CRYPTOGRAPHY SCHEME BASED ON TRANSPARENT FEEDFORWARD NEURAL NETWORK AND ORDERED LOOKUP TABLE

Dr. Ahmed Adel Abdelwahab, Assistant Prof.
Department of Electronics, Communications and Computer Engineering,
Faculty of Engineering, Helwan University, Helwan, Cairo, Egypt.
abdelwahab_99@yahoo.com

(Received September 24, 2005 Accepted December 4, 2005)

A single hidden layer feedforward neural network (FFNN) is called a transparent FFNN if its output vector is a reproduction of the input vector. In this case, the input vector is the target output vector. Once the transparent FFNN is well designed with the m-bit plaintext input set, the designed network is divided into two parts: the transmitter private network encrypting part and the receiver public network decrypting part. The hidden vector which is the output real vector of the private network part is transmitted using the decimal-value ordered lookup table (DVOLT). The m-bit binary cipher vector is the chosen binary index of the lookup table. In this paper, the binary plaintext vector size (m) is chosen to be 16 bits (two bytes). Computer simulation shows that both operations of encryption or decryption of all possible plaintext of 65536 (2^{16}) vectors can be done with zero error and an average processing time of 8.3 msec per 16-bit vector (4.15 msec / byte) per operation. The average hamming distance between the binary plaintext and the binary ciphertext is calculated as 7.8798 for all possible plaintext of 65536 16-bit vectors. This scheme can't be attacked by either brute force or cryptanalysis since there are no binary keys or known mathematical structures. The most important part which must be kept secret is the private matrix of the transmitter private network encrypting part. Each of the private encrypting key and the public decrypting key needs a memory size of about 8.5 Mbytes. The large size of both the private and public keys may limit the applications of the proposed scheme to large workstations intercommunications. This proposed cryptography scheme provides sender authentication and also receiver confidentiality.

KEYWORDS: Cryptography, neural network based cryptography, transparent neural network, and decimal-value ordered lookup table.

