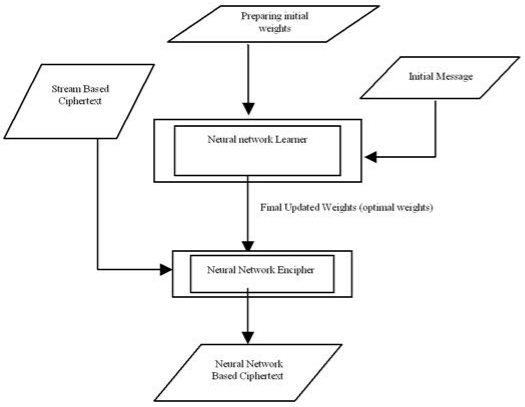


Figure 7: Block diagram of NNBCEM

From figure 8, it is clearly that there are two functions within NNBCEM which are:

1. Neural Network Learner
2. Neural Network Encipher

Which will be explained In the next two subsections.

3.2.1 Neural Network Learner

NNBCEM performs the learning process on the weights through number of steps to find the final updated weights that will be used by the neural network encipher

Neural network learner at first requests the sender to input any message or any file contains the desired message. So this message represents as input for the network, and message must have the same length of the stream based ciphertext.

During the learning process, the neural network learner will prepare two-dimensional array of size ($m*m$), (while m represents the initial message size),then it will perform normalization operation for the initial weights, in order to avoid the problem of overflow in the weight values.The normalization process is implemented by applying the following formula:

$$NW_{ij} = \frac{W_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{m} W_{ij}} \qquad (1)$$

This process is applied on all the initial weights to find the modified weights.

Neural network learner sub-module computes the current actual output as initial output that is used to find the updated weights. The output (Y_j) is found by using the following formula:

$$Y_j = \sum_{i=1}^{m} NW_{ij} * X_i \, such\, that, \, j = 1,2 \ldots m \qquad (2)$$

Where NW_{ij} is the normalized weights and X_i represents the i^{th} element in the initial message file.

The weight updating process is performed by the following formula:

$$\Delta W_{ij} = \zeta * Y_j * \left[X_j - \sum_{k=1}^{J} NW_{ik} * Y_k \right] such\, that\, i, \, j = 1,2 \ldots m \qquad (3)$$

Where (ζ) is the learning factor (positive small value less than one), and (ΔW_{ij}) is the updated weight.

Formula (2) and (3) at the first step works on the normalized initial weights then it will deals with the last updated weights and continuing in this process until the learning process is completed.

3.2.2 Neural Network Encipher

Neural Network Encipher receives the stream based ciphertext and the optimal weights that are obtained from neural network learner (as showed in figure 7).

The encryption process is done by applying the formula 2 where Y_j represents the j^{th} element that will be obtained and written in the output file of neural network based ciphertext, X_i represents the ASCII of the i^{th} element within the file of stream based ciphertext, while W_{ij} is substituted by the optimal weights that are found within neural network learner.

At this stage, the neural network ciphertext is obtained and can be sent to the front end (receiver) which he/she must has the ability to decrypt the neural network ciphertext to obtain the stream based ciphertext and then decrypt it again to obtain the original secret message.

4. Decryption Stage

4.1 Neural Network Based Cipher Decryption Module (NNBCDM)

This sub-module is designed to be used by the receiver to recover the stream

based ciphertext from the received neural network based ciphertext.

NNBCDM uses the inverse of the optimal weights to carry out the decryption process, which means that the sender and receiver sharing the final updated weights (optimal weights) as secret keys, and then the receiver (by NNBCDM) will find the inverse of the optimal weights matrix using any numerical algorithm.The block diagram of NNBCDM is shown in figure 8.

