SECURED TRANSMISSION THROUGH MULTI LAYER PERCEPTRON IN WIRELESS COMMUNICATION (STMLP)

Arindam Sarkar[1] and J. K. Mandal[2]

[1]Department of Computer Science & Engineering, University of Kalyani, W.B, India
[2]Department of Computer Science & Engineering, University of Kalyani, W.B, India

ABSTRACT

In this paper, a multilayer perceptron guided encryption/decryption (STMLP) in wireless communication has been proposed for exchange of data/information. Multilayer perceptron transmitting systems at both ends generate an identical output bit and the network are trained based on the output which is used to synchronize the network at both ends and thus forms a secret-key at end of synchronizations of the networks. Weights or hidden units of the hidden layer help to form a secret session key. The plain text is encrypted through chaining , cascaded xoring of multilayer perceptron generated session key. If size of the final block of plain text is less than the size of the key then this block is kept unaltered. Receiver will use identical multilayer perceptron generated session key for performing deciphering process for getting the plain text. Parametric tests have been done and results are compared in terms of Chi-Square test, response time in transmission with some existing classical techniques, which shows comparable results for the proposed technique. Variation numbers of input vectors and hidden layers will increase the confusion /diffusion of the schemeand hence increase the security. As a result variable energy based techniques may be achieved which may be applicable devices/interface of the heterogeneous sizes of the network/device.

KEYWORDS

Multilayer Perceptron, Session Key, Wireless Communication.

1. INTRODUCTION

In recent times wide ranges of techniques are developed to protect data and information from eavesdroppers [1, 2, 3, 4, 5, 6, 7, 8, 9]. Algorithms have their virtue and shortcomings. For Example in DES, AES algorithms [1] the cipher block length is nonflexible. In NSKTE [4], NWSKE [5], AGKNE [6], ANNRPMS [7] and ANNRBLC [8] technique uses two neural network one for sender and another for receiver having one hidden layer for producing synchronized weight vector for key generation. Attacker can get an idea about sender and receiver's neural machine as session architecture of neural machine is static. In NNSKECC algorithm [9] any intermediate blocks throughout its cycle taken as the encrypted block and this number of iterations acts as secret key. Here, if n number of iterations are needed for cycle formation and if intermediate block is chosen as an encrypted block after $n/2^{th}$ iteration then exactly same number of iterations i.e. n/2 are needed for decode the block which makes easier the attackers life. This paper proposed a multilayer perceptron guided encryption technique in wireless communication to overcome the problem.

The organization of this paper is as follows. Section 2 of the paper deals with the problem domain and methodology. Proposed Multilayer Perceptron based key generation has been discussed in

DOI : 10.5121/ijmnct.2014.4401

section 3. Triangularization encryption technique is given in section 4. Triangularization decryption has been presented in section 5. Section 6 presents energy computation technique. Complexity analysis of the technique is given in section 7. Experimental results are described in section 8. Analysis of the results presented in section 9. Analysis regarding various aspects of the technique has been presented in section 10. Conclusions and future scope are drawn in section 11 and that of references at end.

2. PROBLEM DOMAIN AND METHODOLOGY

In security based communication the main problem is distribution of key between sender and receiver. As, during exchange of key over public channel intruders can intercept the key as a middleman. The problem has been addressed and a technique has been proposed addressing the issue. These are presented in section 2.1 and 2.2 respectively.

2.1. Man-In-The-Middle Attack

Intruders intercepting in the middle between sender and receiver try to capture all the information transmitting from both. Diffie-Hellman key exchange technique [1] suffers from this type of problems. Intruders can act as sender/ receiver simultaneously and try to steal secret session key at the time of exchanging key via public channel.

2.2. Methodology

Well known problem of man in the middle attack has been addressed in STMLP where secret session key is not exchanged over public insecure channel. At end of synchronization both parties' generates identical weight vectors and activated hidden layer outputs for both the parties become identical. This identical output of hidden layer for both parties are used as one time secret session key for secured data exchange.

The basic idea here is to design such a program with effective GUI which helps people to understand the underlying calculations. In this case this would be the Tree Parity Machine and the various encryption and decryption techniques. First we need to figure out what are main functions of our system. Since we are going to work on various Neural network structures we need a menu to choose from. Again after that, two different Neural network need mutual synchronization and associated statistical data like type of network and total time required to synchronize mutually. Then at the end we need a menu to choose various encryption and decryption techniques and statistical modules to compute probable power consumption by the network. So we need various menus / forms to cater our need of various functions within their scope. So the schematic view looks like the figure 1.

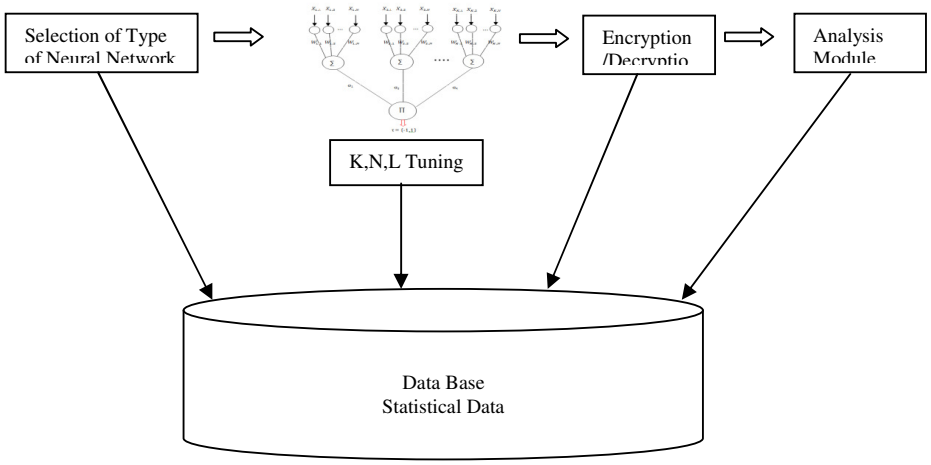

Figure 1. Schematic diagram

3.MULTILAYER PERCEPTRON BASED SESSION KEY GENERATION

A multilayer perceptron synaptic simulated weight based undisclosed key generation is carried out between recipient and sender. Figure 2 shows multilayer perceptron based synaptic simulation system. Same single hidden layer among multiple hidden layers for a particular session. All other hidden layers goes in deactivated mode with the incoming input. The key generation technique with analysis using random number of nodes (neurons) along with the corresponding algorithm is discussed in the subsections 3.1 to 3.5.

