Cryptic Neural Nets

Nitesh Goel[1] and Arpana[2]

M.Tech Scholar, SGI, Samalkha, Kurukshetra University, India
Associate Professor, SGI, Samalkha, Kurukshetra University, India

***Abstract*- The main purpose of a cryptography implementation is to make it incomprehensible to take an encrypted text and reproduce the original plain text without the corresponding correct decryption key. With good cryptography, the messages should be encrypted by using incredibly long keys and using encryption algorithms such that brute force attacks against the algorithm or the keys are impossibly difficult. Many cryptic processes are used based on number analysis and manipulation but also suffer from the disadvantage of complexity, time factor and computation multiplicity. To override the above disadvantages, ANN is an adaptive system that changes its structure based on external or internal information that flows through the network. This paper hereby aims to tap into the ANN approach towards the cryptic key exchange encountered with non-classical computing.**

***Keywords*—Neural Networks, Synchronization, key exchange, Cryptography, encryption, decryption**

I. INTRODUCTION

Cryptography, derived from Greek word, *kryptos*, meaning “secret” and *graph*, meaning “writing”. Cryptography is the art of changing any information into incomprehensiveness in such a way so as to allow only a secret method of uncovering this information. Good cryptography gets its security by using really long keys and using sturdy encryption mechanisms that are resistant to multiple forms of attacks.

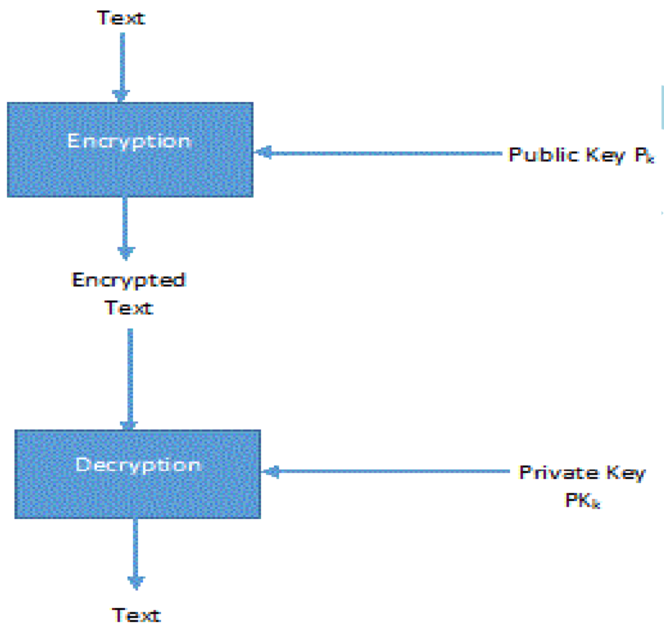

Figure -1

The point of interest in the cryptography and cryptanalysis is the confidential transmission of information or message between two communicating parties. Anyone or anything that is snooping on the communication channel should not be able to obtain the encoded confidential information. Nowadays, a common secret key based on number computation could be created over a public medium accessible to all but it might not be possible to calculate the discrete logarithm of sufficiently large numbers.

Important approaches used in cryptography include asymmetric and symmetric encryption. In symmetric encryption, two communicating parties share a common encryption-decryption key. The sender encrypts the original message (Text), referred as plain text, using a key (P_k) to generate incomprehensive message, referred as cipher text (Encrypted Text). In asymmetric encryption, a pair of keys are involved, one for encryption and second for decryption. Bits are always used to measure the length of the keys used in cryptanalysis. It is directly proportional to the possible number of keys and hence more secure is the algorithm.

An artificial neuron network (ANN) is a computational model based on the structure and functions of biological neural networks. Information that flows through the network affects the structure of the ANN because a neural network changes - or learns, in a sense - based on that input and output. ANNs are considered nonlinear statistical data modeling tools where the complex relationships between inputs and outputs are modeled or patterns are found.