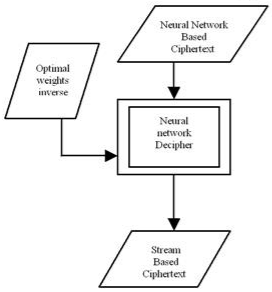


Figure 8: Block diagram of NNBCDM

Neural network decipher implement the decryption process by implementing formula (4).

$$X_j = \sum_{i=1}^{m} (IW)_{ij} * Y_i such that, j = 1,2 \ldots m \quad (4)$$

Where Y_i is the ASCII code of the i^{th} element in the input neural network based ciphertext file, IW_{ij} is the inverse weight that represents the connection strength between the input node i and the output j in the adopted neural network i.e. between the i^{th} code in the neural network based ciphertext file and the j^{th} code in the stream based ciphertext, and X_j represents the j^{th} code of the stream based ciphertext that represent the output of NNBCDM

Stream Based Cipher Decryption Module (SBCDM)

This module uses the principles of the stream cryptographic method in order to obtain the original message.

The following figure represents the block diagram of the Stream Based Cipher Decryption Module (SBCDM).

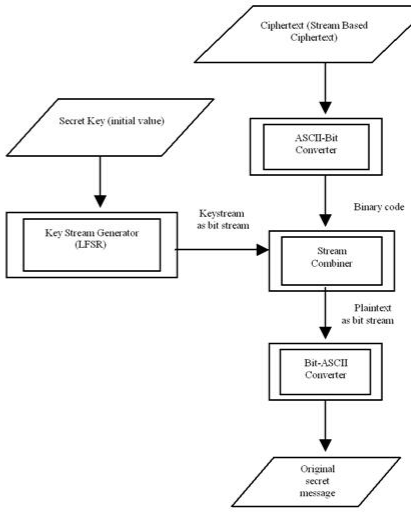


Figure 9: Block diagram of SBCDM

SBCDM performs the following processes that must be carried out in order as follow:

1. Converting the stream based from the character representation to its corresponding binary representation.
2. Generating a keystream using LFSR
3. Combining (bit-by-bit) the binary code (obtained from step one) with the bit stream (obtained from step two) in order to obtain the bit representation of the original message.
4. Converting the bit code of the original message (obtained from step three) to the ASCII code and then to the character code which represents the original secret message.

Finally after we explain the main modules that are implemented in our system separately, we will present 2 block diagrams that show our suggested system design in details.

The first block diagram (figure 10) shows our system from the sending point of view and the second block (figure 11) diagram shows the system from the receiver point of view.

