# Symmetric Cryptography using Neural Networks

Naveena Vempada[1], Dr. Sivakoti Satyanarayana[2]
[1]*M.tech Scholar, Dept. of CSE, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India.*
[2]*Professor, Dept. of CSE, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India.*

***Abstract-*** Cryptography is a skill of sending the data in such a form that only those for whom it is intended can read it. There are number of methods to perform cryptography, one of such method is Neural Cryptography. Neural Cryptography is an emergent field that aims to combine cryptography with Neural Networks for applications in cryptanalysis and encryption. In this paper, show Neural Networks is capable of performing symmetric encryption in an adversarial setting and improve on the known literature on this topic. And also show that Neural Networks is capable of detecting known cryptographically insecure communication.

## I. INTRODUCTION

Symmetric encryption is a form of encryption in which the sender and receiver use the same key to encrypt a plaintext and decrypt the corresponding cipher text. Traditionally, symmetric encryption algorithms have used either block or stream ciphers. However, it has been demonstrated that in a system of neural networks, with end-to-end adversarial training, they can learn how to perform forms of 'encryption' and 'decryption' without the use of a specific cryptographic algorithm.

Tree parity machine

The parity machine (PM) is a neural network applied in cryptography to generate a secret key. It is also used for a key exchange protocol. The TPM network is in fact an FNN that has input layer neurons constructed in the McCulloh-Pitts model (Protić, 2015), (Dolecki and Kozera, 2015). In the second layer, the network has neurons with specific activation functions. The outcome of the output neuron is the results of the entire PM network. Each PM network is described by three parameters: the number of hidden neurons - K, the number of input neurons connected to each hidden neuron - N, and the maximum value for weight {-L, ... L}. A PM consists of KN random input elements xji = ±1, j = 1…N, K binary hidden units σi = ±1, i = 1,…, K, and one binary output unit τ=Πiσi, where σi is determined via the function σi = sign(Σjwjixji). A PM that has three neurons in the hidden layer (K=3) is called the three parity machine.

The advantage of neural cryptography is that the algorithm used in generating a secret key can be a simple perception of the Tree Parity Machine (TPM). Synchronization ofTPMs bymutual learning only works if both machines receive a common sequence of (random) input vectors. For that purpose, each communication party uses a separate, but identical pseudo-random number generator (PRNG). By mutual learning, two parties synchronize their networks without transmitting inputs over a public channel. Having reached full synchronization, parties can authenticate each other by knowing weight vectors which are identical, and represent a secret key.

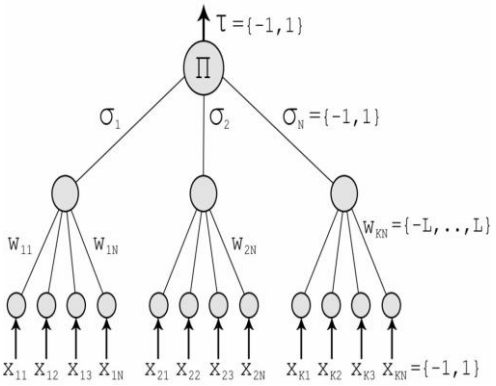


Fig.1: Tree parity machine model

## II. EXISTING SYSTEM

The basic need to provide security is using cryptography. Cryptosystem are commonly used for protecting the integrity, confidentiality, and authenticity of information resources. The design of cryptography algorithm is complicated by the fact that a variety of cryptanalytic attacks are available that can often be successfully used to recover the key or the plaintext. Cryptography has two style of encrypting data; symmetrical and asymmetrical. Symmetric encryptions use the same key for encryption and decryption process, and also can be defined as a secret-key, shared-key, and private-key. Asymmetric cryptography uses different encryption keys for encryption and decryption process. In this case an end user on a network, public or private, has a pair of key; one for encryption and one for decryption. These keys can be identical as a public and a private key. Key generation is the most significant issue in cryptography technique. In recent times wide ranges of techniques are developed to protect data and information from eavesdropper.