The development of ANNs comes from simulating intelligent tasks which are performed by human brain. They are widely used by major soft computing techniques that have the capability to capture and model complex input and output relationships of any system. The advantages of ANNs are the ability to generalize results obtained from known situations to unforeseen situations, the fast response time in operational phase, the high degree of structural parallelism, reliability and efficiency. If a set of input-output data pairs which belongs to a problem is available, ANNs can learn and exhibit good performance. For these reasons, application of ANNs has emerged as a pivotal area of research, since their adaptive behaviors have the potential of modeling strongly nonlinear characteristics.

The basic computational element is generally known as a neuron or a node. It receives input from an external source. Each input has a weight associated with it (w), which can be adapted to model synaptic learning in ANNs. The node usually computes some generic function (f) which is a weighted sum of all the inputs of the network. Tree Parity

Machines, are multi-layer feed-forward networks. Their general structure is shown below:

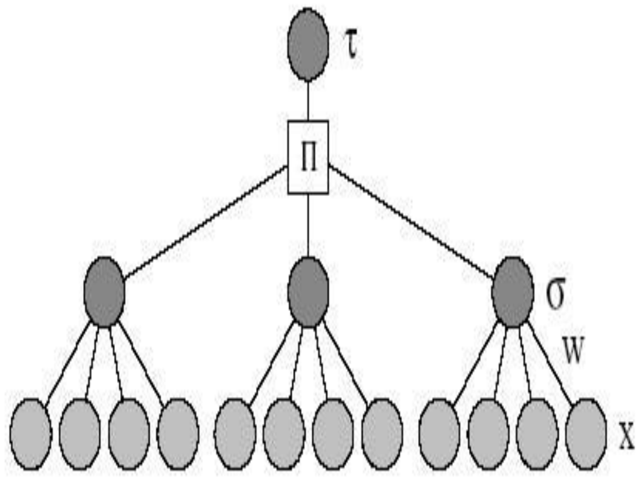

Figure-2

Such a neural network consists of *K* hidden units, which are perceptrons with independent receptive fields. Each one has *N* input neurons and one output neuron. All input values are binary.

Neural network based cryptography is a branch of cryptography dedicated to analyzing the application of stochastic algorithms, especially neural network algorithms, for use in encryption and cryptanalysis. Neural networks are well known for their ability to selectively explore the solution space of a given problem. This feature finds a natural niche of application in the field of cryptanalysis. Considering the search for successful and easy-to-use secure cryptanalytic measures, we focus the research to Neural Network – Secure Cryptography implementation providing a method, which would be exceedingly difficult to break without having exact knowledge of the methodology behind it.

II. RELATED WORK

There has been an increasing interest in the application of different classes of neural networks to problems related to cryptography in the past few years. Recent works have examined the use of neural networks in cryptosystems. This can be categorized into three sections:

1. Synchronization neural networks
   Synchronization of neural networks by mutual learning and its application to cryptography, 2004
   Einat Klein presented a secured cryptography secret-key based on neural network in a public channel. The proposed model has two neural networks that are trained on their alternate output synchronized to an equal time dependent weight vector through a chaos synchronization system that starting from different initial conditions. The system combined the neural networks with the logistic chaotic map. Both the partners used their neural networks as input for the logistic maps which generated the output bits to be learned, by mutually learning. The two neural networks approach each other and generated a matching signal to the chaotic maps. The chaotic synchronization applied in the neural cryptography enhanced the cryptography systems and improved the security.
2. Chaotic Neural networks
   Cryptography based on delayed chaotic neural networks, 2006

   Wenwu Yu proposed an encryption techniques based on the chaotic hopfield neural networks with time varying delay. The chaotic neural network is used for generating binary sequences for masking the plaintext. The binary value of the binary sequence chooses the chaotic logistic map randomly, that used for generated the binary sequences. The plaintext is masked by switching of the chaotic neural network maps and permutation of generated binary sequences. Simulation results show that the proposed chaotic cryptography is more functional in the secure transmission of large multi-media files over public data communication network.

3. Multi-layer Neural networks
   Implementation of neural - cryptographic system using FPGA, 2011
   Karam M. presented a stream cipher system based on pseudo Random Number Generator (PRNG) through using artificial Neural Networks (ANN). The PRNG model has a high statistical randomness properties for key sequence using ANN. The proposed neural pseudo random number generator consists of two stages; the first stage is generating a long sequence of patterns from perfect equation and initial value. So these patterns possess the randomness and unpredictable properties. The total number of equations and initial values depend on the number of bits that represented the initial value, the second stage is an artificial neural network (ANN) that gets the outputs of the previous stage and set it as input to the NN.