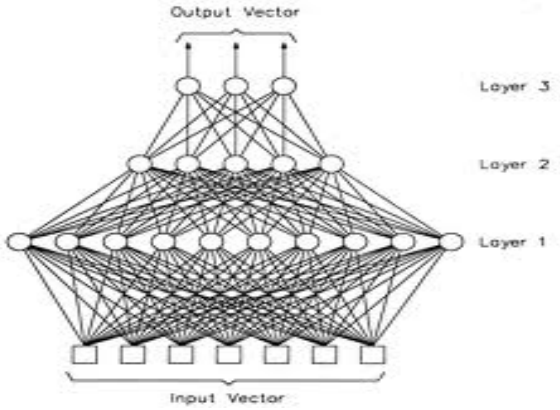

Figure 2. A multilayer perceptron with 3 hidden Layers

Multilayer perceptron in each session acts as a single layer network with dynamically chosen one activated hidden layer and K no. of hidden neurons, N no. of input neurons having binary input vector, $x_{ij} \in \{-1,+1\}$, discrete weights, are generated from input to output, are lies between -L and +L, $w_{ij} \in \{-L,-L+1,\ldots,+L\}$.Where i = 1,…,K denotes the i^{th} hidden unit of the perceptron and j = 1,…,N the elements of the vector and one output neuron. Output of the hidden units is calculated by the

weighted sum over the current input values . So, the state of the each hidden neurons is expressed using (eq.1)

$$h_i = \frac{1}{\sqrt{N}} w_i x_i = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} w_{i,j} x_{i,j} \quad (1)$$

Output of the i[th] hidden unit is defined as

$$\sigma_i = \mathrm{sgn}(h_i) \quad (2)$$

In case of $h_i = 0, \sigma_i = -1$ to produce a binary output. Hence, $\sigma_i = +1$, if the weighted sum over its inputs is positive, or else it is inactive, $\sigma_i = -1$. The output of a perceptron is the product of the hidden units expressed in (eq. 2).

$$\tau = \prod_{i=1}^{K} \sigma_i \quad (3)$$

3.1 Simulation

Input:-Random weights, input vectors for both multilayer perceptrons. Output:-Secret key through synchronization of input and output neurons as vectors. Method:-Random vectors generated, fed into the networks. Vectors are updated only when output of machines produce identical output . The process continue till both machines are fully synchronized.

Step 1. *Initialize of random weight values of synaptic links between input layer and randomly selected activated hidden layer.*

$$\text{Where, } w_{ij} \in \{-L, -L+1, \ldots, +L\} \quad (4)$$

Step 2. *Repeat step 3 to 6 until the full synchronization is achieved, using Hebbian-learning rules.*

$$w_{i,j}^{+} = g\left(w_{i,j} + x_{i,j}\tau\Theta(\sigma_i\tau)\Theta\left(\tau^A\tau^B\right)\right) \quad (5)$$

Step 3. *Generate random input vector X. Inputs are generated by a third party or one of the communicating parties.*

Step 4. *Compute the values of the activated hidden neurons of activated hidden layer using (eq. 6)*

$$h_i = \frac{1}{\sqrt{N}} w_i x_i = \frac{1}{\sqrt{N}} \sum_{j=1}^{N} w_{i,j} x_{i,j} \quad (6)$$

Step 5. *Compute the value of the output neuron using*

$$\tau = \prod_{i=1}^{K} \sigma_i \quad (7)$$

Compare the output values of both multilayer perceptron by exchanging the system outputs.
if Output (A) ≠ Output (B), Go to step 3
else if Output (A) = Output (B) then one of the suitable learning rule is applied only the hidden units are trained which have an output bit identical to the common output.

Update the weights only if the final output values of the perceptron are equivalent. When synchronization is finally achieved, the synaptic weights are identical for both the system.

3.2 Multilayer Perceptron Learning

At the beginning of the synchronization process multilayer perceptron of A and B start with uncorrelated weight vectors $w_i^{A/B}$. For each time step K, public input vectors are generated randomly and the corresponding output bits $\tau^{A/B}$ are calculated. Afterwards A and B communicate their output bits to each other. If they disagree, $\tau^A \neq \tau^B$, the weights are not changed. Otherwise learning rules suitable for synchronization is applied. In the case of the Hebbian learning rule [10] both neural networks learn from each other.

$$w_{i,j}^{+} = g\left(w_{i,j} + x_{i,j}\Theta(\sigma_i\tau)\Theta\left(\tau^A\tau^B\right)\right) \tag{8}$$

The learning rules used for synchronizing multilayer perceptron share a common structure. That is why they can be described by a single (eq. 4)

$$w_{i,j}^{+} = g\left(w_{i,j} + f\left(\sigma_i, \tau^A, \tau^B\right)x_{i,j}\right) \tag{9}$$

with a function $f\left(\sigma_i, \tau^A, \tau^B\right)$, which can take the values -1, 0, or +1. In the case of bidirectional interaction it is given by

$$f\left(\sigma_i, \tau^A, \tau^B\right) = \Theta\left(\sigma\tau^A\right)\Theta\left(\tau^A\tau^B\right)\begin{cases}\sigma & \textit{Hebbian learning}\\ -\sigma & \textit{anti-Hebbian learning}\\ 1 & \textit{Random walk learning}\end{cases} \tag{10}$$

The common part $\Theta\left(\sigma\tau^A\right)\Theta\left(\tau^A\tau^B\right)$ of $f\left(\sigma_i, \tau^A, \tau^B\right)$ controls, when the weight vector of a hidden unit is adjusted. Because it is responsible for the occurrence of attractive and repulsive steps [6].