## III. PROPOSED SYSTEM

1. **Interacting Neural Networks and Cryptography**

i. Two identical dynamic systems, starting from different initial condition can be synchronized by a common external signal which is coupled to the two systems. Two networks which are trained on their mutual output can synchronize to a time dependent state of identical synaptic weights. ii. This phenomenon is also applied to cryptography. Neural networks learn from examples. Training means, that synaptic weights adopt by simple rules to the input/output pairs. After the training phase the neural network is able to generalize: it can classify the input pattern which does not belong to the training set. iii. The two patterns A and B do not have to share the common secret key but use their identical weights as a secret

key need for encryption. iv. In neural network an attacker E who knows all the details of the algorithm and record any communication transmitted through this channel finds it difficult to synchronize with the parties, and hence to calculate the common secret key.

2. **Neural Key Exchange:** The most used protocol for key exchange between two parties A and B in the practice is Diffie-Hellman protocol. Neural key exchange, which is based on the synchronization of two tree parity machine, should be a secure replacement.

**Tree Parity Machine**

**3. Tree Parity Machine** is special type of multi-layer feed-forward neural network. It consist of one output neuron, K hidden neurons and K*N input neurons. Inputs to the networks take 3 values: xij∈ −1,0,1 The weights between input and hidden neurons take the values: wij∈ $-L$, … ,0, … . , $+L$ Output values of each hidden neurons is calculated as a sum of all multiplication of input neurons and these weights: $\sigma i$ = ( $ijNj$=1 ) Sig num is a simple function, which returns -1,0 or 1: Sgn (x)= −1 $if x$< 0, 0 $if x$ = 0, 1 $if x$> 0. If the scalar product is 0, the output of the hidden neuron is mapped to -1 in order to ensure a binary output values. The output of neural network is then computed as the multiplication of all values produced by hidden elements: $\tau$ = $\sigma iKi$=1 Output of the tree parity machine is binary. 4. Secret Key Generation The different stages in the secret key generation procedure which is based on neural networks can be stated as follows: 1. Determination of neural network parameter: K the number of hidden layers units, n the input layer units for each hidden layer unit, l the range of synaptic weight values is done by the two machine A and B. 2. The network weight to be initialized randomly. 3. The following steps are repeated until synchronization occurs. 4. Inputs are generated by the third party (key distribution center). 5. The inputs of the hidden units are calculated. 6. The output bit is generated and exchange between the two machine A and B. 7. If the output vectors of both the machine agree with each other than the corresponding weights are modified using the learning rules. 8. When synchronization is finally occurred, the synaptic weights are same for both the networks. And these weights are used as secret key.

## IV. SYSTEM DESIGN

**UML**

Unified Modeling Language is the one of the most exciting tools in the world of system development today. Because UML enables system builders to create blue prints that capture their visions in a standard, easy to understand way and communicate them to others. The UML is brain child of Grady Brooch, James Rumbaugh and Ivar Jacobson.

The UML is a language for

1 Visualizing
2 Specifying
3 Constructing
4 Documenting

These are the artifacts of a software-intensive system.The abbreviation for UML is Unified Modeling Language and is being brought of a designed to make sure that the existing ER Diagrams which do not serve the purpose will be replaced by this UML Diagrams where in these language as its own set of Diagrams.

Some of the Diagrams that help for the Diagrammatic Approach for the Object Oriented Software Engineering are

- Class Diagrams
- Use Case Diagrams
- Sequence Diagrams
- State Chart Diagrams
- Activity Diagrams

Using the above mentioned diagrams we can show the entire system regarding the working of the system or the flow of control and sequence of flow the state of the system and the activities involved in the system.

**Components of the UML**

The UML consists of a number of graphical elements that combine to form diagrams. Because it's a language, the UML has rules for combining these elements. The purpose of the diagrams to present multiple views of the system, and this set of multiple views is called a Model. A UML Model of a system is something like a scale model of a building. UML model describes what a system is supposed to do. It doesn't tell how to implement the system. The following are the main nine component Diagrams of U M L:

**Class Diagram**

A Class is a category or group of things that has similar attributes and common behavior. A Rectangle is the icon that represents the class it is divided into three areas. The upper most area contains the name, the middle area contains the attributes and the lowest areas show the operations. Class diagrams provides the representation that developers work from. Class diagrams help on the analysis side, too.