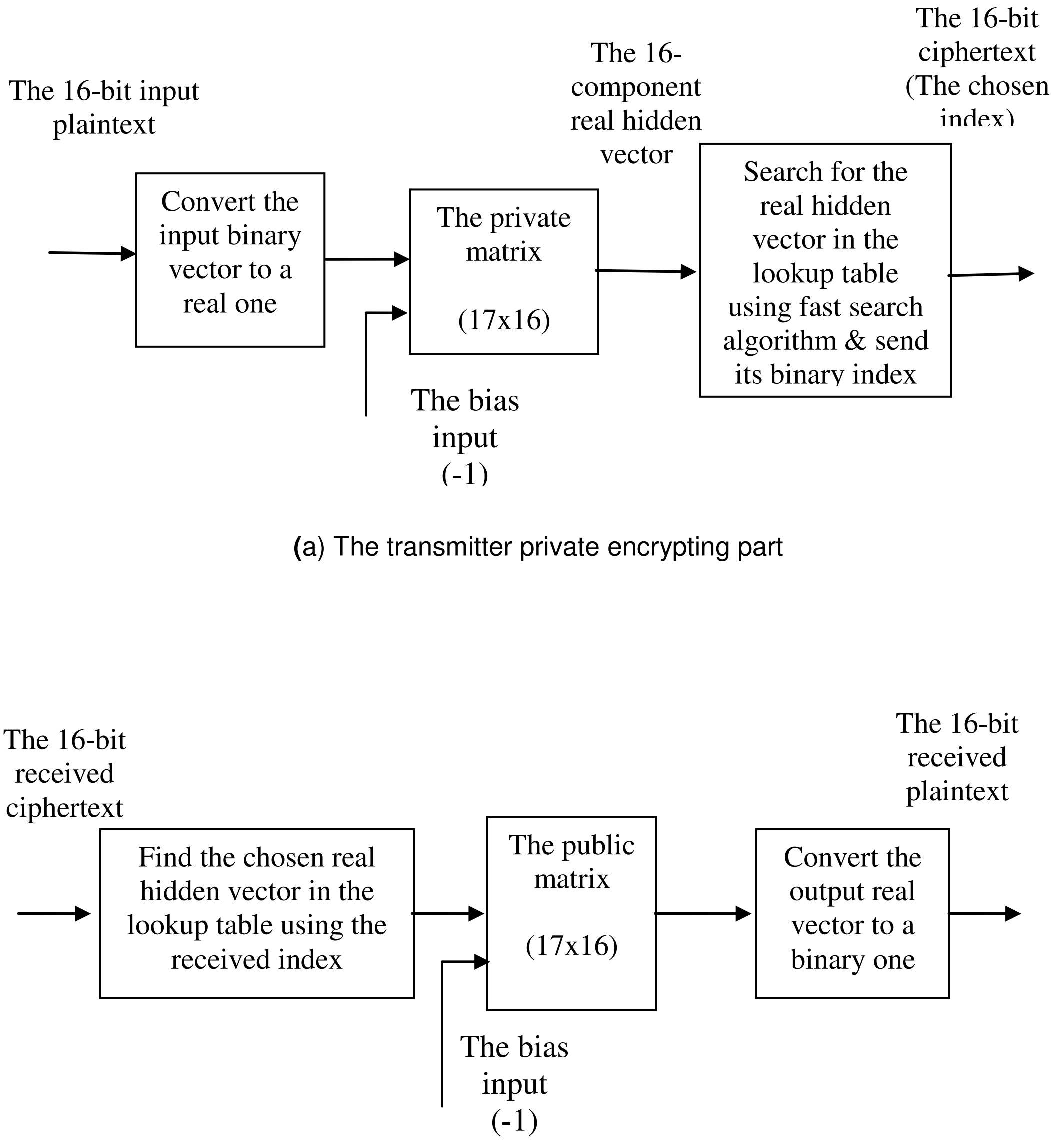
I. INTRODUCTION

The explosive growth in computer system and their interconnections via networks has increased the dependence of both the organizations and individuals on the information stored and communicated using these systems. Data encryption is the most important part in computer network security in order to protect data from disclosure and to guarantee the authenticity of data and messages. Symmetric key and public key are the two most important modern kinds of cryptography. Symmetric key algorithms such as DES, IDEA, AES or blowfish are based on diffusion and confusion of several rounds of main function execution using one key for both encryption and decryption. On the other hand, public key algorithms such as RSA, ElGamal or elliptic curve cryptography are based on mathematical functions [1, 2]. Public key cryptography is asymmetric, involving the use of two separate keys, in contrast to symmetric encryption, which uses only one key. Public key algorithms rely on one key for encryption and a different but related key for decryption. In this paper, a neural network based cryptography scheme is introduced. In this proposed scheme, there is a private key which is used only for encryption while the other different but related public key is only used for decryption.

II. THE PROPOSED CRYPTOGRAPHY SCHEME

Feedforward neural network (FFNN) is fast and accurate if it is designed with Back propagation learning algorithm [3-5]. In this paper, the transparent FFNN is introduced as a single hidden layer FFNN which reproduces its input vector as its output vector. In this case, the input vector is the target output vector and the Back propagation learning algorithm is used during the learning phase. The input vector set is deterministic and consists of all binary m-bit vectors whose decimal values are from 0 to (2^m -1) which represents all possible concatenations of m-bit plaintext. Once the transparent FFNN is well designed with the m-bit plaintext input set, the designed network is divided into two parts: the transmitter private network encrypting part and the receiver public network decrypting part. The hidden vector which is the output real vector of the private network part is transmitted using the decimal-value ordered lookup table (DVOLT). The m-bit binary cipher vector is the binary index of the chosen lookup table vector. In this paper, the binary plaintext vector size (m) is chosen to be 16 bits (two bytes). The transmitter's private network part consists of the input layer with 16 input nodes plus a bias node, the input private matrix and the hidden layer. The size of the input private matrix which is fully connected between the input and the hidden layers is equal to (17 x 16). The hidden layer consists of 16 neurons. The receiver's public network part consists of the output public matrix whose input vector is the chosen lookup table real vector and the output layer whose output vector is the decrypted binary vector (reconstructed plaintext). The size of the public network part input vector is 17 which is the chosen lookup table real vector plus a bias input. The output layer consists of 16 output neurons. Therefore, the size of the public matrix is also equal to (17 x 16). **Figure 1** depicts the proposed scheme.