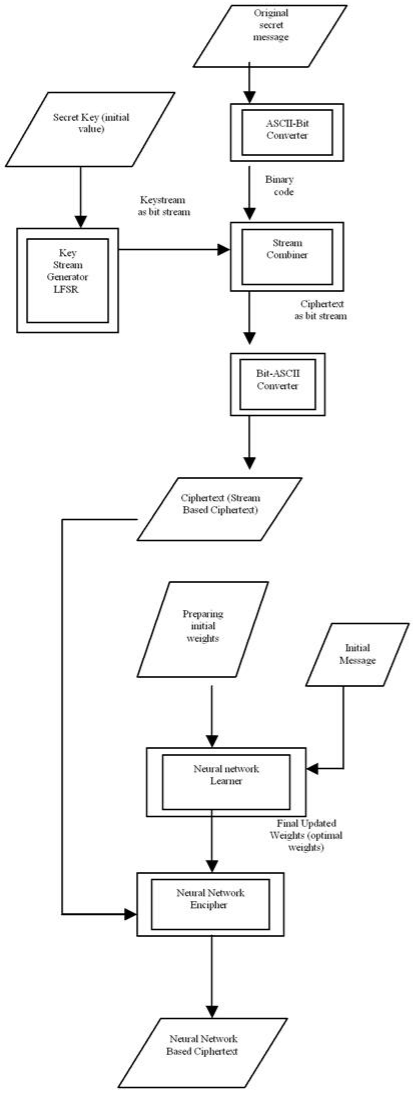


Figure 10: Block diagram of the system sending point of view

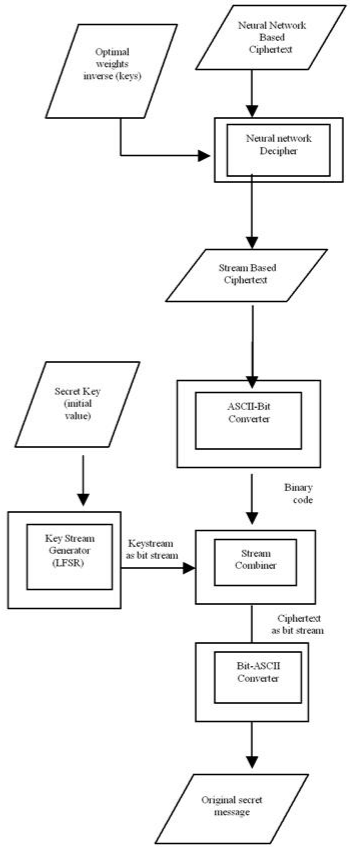


Figure 11: Block diagram of the system receiver point of view

5. Experimental Results

In this section we will test our system stages including their modules by selecting an arbitrary message that represents secret information and we will show the encryption process and the decryption process according to figure 11 and 12 in order to show the practical side of our work.

Now consider the secret message is:

"**This boy is very annoying kid and need to be treated carefully** ".

It is clearly that the length of this message is 62 characters or i.e. 62 byte therefore it is equal 496 bits (62*8), so it needs 496 bit from the LFSR to perform the combining process. Now we will implement SBCEM as in figure 4, this implementation is showed in figure 124.