3.3 Weight Distribution within Multilayer Perceptron

In case of the Hebbian rule (eq. 8), A's and B's multilayer perceptron learn their own output. Therefore the direction in which the weight $w_{i,j}$ moves is determined by the product $\sigma_i x_{i,j}$. As the output σ_i is a function of all input values, $x_{i,j}$ and σ_i are correlated random variables. Thus the probabilities to observe $\sigma_i x_{i,j} = +1$ or $\sigma_i x_{i,j} = -1$ are not equal, but depend on the value of the corresponding weight $w_{i,j}$ [11, 13, 14, 15, 16].

$$P\left(\sigma_i x_{i,j} = 1\right) = \frac{1}{2}\left[1 + erf\left(\frac{w_{i,j}}{\sqrt{NQ_i - w_{i,j}^2}}\right)\right] \tag{11}$$

According to this equation, $\sigma_i x_{i,j} = \mathrm{sgn}(w_{i,j})$ occurs more often than the opposite, $\sigma_i x_{i,j} = -\mathrm{sgn}(w_{i,j})$. Consequently, the Hebbian learning rule (eq. 8) pushes the weights towards the boundaries at -L and +L. In order to quantify this effect the stationary probability

distribution of the weights for $t \to \infty$ is calculated for the transition probabilities. This leads to [11].

$$P(w_{i,j} = w) = P_0 \prod_{m=1}^{|w|} \frac{1 + erf\left(\frac{m-1}{\sqrt{NQ_i - (m-1)^2}}\right)}{1 - erf\left(\frac{m}{\sqrt{NQ_i - m^2}}\right)} \tag{12}$$

Here the normalization constant ρ_0 is given by

$$P_0 = \left(\sum_{w=-L}^{L} \prod_{m=1}^{|w|} \frac{1 + erf\left(\frac{m-1}{\sqrt{NQ_i - (m-1)^2}}\right)}{1 - erf\left(\frac{m}{\sqrt{NQ_i - m^2}}\right)} \right)^{-1} \tag{13}$$

In the limit $N \to \infty$ the argument of the error functions vanishes, so that the weights stay uniformly distributed. In this case the initial length of the weight vectors is not changed by the process of synchronization.

$$\sqrt{Q_i(t=0)} = \sqrt{\frac{L(L+1)}{3}} \tag{14}$$

But if N is finite, the probability distribution itself depends on the order parameter Q_i Therefore its expectation value is given by the solution of the following equation:

$$Q_i = \sum_{w=-L}^{L} w^2 P(w_{i,j} = w) \tag{15}$$

3.4 Order Parameters

In order to describe the correlations between two multilayer perceptron caused by the synchronization process, one can look at the probability distribution of the weight values in each hidden unit. It is given by (2L + 1) variables.

$$P_{a,b}^{i} = P(w_{i,j}^{A} = a \wedge w_{i,j}^{B} = b) \tag{16}$$

which are defined as the probability to find a weight with $w_{i,j}^{A} = a$ in A's multilayer perceptron and $w_{i,j}^{B} = b$ in B's multilayer perceptron. In both cases, simulation and iterative calculation, the standard order parameters, which are also used for the analysis of online learning, can be calculated as functions of $P_{a,b}^{i}$ [12].

$$Q_i^A = \frac{1}{N} w_i^A w_i^A = \sum_{a=-L}^{L} \sum_{b=-L}^{L} a^2 P_{a,b}^{i} \tag{17}$$

$$Q_i^B = \frac{1}{N} w_i^B w_i^B = \sum_{a=-L}^{L} \sum_{b=-L}^{L} b^2 P_{a,b}^{i} \tag{18}$$

$$R_i^{AB} = \frac{1}{N} w_i^A w_i^B = \sum_{a=-L}^{L} \sum_{b=-L}^{L} ab P_{a,b}^{i} \tag{19}$$

The level of synchronization is given by the normalized overlap between two corresponding hidden units as

$$\rho_i^{AB} = \frac{w_i^A w_i^B}{\sqrt{w_i^A w_i^A}\sqrt{w_i^B w_i^B}} = \frac{R_i^{AB}}{\sqrt{Q_i^A Q_i^B}} \quad (20)$$

3.5Secret Session Key

At end of full weight synchronization process, weight vectors between input layer and activated hidden layer of both multilayer perceptron systems become identical. Activated hidden layer's output of source multilayer perceptron is used to construct the secret session key. This session key is not get transmitted over public channel because receiver multilayer perceptron has same identical activated hidden layer's output. Compute the values of the each hidden unit by

$$\sigma_i = \operatorname{sgn}\left(\sum_{j=1}^{N} w_{ij} x_{ij}\right) \quad \operatorname{sgn}(x) = \begin{cases} -1 \; if \; x < 0, \\ 0 \quad if \; x = 0, \\ 1 \quad if \; x > 0. \end{cases} \quad (21)$$

For example consider 8 hidden units of activated hidden layer having absolute value (1, 0, 0, 1, 0, 1, 0, 1) becomes an 8 bit block. This 10010101 become a secret session key for a particular session and cascaded xored with recursive replacement encrypted text. Now final session key based encrypted text is transmitted to the receiver end. Receiver has the identical session key i.e. the output of the hidden units of activated hidden layer of receiver. This session key used to get the recursive replacement encrypted text from the final cipher text. In the next session both the machines started tuning again to produce another session key.

Identical weight vector derived from synaptic link between input and activated hidden layer of both multilayer perceptron can also becomes secret session key for a particular session after full weight synchronization is achieved.