The size of m determines the plaintext size which should be large enough to acquire the proposed cryptography system good diffusion property so that the plaintext statistical characteristics do not reflect on the ciphertext. On the other hand, the choice of m determines the number of input layer nodes as well as the number of hidden layer neurons and that of the output layer. Therefore, m is chosen to be 16 to compromise between security strength and complexity of the proposed scheme. In order to increase the security strength of the proposed scheme, the input binary message can be divided into 64-bit vectors. A 64-bit input permutation can be defined and used. Four 16-bit input vectors are obtained from each 64-bit permutated input vector which can be encrypted using the proposed scheme.


The 16-bit input plaintext
Convert the input binary vector to a real one
The private matrix (17x16)
The bias input (-1)
The 16-component real hidden vector
Search for the real hidden vector in the lookup table using fast search algorithm & send its binary index
The 16-bit ciphertext (The chosen index)
The 16-bit received ciphertext
Find the chosen real hidden vector in the lookup table using the received index
The public matrix (17x16)
The bias input (-1)
Convert the output real vector to a binary one
The 16-bit received plaintext

(a) The transmitter private encrypting part

(b) The receiver public decrypting part

Figure 1: the proposed cryptography scheme.

The binary input plaintext vector is converted into real vector which maps the logical one binary bit to 0.9 and the logical zero binary bit to 0.1. This real input vector is more suitable for the single hidden layer FFNN. The initial coefficients of both of the private and public matrices are random real numbers between zero and one. The NN is fully connected with sigmoid neuron function. The hidden layer output vectors which need to be transmitted to the receiver are real numbers between -1 to 1. Transmitting these hidden layer output vectors requires much more than m bits per vector. Instead of transmitting these hidden layer output vectors, both the sender (encryptor) and the receiver (decryptor) must have an ascending ordered version of all possible real hidden vectors as a lookup table of size (2^m x m). The transmitter now needs to transmit the index - of only m-bit size – of the corresponding correct real hidden vector so that the size of both plaintext vectors and cipher vectors is the same. The lookup table size will increase as m increases. The author has introduced the decimal-value ordered lookup table and a very fast decimal-value ordered lookup table search algorithm for any size m.

A. *Decimal-Value Ordered lookup table Algorithm*

In this algorithm, we assume the vector set is exhaustive and deterministic i.e. the set consists of all expecting possible vectors. The vector set size equals to (2^m x m) where m is the vector size and the lookup table will be a permuted version of the vector set. Therefore, the lookup table size is also equal to (2^m x m).

The decimal-value ordered lookup table can be formed by following the following steps:

Step 1: Each vector v_i , i = 1,2,3,...,2^m in the vector set is binarized against a carefully selected threshold t, where v(i,j) sets to one if v(i,j) > t otherwise, it resets to zero and j = 1,2,...m.

Step 2: Find the vector of the decimal values in which each component represents the integer decimal value of the m-bit binary word corresponding to a binarized vector in the vector set. Therefore, the size of the decimal-value vector is equal to 2^m. These integer decimal values do not have to have all values from 0 to (2^m -1) but there will be many repeated decimal values. The number of repeated certain decimal value should be small since their corresponding vectors lie in the same hypercube of dimension m.