III. SYSTEM DESIGN

The basic idea is to employ synchronization in artificial neural networks using a cryptographic protocol using key exchanges. Considering two entities engaged in exchanging confidential message using an unsecured medium. In fact, assume it is a top-secret piece of information. Considering they can't use asymmetric algorithms such as RSA and cannot meet up to decide upon a mutual agreed singular key for this information; therefore, in order to protect the confidential message against network eavesdropping or network sniffing, or in simple words, snooping, the sender entity use symmetric encryption algorithm to encrypt the confidential information.The other party, upon receiving it, must be aware of the sender's encryption key to use this information.

A. BASICS OF CRYPTED NEURAL NET

In this paper, using neural networks, an experimental study was conducted in cryptanalysis, which specifically involves:

The topology design of the neural network; • Designing the methods of the training algorithm of the neural network;

• Designing the training set based on the learning rules.

The Tree Parity Machines (TPM), one for each of the communicating parties, are pivotal to the cryptography process or the key exchange protocol. The learning rules which are used for synchronizing Tree Parity Machines, share a common structure. Therefore, they can easily be shown by a single equation:

$$w^{+}_{i,j} = g(w_{i,j} + f(\sigma_i,\tau^A,\tau^B)x_{i,j})$$

The differentiating factor between the learning rules is, whether and how the output σ_i of a hidden unit in network affects $\_w_{i,j} = w^{+}_{i,j} - w_{i,j}$.

At the beginning of the synchronization process both the Tree Parity Machines start with randomly chosen and uncorrelated weight vectors $w_i^{A/B}$. In each time step *K*, public input vectors xi are generated randomly and the corresponding output bits $\tau^{A/B}$ are calculated.

There exists three different types of learning rules involved in the synchronization of the corresponding weights of the Tree Parity Machines:

- Both the network or the TPM learn from each other in the case of Hebbian learning rule:

$$w^{+}_{i,j} = g(w_{i,j} + x_{i,j}\tau\Theta(\sigma_i\tau)\Theta(\tau^A\tau^B))$$

- Another way could be that both networks are trained with the opposite of their output. This can be achieved by the anti-Hebbian learning rule:

$$w^{+}_{i,j} = g(w_{i,j} - x_{i,j}\tau\Theta(\sigma_i\tau)\Theta(\tau^A\tau^B))$$

- The set value of the output is same for all participating network or machines, then it is the random-walk learning rule:

$$w^{+}_{i,j} = g(w_{i,j} + x_{i,j}\,\Theta(\sigma_i\tau)\Theta(\tau^A\tau^B))$$

In the above equations, the significantly important function is theta(Θ). $\Theta(a, b) = 0$ if $a <> b$; else $\Theta=1$. The *g(...)* function keeps the weights in the range (*-L....+L*). *x* is the input vector and *w* is the weights vector.

feature is significantly useful for the cryptic implementation while synchronizing the neural networks or the Tree Parity Machines.

B. THE DESIGN BASICS

A common sequencing of input vectors is required for both the Tree Parity Machines looking to achieve synchronization of the system on the basis of mutual learning. This way, an authentication mechanism for the key exchange can be implemented using the neural net.

Both the Tree Parity Machines at each of the involved parties in the exchange, are initialized with secret seed. Both the TPMs use identical pseudo-random number generator separately. Since the secret seed is shared, the same ordering of bits are produced by the TPMs, which are required in the synchronization process. Thus, synchronization is achieved between the two machines without any type of communication over the unsecure medium.

Any intruder or eavesdropper cannot penetrate the synchronization process owing to the fact that the input vectors involved are not available. Thus, both the involved parties must be aware of the state having the secret seed used during initialization. Since this unique state cannot be evaluated from the public shared bits, it is a no-information-available protocol.