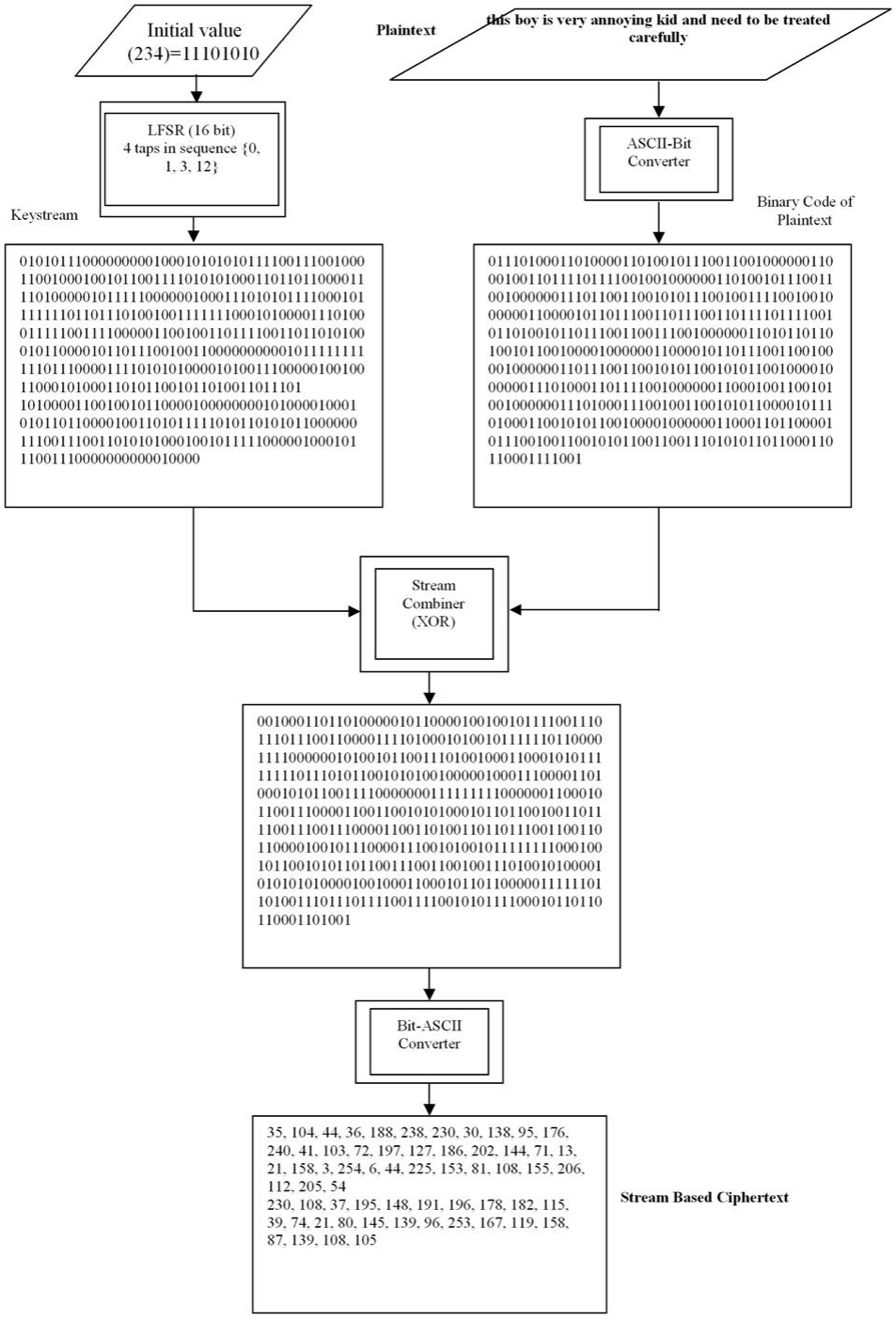


Figure 12: Implementation of SBCEM

It is clearly that the ciphertext (stream based ciphertext) size equals the length of the secret message (62 characters).Note that each word is followed by a space and that is not a weakness because our Neural Network produces different cipher code for each character even if this character is repeated.

The sender will transmit the characters of the resulted ASCII codes (in figure 13) to the NNBCEM, but some of these characters are not printable so we will treat with the ASCII rather than the characters. Implementation of NNBCEM is shown in figure 13.