Step 3: This integer decimal-value vector is sorted (ordered) in ascending order where the sorted values are kept in vector n called the decimal-value ordered vector and their corresponding indexes (positions) in the real input set are kept in vector p. The sorting mechanism is done in order to have all repeated decimal values grouped and ordered. Each group has a unique decimal-value and a known size which can be easily reached and searched for the input vector.

Step 4: The real final decimal-value ordered lookup table is formed by reordering the real vector set according to the p vector so that each vector is placed in a position of its decimal-value group.

B. *At the transmitter*

The transmitter should have a copy of the real decimal-value ordered lookup table of size (2^m x m) as well as a copy of the integer decimal-value ordered vector n of size (2^m).

For each real hidden vector -the output of the private network part- to be transmitted using the above lookup table algorithm, a fast decimal-value ordered lookup table search algorithm is introduced and described in the following steps:

Step 1: Binarize the real hidden vector using the same threshold t.

Step 2: Calculate the integer decimal value which corresponds to the binary m-bit word of the binarized vector.

Step 3: Find the index (position) of the decimal value in the decimal-value ordered vector n to find the corresponding group and its size.

Step 4: Search for the real hidden vector in the corresponding group for ***zero error*** using a distortion measure such as mean square error (mse) and find its decimal index (position) in the lookup table.

Step 5: Convert the decimal index of the chosen hidden vector to m-bit binary word which is the transmitting ciphertext.

The actual search time of the encoding process depends on the size of the decimal-value group which is much smaller than that of the whole lookup table.

C. *At the receiver*

The receiver should have only a copy of the real hidden vector lookup table of size (2^m x m).

Step 1: Calculate the decimal value of the binary m-bit word of the received binary vector (ciphertext) which is the decimal index of the correct hidden real vector in the lookup table.

Step 2: Apply the chosen lookup table real vector to the public network part whose output is a real output vector.

Step 3: Binarize the real output vector using the same threshold t in m-bit word which is the decrypted plaintext.