4. ENCRYPTION

For encryption a triangular based technique has been described. During plain text encryption, in the first phase consider a block $S = s^0_0\, s^0_1\, s^0_2\, s^0_3\, s^0_4\, s^0_5 \ldots s^0_{n-2}\, s^0_{n-1}$ of size n bits, where $s^0_i = 0$ or 1 for $0 <= i <= (n-1)$. Now, starting from MSB (s^0_0) and the next-to-MSB (s^0_1), bits are pair-wise XORed, so that the 1st intermediate sub-stream $S^1 = s^1_0\, s^1_1\, s^1_2\, s^1_3\, s^1_4\, s^1_5 \ldots s^1_{n-2}$ is generated consisting of (n-1) bits, where $s^1_j = s^0_j \oplus s^0_{j+1}$ for $0 <= j <= n-2$, $\oplus$ stands for the exclusive OR operation. This 1st intermediate sub-stream S^1 is also then pair-wise XORed to generate $S^2 = s^2_0\, s^2_1\, s^2_2\, s^2_3\, s^2_4\, s^2_5 \ldots s^2_{n-3}$, which is the 2nd intermediate sub-stream of length (n-2). This process continues (n-1) times to ultimately generate $S^{n-1} = s^{n-1}_0$, which is a single bit only. Thus the size of the 1st intermediate sub-stream is one bit less than the source sub-stream; the size of each of the intermediate sub-streams starting from the 2nd one is one bit less than that of the sub- stream wherefrom it was generated; and finally the size of the final sub-stream in the process is one bit less than the final intermediate sub-stream. Table 1 and figure 3 show the process.

Table 1

Options for choosing Target Block from Triangle

Option Serial No.	Target Block	Method of Formation
001	$s^0_0\, s^1_0\, s^2_0\, s^3_0\, s^4_0\, s^5_0 \ldots s^{n-2}_0\, s^{n-1}_0$	Taking all the MSBs starting from the source block till the last block generated
010	$s^{n-1}_0\, s^{n-2}_0\, s^{n-3}_0\, s^{n-4}_0\, s^{n-5}_0 \ldots s^1_0\, s^0_0$	Taking all the MSBs starting from the last block generated till the source block
011	$s^0_{n-1}\, s^1_{n-2}\, s^2_{n-3}\, s^3_{n-4}\, s^4_{n-5} \ldots s^{n-2}_1\, s^{n-1}_0$	Taking all the LSBs starting from the source block till the last block generated
100	$s^{n-1}_0\, s^{n-2}_1\, s^{n-3}_2\, s^{n-4}_3\, s^{n-5}_4 \ldots s^1_{n-2}\, s^0_{n-1}$	Taking all the LSBs starting from the last block generated till the source block

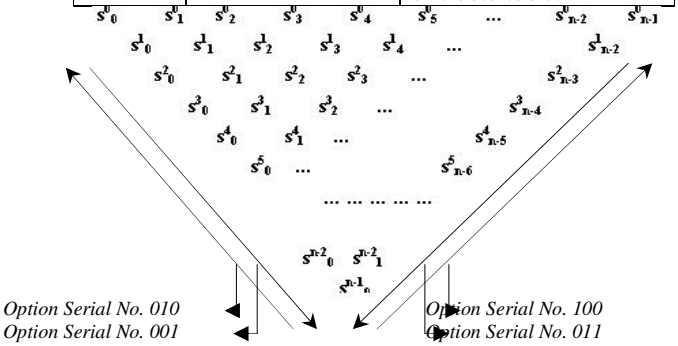

Figure 3. Options diagram for choosing Target Block from Triangle

Table 1describes different options for choosing target block from triangle. This option is generated by modulo 4 division of the value of output neuron then adding 1. Then take the binary version of the decimal no. Each block size is represented by 5 bits and 3 bits are used to denoting the option no. for that block. So, total 8 bits are used to describe a single block length and option chosen. For multiple blocks several 8bits are attached together preceded by first 8 bits ($2^8 = 256$ blocks can be formed in one session) to describe total no. of block to forms intermediate sub key. Maximum length of this sub key will be (256 blocks X 8 bits per block) 2048 bits. This sub key is padded in the front of the encrypted text. Then the multilayer perceptron generated synchronized one time session key is repeatedly xored with the intermediate traingularized cipher text by considering same key & traingularized cipher text length. This mechanism is performed until all the blocks get exhausted.

5. DECRYPTION

During decryption, the receiver's multilayer perceptron generated synchronized one time session key is xored with the cipher text. The technique of performing xoring is same that was in encryption process. Finally from the outcomes intermediate encrypted block (E) and sub key block is extracted and now key is use to decipher the E to get the source stream. To ease the explanation of decryption technique, let us consider, $e^0_{i-1} = s^{i-1}_{n-i}$ for $1 <= i <= n$, so that the encrypted block becomes $E = e^0_0\, e^0_1\, e^0_2\, e^0_3\, e^0_4 \ldots e^0_{n-2}\, e^0_{n-1}$. After the formation of the triangle, for the purpose of decryption, the block $e^{n-1}_0\, e^{n-2}_0\, e^{n-3}_0\, e^{n-4}_0\, e^{n-5}_0 \ldots e^1_0\, e^0_0$, i.e., the block constructed by taking all the MSBs of the blocks starting from the finally generated single-bit block E^{n-1} to E, are to be taken together and it is to be considered as the decrypted block. Figure 4 show the

triangle generated and hence the decrypted block obtained. Here the intermediate blocks are referred to as E^1, E^2, …, E^{n-2} and the final block generated as E^{n-1}.