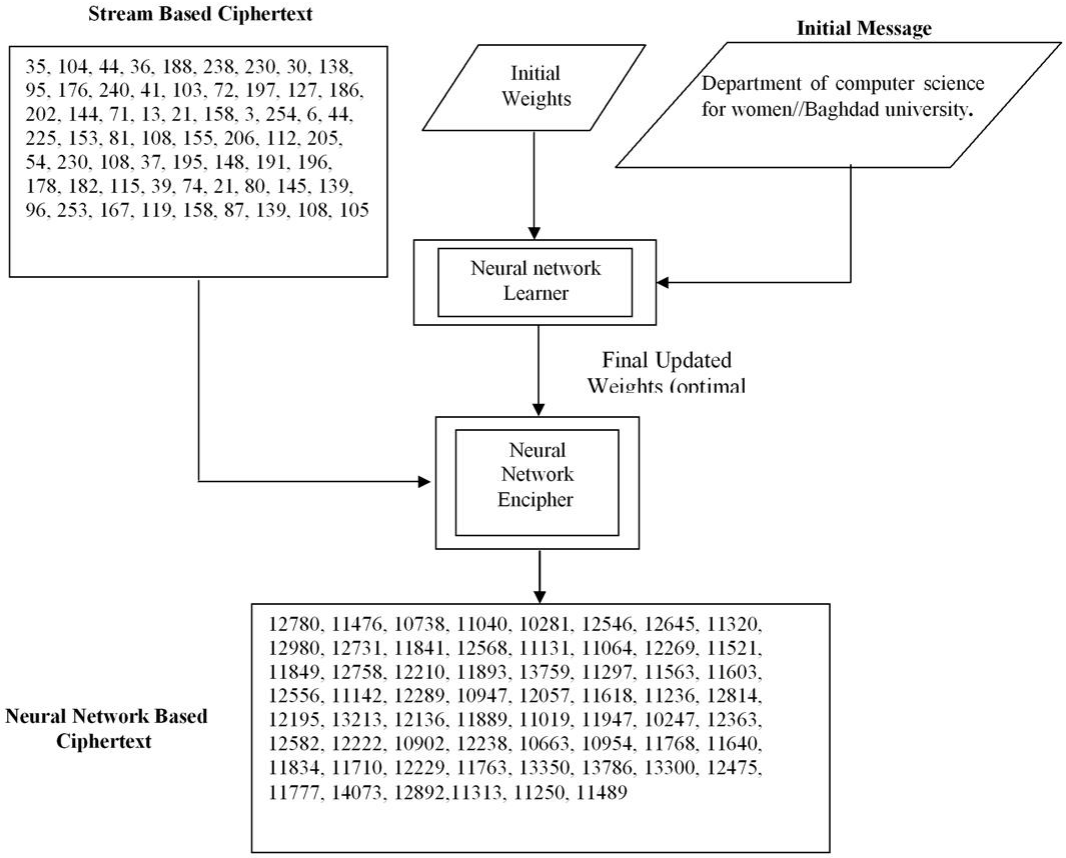


Figure 13: Implementation of NNBCEM

After implementing NNBCEM, the sender will obtain the neural network based ciphertext, which will be sending to the receiver.

After we implementing the NNBCEM in the last figure, the stream based ciphertext (as an integer numbers) will be sending to the receiver. The receiver must at first implement NNBCDM to recover the stream based ciphertext.