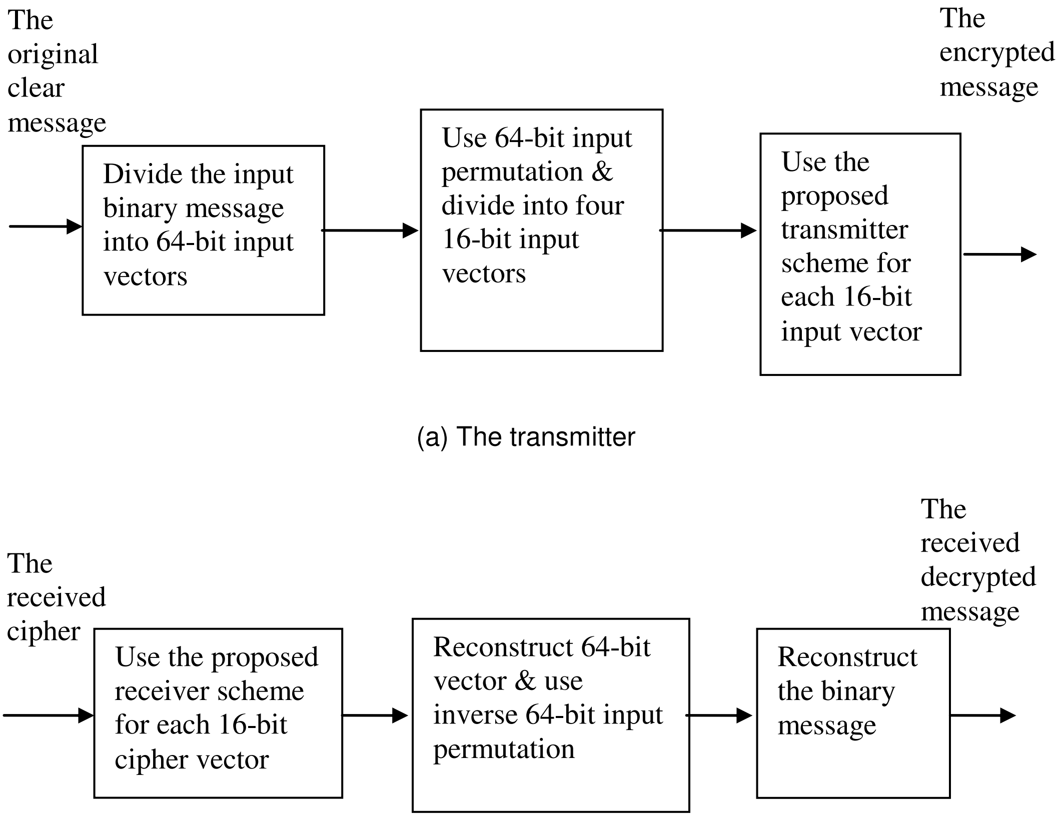
III. COMPUTER SIMULATION RESULTS

In order to evaluate the proposed neural network based cryptography scheme, the binary plaintext vector size is chosen to be 16 bits. A single hidden layer transparent feedforward neural network is designed using the back propagation algorithm. The input layer has 16 input nodes plus a bias one, the hidden layer has 16 neurons plus a bias input and the output layer has 16 neurons. Both bias inputs are constants and need not to be transmitted. Therefore, both of the private network part and the public network part matrices have the same size of (17 x 16). The input vector set consists of all possible concatenations of 16 binary bits, thus the size of the input vector set is (2^{16} x 16). The input vector set is converted into a real one by setting 0 and 1 into 0.1 and 0.9 respectively. For a transparent FFNN, the target vector is the same as the input vector. Starting with random private and public matrices, an excellent transparent FFNN is obtained with zero error. At each iteration, during the learning phase, a different permutation from the input vector set is used. Once, the transparent single hidden layer FFNN is designed, it is split into the private and public parts and the output real hidden vectors of the hidden layer are saved in order to design the decimal-value ordered lookup table which has the same size of (2^{16} x 16). The ascending order decimal-value integer vector n of size (2^{16}) is also designed for the transmitter encrypting process using the fast decimal-value ordered lookup table search algorithm. The cipher binary vector is the binary index of the correct hidden vector in the final ordered lookup table.

An elapsed time = 18.1983 minutes is calculated for both operations of encryption and decryption of all possible plaintext of 65536 16-bit vectors with zero error i.e. an average processing time of 8.3 msec per 16-bit vector (4.15 msec/byte) per operation on Pentium III 866 MHz using Matlab 6. The average hamming distance between the binary plaintext and the binary ciphertext is calculated as 7.8798 for all possible plaintext of 65536 16-bit vectors. For large message, the input binary message can be divided into 64-bit vectors. A 64-bit input permutation can be defined and used. Four 16-bit input vectors are obtained from each 64-bit permutated input vector which can be encrypted using the proposed scheme. **Figure 2** shows the proposed cryptography scheme for any size input binary message.

A. *Key Distribution for the Proposed Cryptography Scheme*

This proposed cryptography system provides sender authentication since no one but the sender has the private matrix and it provides also receiver confidentiality since no one but the receiver has the public matrix. Moreover, a secret permutation can enhance sender authentication and receiver confidentiality. All of the private matrix, the public matrix and the decimal-value ordered lookup table consist of real numbers which require eight bytes each while the decimal-value ordered vector consists of integer numbers which require only two bytes (16 bits) each. The private key consists of (17x16) private matrix of real numbers of the input layer (2176 bytes), a copy of the (65536 x 16) decimal-value ordered lookup table (8388608 bytes) and the 65536 decimal-value ordered vector (131072 bytes) which sum to 8521856 bytes (8.522 Mbytes). The public key consists of (17x16) public matrix of real numbers of the output layer (2176 bytes) and a copy of the (65536 x 16) decimal-value ordered lookup table (8388608 bytes) which sum to 8390784 bytes (8.391 Mbytes). The large size of both the private and public keys may limit the applications of the proposed system to large workstations intercommunications. The private key can be locally generated by the user and needs no transmission. However, the public key must be transmitted via a secure channel.