IV. DESIGN IMPLEMENTATION

The below mentioned technique consisting of several steps for ANNS, or particularly, Tree Parity Machines, for unique key generated for exchange over public medium are employed, which would be essentially communicated during the entire cryptanalysis process:

1. *Parameterization*: First up, the neural network or the parity machines parameters are determined, i.e.: k - the number of hidden layers in the neural net, n- the input layer corresponding to each of the hidden layers in the neural net, l- refers to the range of synaptic weight values to be used in the neural net.
2. *Initialization*: Using the above mentioned parameters, random initialization of the weights in the network, or the secret seed state initialization.
3. *Recursive*: Repeat the following steps (4 -7) until synchronization occurs.
4. *Compute neuron values*: Calculate the inputs of the hidden units in the neural machines.
5. *Generation*: The generation of the output bit and exchange between the two participating neural machines X and Y.
6. *Comparison and rule application*: If the generated outputs of both the tree parity machines are identical, i.e. $\tau^X = \tau^Y$, then the corresponding weights of these neural machines are updated using the aforementioned random-walk learning rule, Hebbian learning rule and Anti-Hebbian learning rule.
7. After synchronization is completed, the synaptic weights for both the machines are the same.
8. The weights are used as secret exchange protocol key in cryptic communication. In every case, the above mentioned

Hence, one can clearly deduce that the above mentioned process is a straight implementation of bidirectional learning.

V. CONCLUSION

The crypted Neural Net$_s$ paves the way for the next level of development in cryptanalysis techniques. In this technique,

the transmission only involves the input and the output vectors over the unsecure or public medium. Thus, the synchronization process encapsulates each party having a knowledge of the secret seed state of only its own Tree Parity Machine. It can be termed as a confidential-key technique, where is key is entirely dependent on the architecture of the network and weights imbibed in them. If these parameters, viz., architecture and the weights, are known, then the cryptanalysis is a trivial process. Another important point to note that both the encryption and decryption involve weights and the architecture. Missing either of the two is not enough for the eavesdropper to break it. Thus, this system becomes entirely difficult to crack unless the methodology in entirely known.
However, there still exists an insignificant probability factor that an intruder might be successful before the engaged parties have completed their procedure of key exchanging owing to the stochastic nature of the synchronization process. But fortunately, this probability lowers exponentially with increase in synaptic depth for almost every possible combination of the applied learning rules.

REFERENCES

[1] "*A New Technique on Neural Cryptography with Securing of Electronic Medical Records in Telemedicine System*" – N.Prabakaran, Department of Mathematics, Anna University, 2008.

[2] A. Forouzan., "*Cryptography and Network Security*", First Edition. McGraw-Hill, (2007), USA.

[3] AtulKahate (2009), "*Cryptography and Network Security*", second edition, McGraw-Hill.

[4] William Stalling, "*Network Security Essentials (Applications and Standards)*", Pearson Education, 2004.

[5] Fausett, L.V. 1994 "*Fundamentals of Neural Networks*", Prentice-Hall, Inc., Englewood Cliffs, New Jersey.

[6] Adel A. El-Zoghabi, Amr H. Yassin, Hany H. Hussien, "*Survey Report on Cryptography Based on Neural Network*", IJETAE, 2013.

[7] Kinzel, W., 2002,"*Theory of Interacting Neural Network*".

[8] W. Kinzel, R. Metzler, and I. Kanter. "*Dynamics of interacting neural networks*". J. Phys. A: Math. Gen., 33(14):L141–L147, 2000

[9] L. Ein-Dor and I. Kanter. "*Confidence in prediction by neural networks*". Phys. Rev. E, 60(1):799–802, 1999.

[10] T. L. H. Watkin. "*Optimal learning with a neural network*". Europhys. Lett., 21(8):871–876, 1993.

[11] E. Klein, R. Mislovaty, I. Kanter, and W. Kinzel. "*Public-channel cryptography using chaos synchronization*". Phys. Rev. E, 72:016214, 2005.

[12] A. Klimov, A. Mityaguine, and A. Shamir. "*Analysis of neural cryptography*". In Y. Zheng, editor, Advances in Cryptology—ASIACRYPT 2002, page 288. Springer, Heidelberg, 2003.