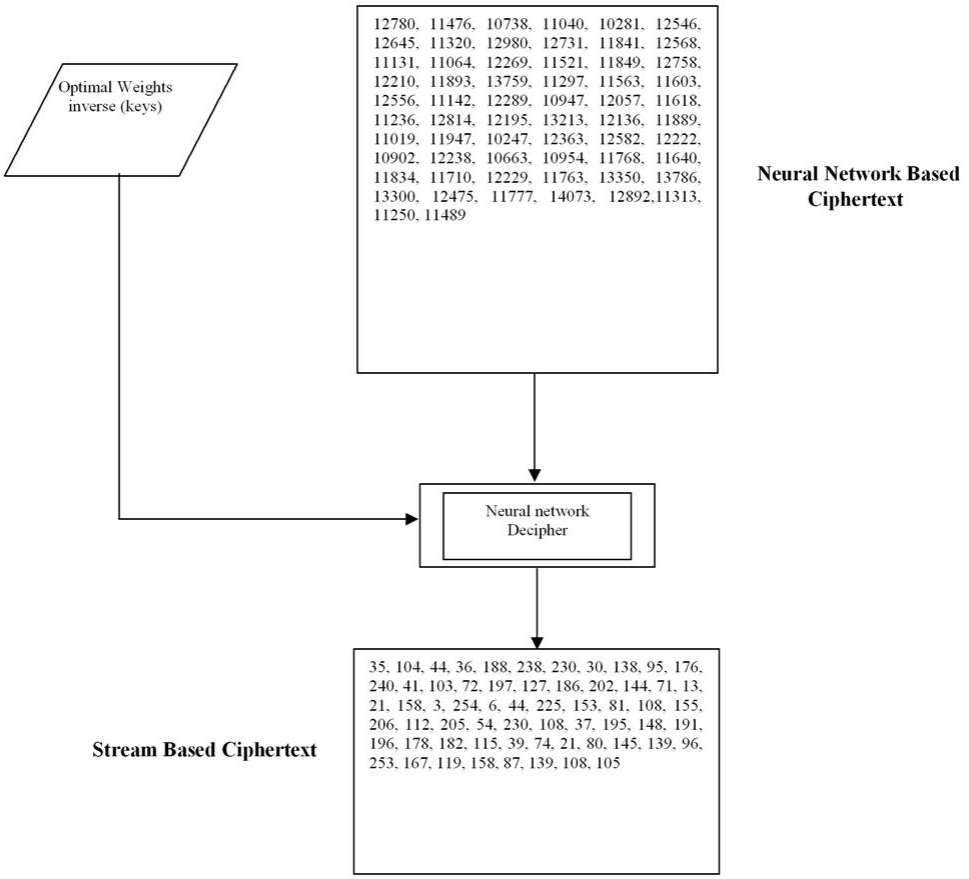


Figure 14: Implementation of NNBCDM

After the receiver implements the NNBCDM and he/she recovers the stream based ciphertext, it is clearly that the NNBCEM and NNBCDM are with good performance, because the stream based ciphertext is recovered within NNBCDM and it is identical to the original stream based ciphertext that is forwarded to the receiver.

The receiver (as a last step) must implement SBCDM in order to extract the secret message as showed in figure 15.