The original clear message
Divide the input binary message into 64-bit input vectors
Use 64-bit input permutation & divide into four 16-bit input vectors
Use the proposed transmitter scheme for each 16-bit input vector
The encrypted message
The received cipher
Use the proposed receiver scheme for each 16-bit cipher vector
Reconstruct 64-bit vector & use inverse 64-bit input permutation
Reconstruct the binary message
The received decrypted message

(a) The transmitter

(b) The receiver

Figure 2: The proposed message cryptography scheme.

B. *Possible Opponent Attacks*

This proposed cryptography scheme has the following property that is it can't be attacked by either brute force or cryptanalysis since there are no binary keys or known mathematical structures.

Although a transparent single hidden layer FFNN can be easily designed, every pair of initial random private and public matrices will produce a completely different final pair. Therefore, an opponent can design another transparent single hidden layer FFNN and get the corresponding decimal-value ordered lookup table. If the opponent can intercept the cipher vector and use his designed lookup table, the decrypted plaintext will be completely different from the correct one which means one can't use a private matrix and a public matrix of two different transparent neural networks. Moreover, fixing the private (public) matrix, it is computationally infeasible to calculate the same corresponding public (private) matrix using the back propagation algorithm. Computer simulation shows 100 % error in all input set vectors if two different lookup tables of two different transparent neural networks are used in encryption and decryption, respectively.

Plaintext attack would require 2^m plaintext and ciphertext pairs which is assumed to be infeasible. Plaintext attack may be useless if a secret m-bit permutation is regularly changed and used. For example, a 16-bit input permutation such that [3 6 9 2 4 8 12 15 16 1 5 7 14 11 13 10] is used for each input vector and its inverse [10 4 1 5 11 2 12 6 3 16 14 7 15 13 8 9] is applied to each decrypted vector.

This proposed cryptography scheme consists of three important parts. Namely they are the private matrix, the public matrix and the decimal-value ordered lookup table. Although, knowing of only the decimal-value ordered lookup table or only the public matrix is completely useless. However, knowing of the private matrix only, the public matrix can't be calculated but the decimal-value ordered lookup table can be calculated. From the intercepted cipher vector, the opponent can get the corresponding real output vector of the hidden layer from the calculated lookup table and the corresponding input plaintext may be deduced. Therefore, the most important part which must be kept secret is the private matrix. Moreover, the transmitter and the receiver must have complete trust in each other. The receiver can forge a message and calculate its cipher since his public key includes both the public matrix and a copy of the decimal-value ordered lookup table.

CONCLUSIONS

A 16-bit plaintext cryptography scheme based on a transparent single hidden layer feedforward neural network is proposed in this paper. The hidden vector that is the output of the hidden layer is transmitted using the decimal-value ordered lookup table. The cipher vector is the binary index of the chosen real vector in the decimal-value ordered lookup table. Computer simulation shows that both operations of encryption and decryption of all possible plaintext of 65536 (2^{16}) vectors can be done with zero error and an average processing time of 8.3 msec per 16-bit vector or 4.15 msec/byte per operation. The average hamming distance between the binary plaintext and the binary ciphertext is calculated as 7.8798 for all possible plaintext of 65536 16-bit vectors. This scheme can't be attacked by either brute force or cryptanalysis since there are no binary keys or known mathematical structures. The most important part which must be kept secret is the private matrix. Each of the private key and public key needs a memory size of about 8.5 Mbytes. The large size of both the private and public keys may limit the applications of the proposed scheme to large workstations intercommunications. This proposed cryptography scheme provides sender authentication and also receiver confidentiality. Moreover, a secret input permutation can enhance this encryption scheme robustness.

REFERENCES

[1] William Stallings, Cryptography and Network Security, Prentice-Hall, 2003.