**Object Diagram**

An object is an instance of a class- A specific thing that has specific values of the attributes and behavior. A Rectangle is the icon that represents the object diagram but the name is underlined. The name of the specific instance is on the left side of the colon, and the name of the class is on the right side of the colon.

**Use-Case Diagram**

A Use-Case is a description of a systems behavior from a users stand point. For system developer this is a valuable tool: it's a tried-and-true technique for gathering system requirements from a users point of view. That is important if the goal is to build a system that real people can use. A little stick figure is used to identify an actor the ellipse represents use-case.

**State Diagram**

At any given time, an object is in particular state. One way to characterize change in a system is to say that its objects change the state in response to events and to time. The UML State Diagram captures this kinds of changes it presents the states an object can be in along with the transitions between the states, and shows the starting point and end point of a sequence of state changes.

A Rounded Rectangle represents a state, along with the solid line and arrow head that represents a transition. The arrow head points to the state being transition into. The solid circle symbolizes starting point and the bulls eye that symbolizes the end point.

**Sequence Diagrams**

In a functioning system objects interacts with one another and these interactions occur over time. The UML Sequence Diagrams shows the time based dynamics of the interaction. The sequence diagrams consists of objects represented in the usualway-as named rectangles (If the name underlined), messages represented as solid line arrows and time represented as a vertical progression.

**Activity Diagrams**

The state diagram shows the states of an object and represents activities as arrows connecting the states. The Activity Diagram highlights the activities. Each activity is represented by a rounded rectangle-narrower and more oval-shaped than the state icon. An arrow represents the transition from the one activity to the next. The activity diagram has a starting point represented by filled-in circle, and an end point represented by bulls eye.

**Collaboration Diagram**

An object diagram shows the objects and their relationships with one another. A collaboration Diagram is an extension of the object diagram. In addition to the associations among objects, the collaboration diagram shows the messages the objects and each other.

**Use case Model**

The **functional model**, represented in UML with use case diagrams, describes the functionality of the system from the user's point of view. Use cases are used during requirements elicitation and analysis to represent the functionality of the system. A use case describes a function provided by the system that yields a visible result for an actor. An actor describes any entity that interacts with the system. An actor can be human or an external system. In this proposed system actors are Administrator, Manager, Technical person and Customer.

Use cases consist of elements that lie inside the system and are responsible for the working i.e. the functionality and behavior of the system. Therefore, use cases are nothing but the actions that the system performs to generate results requested by the actors of the system.

**USE CASE DIAGRAMS**

UML provides the use case diagram to facilitate the process of requirements gathering. The Use case diagram models the interactions between the system's external clients and the use cases of the system. Each use case represents a different capability that the system provides the client. Stick figure represents an actor.