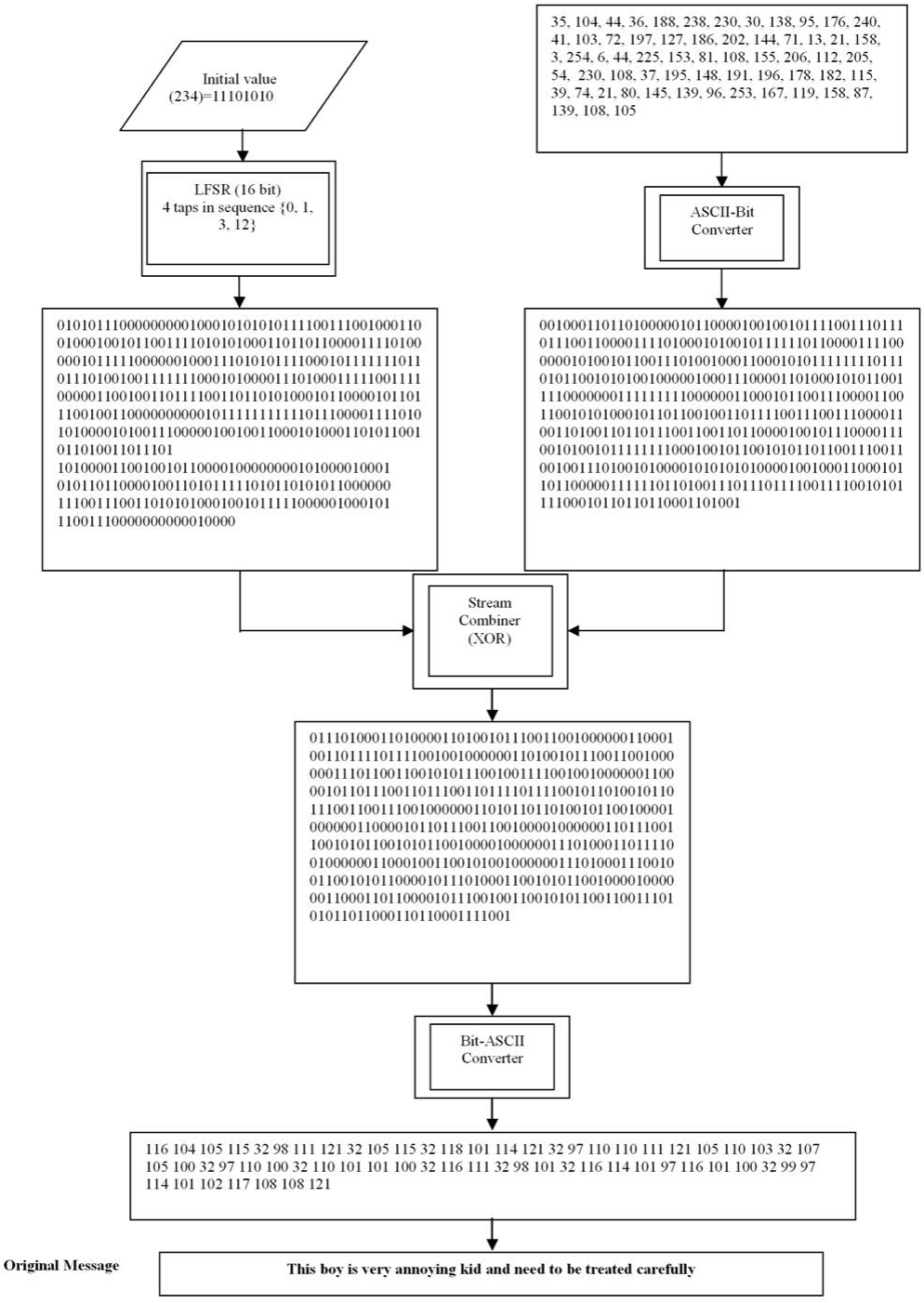


Figure 15: Implementation of SBCDM

6. Conclusions

In this paper, we have presented our hybrid cipher system that use the concepts of neural network which is provides high security. In this work, we implement an interaction between the feedforward neural network and the stream cipher, so the secret message will be encrypted by unsupervised neural network method in addition to the first encryption process which is performed by the stream cipher method.The results demonstrate that the decryption process is performed with good results because the original message has been successfully recovered during the decryption process.

And we achieved our goal because the intruder can not break the transmitted ciphertext because the ciphertext is ciphered using two combined methods not just one method, and each method has its own key length and key values.

7. References

1- Zurada,J.M.,1996,"Introduction to Artificial Neural Systems",Jaico Publishing House.

2- Jain Mao,A.K.,J.1996, "Artificial Neural Networks: A Tutorial", IEEE Computer Society,29(3): 31-44.

3- Kostanic,Ivica,Ham,F.M.,2001,"Principles of Neurocomputing for Science and Engineering", McGraw-Hill,.

4- Lippmann,Richard P.,APRIL 1987," An Introduction to Computing with Neural Nets", IEEE ASSP MAGAZINE.

5- Werner Kinnebrock,1995,"Neural Networks,Fundamentals,Applications,Examples",Technical University Rheinland-Pfalz, 2nd Revised Edition.

6- Denning D.E.,1982, "Cryptography And Data Security",Addison Wesley.

7- HenkC.A.Van Tilborg,1988,"An Introduction to Cryptology", Kluwer Academic Publishers.

8- Beker,H.and Piper,F.1982,"Cipher System:The Protection of Communications", Northwood Publications.

9- Schneier,B.,1997,"Applied Cryptography :Protocols, Algorithms and Source Code", Second Edition, John Wily & Sons Inc.

نظام تشفير هجين باشكال الشبكات العصبية

المدرس المساعد مختار محمد حسن ، المدرس المساعد نور عدنان ابراهيم

الخلاصة

كثيرا ما نشاهد وجود طرق جديدة لكسر النص المشفر المنقول بين طرفين نتيجة لاستعمال طرق معينة من اجل معرفة مفتاح التشفير, لذا فان هدف هذا البحث هو بناء نظام لارسال و استلام البيانات بدون القدرة على كسر النص المنقول من قبل المتطفلين(الدخلاء).حيث ان الطريقة المقترحة هي هجينة بين طريقتي تشفير اعتيادية و تشفير باستعمال الشبكة العصبية.

من المعروف ان امنية اي طريقة تشفير تعتمد على امنية المفتاح المستخدم للتشفير و كذلك طول هذا المفتاح, فعلية قمنا باستعمال مفتاحين الاول يتم توليده لاجل التشفير باستعمال الطريقة التقليدية للتشفير حيث يكون طول المفتاح في هذه الحالة اطول من او مساوي الى طول النص والمفتاح الاخر يمثل الاوزان النهائية الناتجة من عملية تعليم الشبكة العصبية , فبذلك نلاحض ان المفتاح طويل جدا و لا يمكن تفسيره او كسره.

كناتج لهذه العملية المعقدة فاننا قمنا بتوليد نظام تشفير ذات امنية عالية يمنع اي متطفل من معرفة البيانات المنقولة بين طرفين.