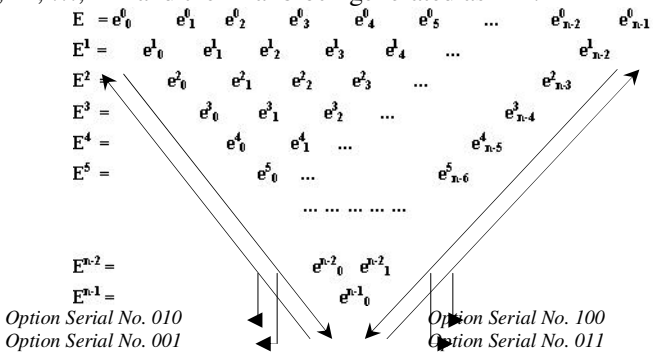

Figure 4. Generation of Source Block from Target

6. ENERGY VARIABILITY

The proposed schemes have a good potential of energy variability which can be incorporated and may be adopted on the fly during transmission. We know that energy required for a blue tooth communication is less than that of a WiFi communication. This variability of energy can be incorporated into the encryption system through incorporating variable number of hidden layers and input neurons. Table 2 shows the proposed network sizes for various types of wireless networks.

Table 2. Network type vs size of neural network

Type of Network	Energy Availability	Network Size and parameters
Wireless Personal Area Networks (Bluetooth, Infrared)	Very Low	No. of input layer neurons= 5 to 10 No. of Hidden layer neurons= 4 to 8 No. of Hidden layer = 1 to 3 Synaptic Depth (L)= +5 to -5
Wireless LAN	Low	No. of input layer neurons= 10 to 25 No. of Hidden layer neurons= 8 to 20 No. of Hidden layer = 3 to 4 Synaptic Depth (L)= +10 to -10
Wireless mesh network	Medium	No. of input layer neurons= 25 to 40 No. of Hidden layer neurons= 20 to 35 No. of Hidden layer = 4 to 5 Synaptic Depth (L)= +15 to -15
Wireless MAN	Moderate	No. of input layer neurons= 40 to 50 No. of Hidden layer neurons= 35 to 45 No. of Hidden layer = 5 to 6 Synaptic Depth (L)= +20 to -20
Wireless WAN	Relatively High	No. of input layer neurons= 50 to 55 No. of Hidden layer neurons= 45 to 50 No. of Hidden layer = 5 to 6 Synaptic Depth (L)= +25 to -25
Cellular network	High	No. of input layer neurons= 55 to 100 No. of Hidden layer neurons= 50 to 70 No. of Hidden layer = 5 to 6 Synaptic Depth (L)= +30 to -30

7. COMPLEXITY ANALYSIS

The complexity of the technique will be O(L), can be computed using following three steps.

Step 1. *To generate a MLP guided key of length N needs O(N) Computational steps. The average synchronization time is almost independent of the size N of the networks, at least up to N=1000.Asymptotically one expects an increase like O (log N).*

Step 2. *Complexity of the encryption technique is O(L).*

Step 2. 1. *Triangularization encryption process takes O(L).*

Step 2. 2. *MLP based encryption technique takes O(L) amount of time.*

Step 3. *Complexity of the decryption technique is O(L).*

Step 3. 1. *In MLP based decryption technique, complexity to convert final cipher text into Tra cipher text T takes O(L).*

Step 3. 2. *Transformation of cipher text T into the corresponding stream of bits S = s_0 s_1 s_2 s_3 s_4…s_{L-1}, which is the source block takes O(L) as this step also takes constant amount of time for merging s_0 s_1 s_2 s_3 s_4…s_{L-1}.*

So, overall time complexity of the entire technique is O(L).

8. RESULTS

In this section the results of implementation of the proposed STMLP technique has been presented in terms of encryption decryption time, Chi-Square test, source file size vs. encryption time along with source file size vs. encrypted file size. The results are also compared with existing RSA [1] technique, existing ANNRBLC[8] and NNSKECC[9].

Table 3. Encryption / decryption time vs. File size

Encryption Time			Decryption Time		
Source Size(bytes)	**STMLP**	**NNSKECC[9]**	**Encrypted Size(bytes)**	**STMLP**	**NNSKECC[9]**
18432	6. 42	7.85	18432	6.99	7.81
23044	9. 23	10.32	23040	9.27	9.92
35425	14. 62	15.21	35425	14. 47	14.93
36242	14. 72	15.34	36242	15. 19	15.24
59398	25. 11	25.49	59398	24. 34	24.95

Table 3 shows encryption and decryption time with respect to the source and encrypted size respectively. It is also observed the alternation of the size on encryption.

In figure 5 stream size is represented along X axis and encryption/decryption time is represented along Y-axis. This graph is not linear, because of different time requirement for finding appropriate MLP key. It is observed that the decryption time is almost linear, because there is no MLP key generation process during decryption.

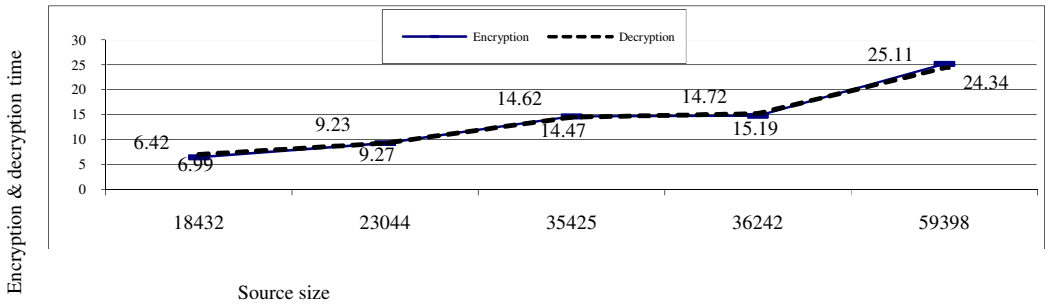

Figure 5 Encryption decryption time against stream size

Table 4 shows Chi-Square value for different source stream size after applying different encryption algorithms. It is seen that the Chi-Square value of STMLP is better compared to the algorithm ANNRBLC [8] and comparable to the Chi-Square value of the RSA algorithm.

Table 4. Source size vs. Chi-Square value

Stream Size (bytes)	Chi-Square value (TDES) [1]	Chi-Square value in (STMLP)	Chi-Square value (ANNRBLC) [8]	Chi-Square value (RSA) [1]
1500	1228.5803	2856.2673	2471.0724	5623.14
2500	2948.2285	6582.7259	5645.3462	22638.99
3000	3679.0432	7125.2364	6757.8211	12800.355
3250	4228.2119	7091.1931	6994.6198	15097.77
3500	4242.9165	12731.7231	10572.4673	15284.728

Figure 6 shows graphical representation of table 4.