[2] Salomaa, A. Public-Key Cryptography, New York: Springer-Verlag, 1996.

[3] Jacek M . Zurada, Introduction to Artificial Neural Systems, West Publishing Company, 1992.

[4] John Hertz, Andres Krogh, abd Richard and G. Palmer, An Introduction To Computing With Neural Nets, Addison-Wesley publishing company, 1991.

[5] Jacques de Villiers and Etienne Barnard, “Backpropagation Neural Nets with One and Two Hidden Layers”, IEEE Transactions on Neural Networks, Vol. 4, No. 1, January 1992.

نظام تشفير يعتمد على الشبكات العصبية الشفافة أمامية التغذية و جدول إلتقاط مرتب

د/أحمد عادل عبد الوهاب أحمد
أستاذ مساعد بقسم هندسة الإلكترونيات و الاتصالات و الحاسبات
كلية الهندسة – جامعة حلوان – حلوان – القاهرة

الشبكات العصبية الشفافة أمامية التغذية تتكون من ثلاث طبقات هي طبقة الدخل ، طبقة الوسط الخفية و طبقة الخرج وسميت بالشفافة حيث يتم تعليم الشبكة العصبية ذات الثلاث طبقات باستخدام خوارزم الانتشار للخلف بحيث يكون متجه الخرج هو نفس متجه الدخل. يتكون متجه الدخل من 16 رقم ثنائي و لذلك فإن مجموعة متجه الدخل تتكون من كل المتجهات الثنائية الممكنة وعددهم (2^{16}) 65536 متجهة ثنائي.

بعد أن يتم تعليم الشبكة العصبية الشفافة ذات الثلاث طبقات ، يتم تقسيمها إلى قسمين: القسم الأول يحتوي طبقة الدخل و طبقة الوسط الخفية و مصفوفة المعاملات التي تربط بينهما ويسمى بالمفتاح الخاص للتشفير و يكون عند المرسل فلا يطلع عليه أحد غيره. القسم الثاني يحتوي مصفوفة المعاملات التي تربط بين طبقة الوسط الخفية و طبقة الخرج ويسمى بالمفتاح العام لفك التشفير المناظر للمفتاح الخاص و يكون عند المرسل إليه فلا يجوز استخدامه إلا لفك تشفير الرسائل المستقبلة من المرسل المناظر.

متجه خرج طبقة الوسط الذي أصبح متجه خرج القسم الأول هو متجه أرقام حقيقية ويتم جدولة هذه المتجهات في جدول حجمه (16 x 65536) ويتم ترتيب المتجهات في الجدول بالقيمة العشرية الذي يعبر عنها كل متجه بعد تحويله لمتجه ثنائي باستخدام المقارنة بقيمة ثابتة لتسهيل عملية البحث عن متجهة معين ويسمى هذا الجدول بعد ترتيب المتجهات العشرية بجدول الالتقاط المرتب. يحتفظ كل من المرسل والمستقبل بصورة من جدول الالتقاط المرتب ويقوم المرسل بالبحث عن متجه خرج القسم الأول ويرسل العنوان بمتجه ثنائي ذو 16 رقم ثنائي و يمثل متجه التشفير لمتجه الدخل الثنائي الأصلي. يقوم المستقبل بالتقاط المتجه العشري من الجدول باستخدام العنوان واستخدامه كمتجه دخل للقسم الثاني ليحصل على المتجه الثنائي الأصلي كمتجه خرج.

تم محاكاة هذا النظام المقترح للتشفير على الحاسب الآلي حيث وجد أمكانية تقسيم أي رسالة ثنائية أصلية إلى متجهات كل منها يتكون من 16 رقم ثنائي تم تشفيرهم بنسبة نجاح 100% كما تم مناقشة أنواع الهجوم المتوقعة للنظام وبيان صعوبة اختراقه مادام المرسل يحافظ على المفتاح الخاص به.