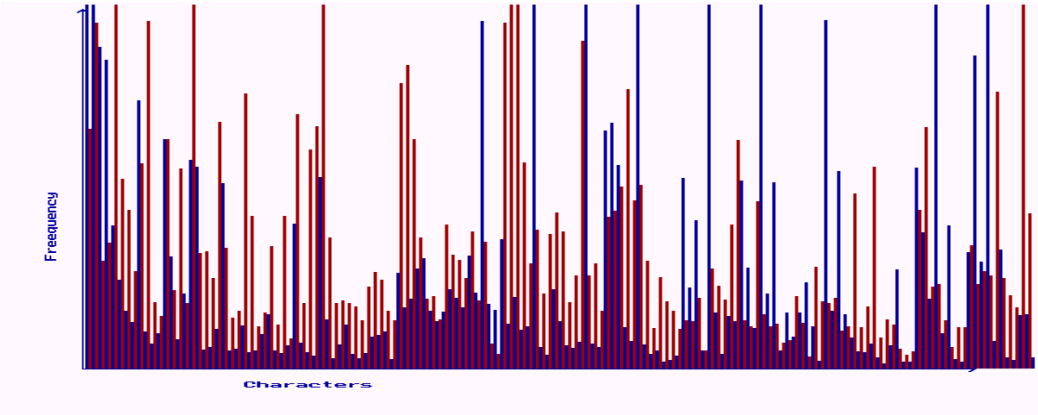

Figure 6. Chi-Square value against stream size

Table 6 shows total number of iteration needed and number of data being transferred for MLP key generation process with different numbers of input(N) and activated hidden(H) neurons and varying synaptic depth(L).

Table 5. Data Exchanged and No. of Iterations For Different Parameters Value

No. of Input Neurons(N)	No. of Activated Hidden Neurons(K)	Synaptic Weight (L)	Total No. of Iterations	Data Exchanged (Kb)
5	15	3	624	48
30	4	4	848	102
25	5	3	241	30
20	10	3	1390	276
8	15	4	2390	289

Following figure 7. Shows the snapshot of MLP key simulation process. This snapshot shows the tunning process of two multilayer perceptron with 4 hidden neurons, 4 input neurons and 6 as a weight value with hebbian learning rule.

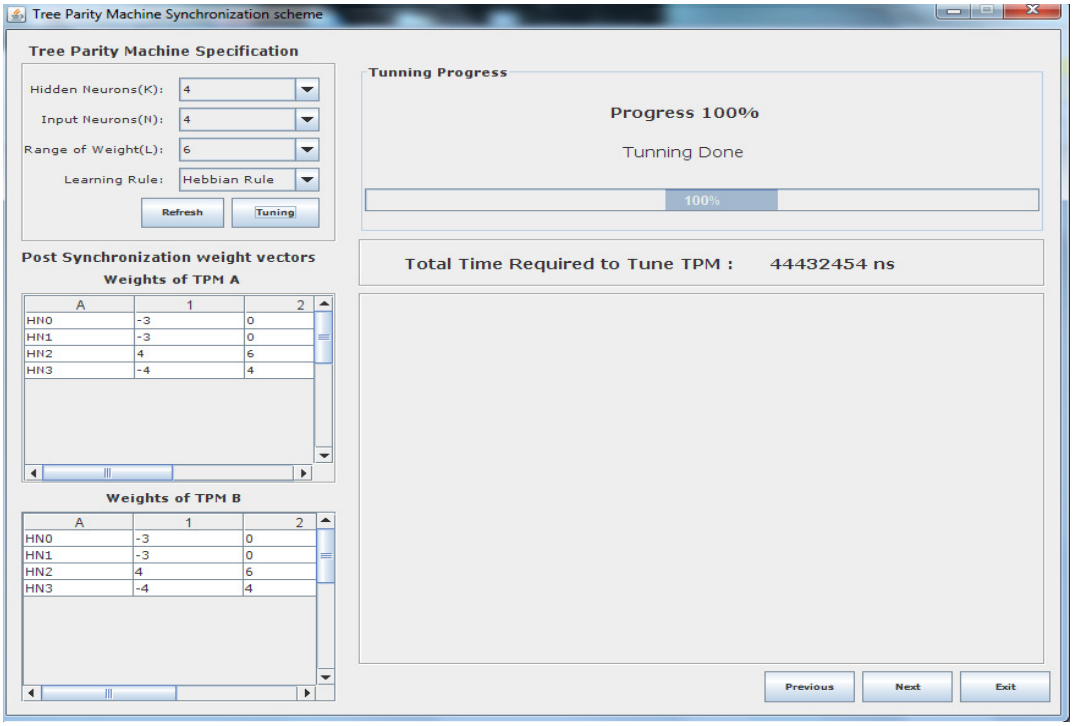

Figure 7. MLP Key Simulation Snapshot

Figure 8 shows the encryption and decryption time of a .txt file. File size taking as a bytes and encryption/ decryption time represnts as a nanosecond.

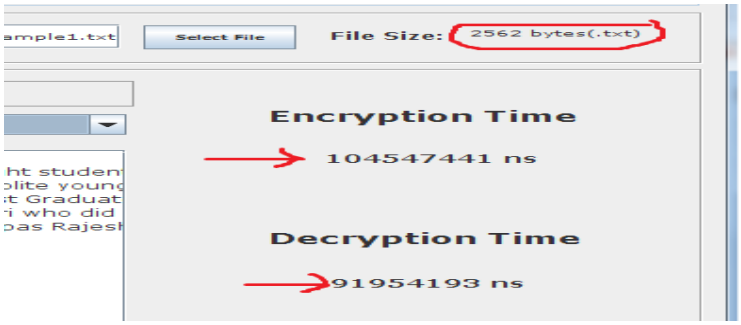

Figure 8. Snapshot of Encryption and decryption time

Figure 9 shows the memory heap representation of the key generation technique. The violet color area represents the memory that has alreay been allocated. Another color represents the total available memory.

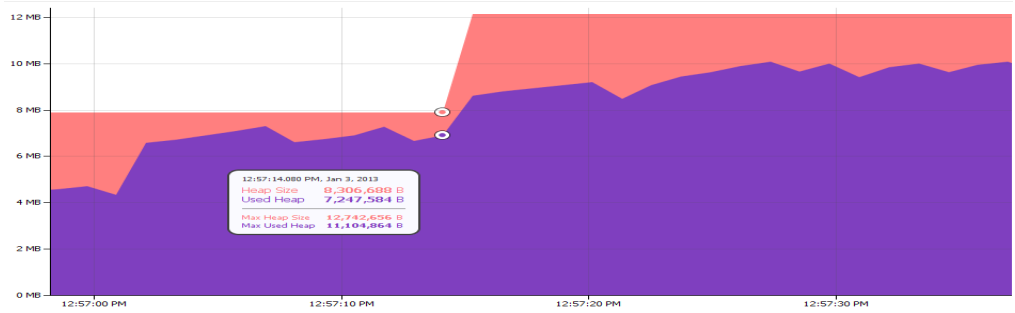

Figure 9. Memory Map for whole application

Figure 10 shows the memory allocation gantt chart during key generation process.

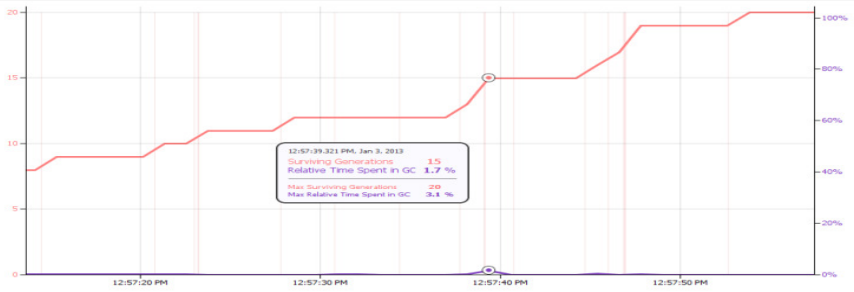

Figure 10. Memory Gantt Chart

9. ANALYSIS

From results obtained it is clear that the technique will achieve optimal performances. Encryption time and decryption time varies almost linearly with respect to the block size. For the algorithm presented, Chi-Square value is very high compared to some existing algorithms. A user input key has to transmit over the public channel all the way to the receiver for performing the decryption procedure. So there is a likelihood of attack at the time of key exchange. To defeat this insecure secret key generation technique a neural network based secret key generation technique has been devised. The security issue of existing algorithm can be improved by using MLP secret session key generation technique. In this case, the two partners A and B do not have to share a common secret but use their indistinguishable weights or output of activated hidden layer as a secret key needed for encryption. The fundamental conception of MLP based key exchange protocol focuses mostly on two key attributes of MLP. Firstly, two nodes coupled over a public channel will synchronize even though each individual network exhibits disorganized behaviour. Secondly, an outside network, even if identical to the two communicating networks, will find it exceptionally difficult to synchronize with those parties, those parties are communicating over a public network. An attacker E who knows all the particulars of the algorithm and records through this channel finds it thorny to synchronize with the parties, and hence to calculate the common secret key. Synchronization by mutual learning (A and B) is much quicker than learning by listening (E) [10]. For usual cryptographic systems, we can improve the safety of the protocol by increasing of

the key length. In the case of MLP, we improved it by increasing the synaptic depth L of the neural networks. For a brute force attack using K hidden neurons, K*N input neurons and boundary of weights L, gives (2L+1)KN possibilities. For example, the configuration K = 3, L = 3 and N = 100 gives us 3*10253 key possibilities, making the attack unfeasible with today's computer power. E could start from all of the (2L+1)3N initial weight vectors and calculate the ones which are consistent with the input/output sequence. It has been shown, that all of these initial states move towards the same final weight vector, the key is unique. This is not true for simple perceptron the most unbeaten cryptanalysis has two supplementary ingredients first; a group of attacker is used. Second, E makes extra training steps when A and B are quiet [10]-[12]. So increasing synaptic depth L of the MLP we can make our MLP safe.

10. SECURITY ISSUE

The main difference between the partners and the attacker in MLP is that A and B are able to influence each other by communicating their output bits τ^A & τ^B while E can only listen to these messages. Of course, A and B use their advantage to select suitable input vectors for adjusting the weights which finally leads to different synchronization times for partners and attackers. However, there are more effects, which show that the two-way communication between A and B makes attacking the MLP protocol more difficult than simple learning of examples. These confirm that the security of MLP key generation is based on the bidirectional interaction of the partners. Each partener uses a seperate, but identical pseudo random number generator. As these devices are initialized with a secret seed state shared by A and B. They produce exactly the same sequence of input bits. Whereas attacker does not know this secret seed state. By increasing synaptic depth average synchronize time will be increased by polynomial time. But success probability of attacker will be drop exponentially Synchonization by mutual learning is much faster than learning by adopting to example generated by other network. Unidirectional learning and bidirectional synchronization. As E can't influence A and B at the time they stop transmit due to synchrnization. Only one weight get changed where, σ_i = T. So, difficult to find weight for attacker to know the actual weight without knowing internal representation it has to guess.

11. FUTURE SCOPE & CONCLUSION

This paper presented a novel approach for generation of secret key proposed algorithm using MLP simulation. This technique enhances the security features of the key exchange algorithm by increasing of the synaptic depth L of the MLP. Here two partners A and B do not have to exchange a common secret key over a public channel but use their indistinguishable weights or outputs of the activated hidden layer as a secret key needed for encryption or decryption. So likelihood of attack proposed technique is much lesser than the simple key exchange algorithm.

Future scope of this technique is that this MLP model can be used in wireless communication. Some evolutionary algorithm can be incorporated with this MLP model to get well distributed weight vector.

ACKNOWLEDGEMENT

The author expresses deep sense of gratitude to the DST, Govt. of India, for financial assistance through INSPIRE Fellowship leading for a PhD work under which this work has been carried out.

REFERENCES

[1] Atul Kahate, Cryptography and Network Security, 2003, Tata McGraw-Hill publishing Company Limited, Eighth reprint 2006.

[2] Sarkar Arindam, Mandal J. K, "Artificial Neural Network Guided Secured Communication Techniques: A Practical Approach" LAP Lambert Academic Publishing (2012-06-04), ISBN: 978-3-659-11991-0, 2012

[3] Sarkar Arindam, Karforma S, Mandal J. K, "Object Oriented Modeling of IDEA using GA based Efficient Key Generation for E-Governance Security (OOMIG) ", International Journal of Distributed and Parallel Systems (IJDPS) Vol.3, No.2, March 2012, DOI : 10.5121/ijdps.2012.3215, ISSN : 0976 - 9757 [Online] ; 2229 - 3957 [Print]. Indexed by: EBSCO, DOAJ, NASA, Google Scholar, INSPEC and WorldCat, 2011.

[4] Mandal J. K., Sarkar Arindam, "Neural Session Key based Traingularized Encryption for Online Wireless Communication (NSKTE)", 2nd National Conference on Computing and Systems, (NaCCS 2012), March 15-16, 2012, Department of Computer Science, The University of Burdwan, Golapbag North, Burdwan –713104, West Bengal, India. ISBN 978- 93-808131-8-9, 2012.

[5] Mandal J. K., Sarkar Arindam, "Neural Weight Session Key based Encryption for Online Wireless Communication (NWSKE)", Research and Higher Education in Computer Science and Information Technology, (RHECSIT- 2012) ,February 21-22, 2012, Department of Computer Science, Sammilani Mahavidyalaya, Kolkata , West Bengal, India. ISBN 978-81- 923820-0-5,2012

[6] Mandal J. K., Sarkar Arindam, "An Adaptive Genetic Key Based Neural Encryption For Online Wireless Communication (AGKNE)", International Conference on Recent Trends In Information Systems (RETIS 2011) BY IEEE, 21-23 December 2011, Jadavpur University, Kolkata, India. ISBN 978-1-4577-0791-9, 2011

[7] Mandal J. K., Sarkar Arindam, "An Adaptive Neural Network Guided Secret Key Based Encryption Through Recursive Positional Modulo-2 Substitution For Online Wireless Communication (ANNRPMS)", International Conference on Recent Trends In Information Technology (ICRTIT 2011) BY IEEE, 3-5 June 2011, Madras Institute of Technology, Anna University, Chennai, Tamil Nadu, India. 978-1-4577-0590-8/11, 2011

[8] Mandal J. K., Sarkar Arindam, "An Adaptive Neural Network Guided Random Block Length Based Cryptosystem (ANNRBLC)", 2nd International Conference on Wireless Communications, Vehicular Technology, Information Theory And Aerospace & Electronic System Technology" (Wireless Vitae 2011) By IEEE Societies, February 28- March 03, 2011,Chennai, Tamil Nadu, India. ISBN 978-87-92329-61-5, 2011

[9] Mandal J. K., Sarkar Arindam, "Neural Network Guided Secret Key based Encryption through Cascading Chaining of Recursive Positional Substitution of Prime Non-Prime (NNSKECC)", International Confference on Computing and Systems, ICCS – 2010, 19–20 November, 2010,Department of Computer Science, The University of Burdwan, Golapbag North, Burdwan – 713104, West Bengal, India.ISBN 93-80813-01-5, 2010

[10] R. Mislovaty, Y. Perchenok, I. Kanter, and W. Kinzel. Secure key-exchange protocol with an absence of injective functions. Phys. Rev. E, 66:066102,2002.

[11] A. Ruttor, W. Kinzel, R. Naeh, and I. Kanter. Genetic attack on neural cryptography. Phys. Rev. E, 73(3):036121, 2006.

[12] A. Engel and C. Van den Broeck. Statistical Mechanics of Learning. Cambridge University Press, Cambridge, 2001.

[13] T. Godhavari, N. R. Alainelu and R. Soundararajan "Cryptography Using Neural Network " IEEE Indicon 2005 Conference, Chennai, India, 11-13 Dec. 2005.

[14] Wolfgang Kinzel and ldo Kanter, "Interacting neural networks and cryptography", Advances in Solid State Physics, Ed. by B. Kramer (Springer, Berlin. 2002), Vol. 42, p. 383 arXiv- cond-mat/0203011, 2002

[15] Wolfgang Kinzel and ldo Kanter, "Neural cryptography" proceedings of the 9th international conference on Neural Information processing(ICONIP 02).

[16] Dong Hu "A new service based computing security model with neural cryptography"IEEE07/2009.

Arindam Sarkar

INSPIRE FELLOW (DST, Govt. of India), MCA (VISVA BHARATI, Santiniketan, University First Class First Rank Holder), M.Tech (CSE, K.U, University First Class First Rank Holder). Total number of publications 25.

Jyotsna Kumar Mandal

M. Tech.(Computer Science, University of Calcutta), Ph.D.(Engg., Jadavpur University) in the field of Data Compression and Error Correction Techniques, Professor in Computer Science and Engineering, University of Kalyani, India. Life Member of Computer Society of India since 1992 and life member of cryptology Research Society of India. Dean Faculty of Engineering, Technology & Management, working in the field of Network Security, Steganography, Remote Sensing & GIS Application, Image Processing. 25 years of teaching and research experiences. Eight Scholars awarded Ph.D. and 8 are pursuing. Total number of publications 267.