## V. RESULTS

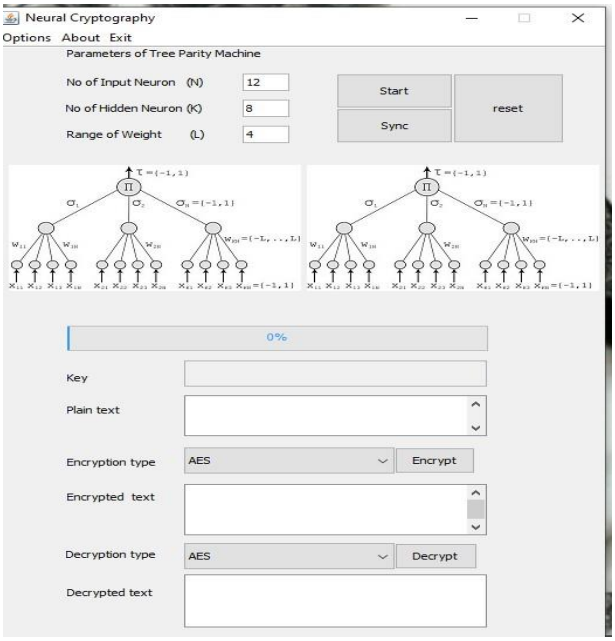


Fig.2: Input home page

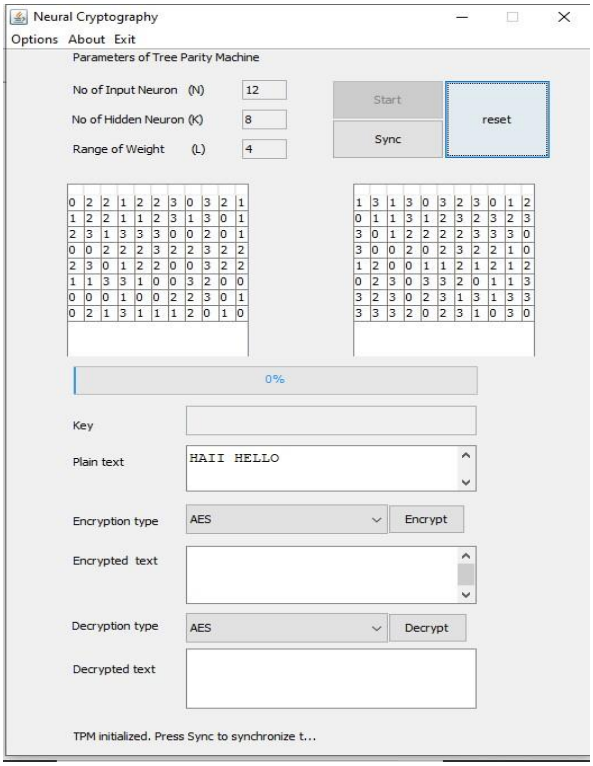


Fig.3: with input data

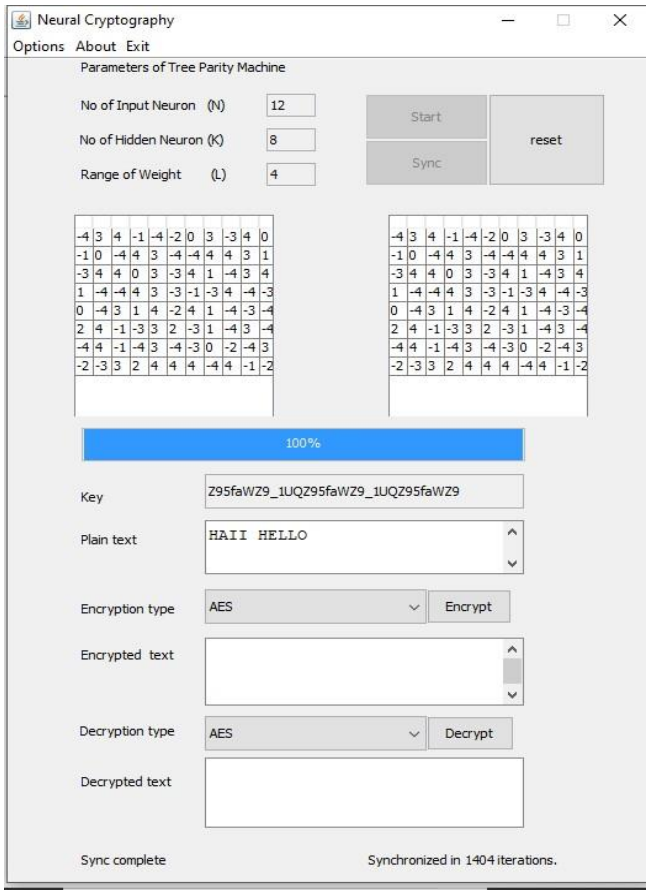


Fig.4: Plain text with key

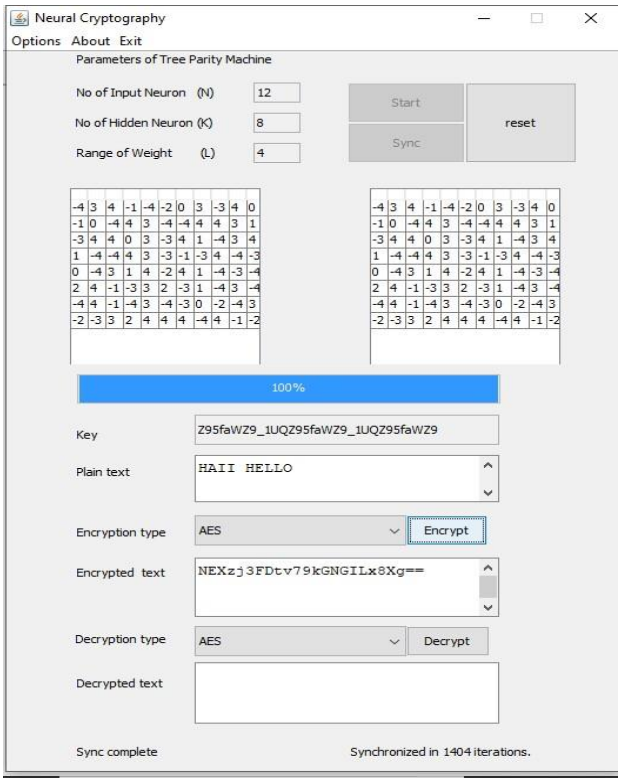


Fig.5: encryption process

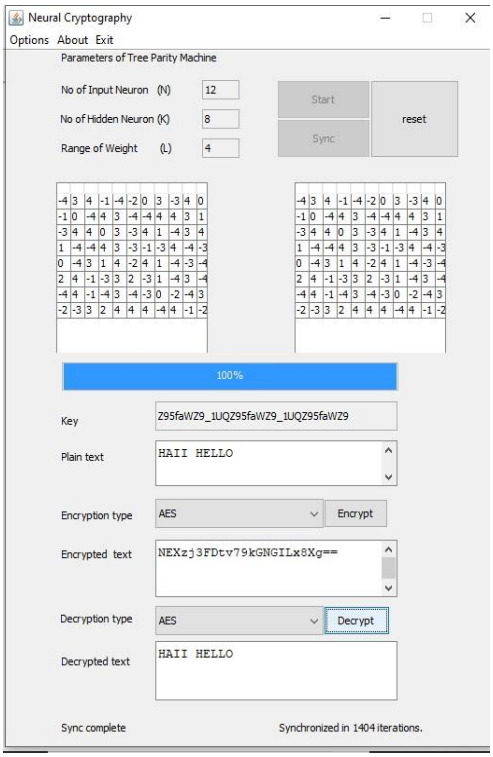


Fig.6: Decrypted data

## V. CONCLUSION

In the last decade, mutual learning based on the parity machine has become popular to be used for cryptography. In this paper, a three parity machine that is a special type of a feed forward neural network with three artificial neurons in the hidden layer, one output neuron and KN inputelements is presented. Inputs to the network are binary numbers, while the weights between inputs and hidden neurons take predefined values. The output of the hidden neuron is calculated as a weighted sum of all multiplications of inputs and weights. If the scalar product is zero, the output of the hidden neuron is mapped to -1, otherwise it is mapped to 1, in order to ensure a binary output from the network. The output bit of the network is a product of the three bits of the hidden units. Mutual learning is used for synchronization between two parties thatcommunicate across a public or private channel. During the synchronization, both communication parties use the same tree parity machines, receive common binary inputs generated randomly from the same random number generator, and exchange output bits. Adjusting the weights according to the learning rules leads to full synchronization in a finite number of steps. Networks trained on their mutual inputs synchronize to an identical time dependant weight vectors. This phenomenon is used to generate a secret key.

## VI. REFERENCES

[1]. Navita Agarwal, Prachi Agarwal, "Use of Artificial Neural Network in the Field of Security", MIT International Journal of Computer Science & Information Technology, Vol. 3, No. 1, 42-44, 2013.

[2]. Ajit Singh, Havir Singh, "Cryptography for Secret Key Exchange and Encryption with AES", International Journal of

Advanced Research in Computer Science and Software Engineering, Vol. 3, Issue. 5, 376-381, 2013.
[3]. Siddeq Y. Ameen, Ali H. Mahdi, "AES Cryptosystem Development using Neural Networks", Inernational Journal of Computer and Electrical Engineering, Vol. 3, No. 2, 315-318, 2011.
[4]. Eva Volna, Martin Kotyrba, Vaclav Kocian, Michal Janosek, "Cryptography based on neural network", Proceedings 6th European Conference on Modelling and Simulation, 2012.
[5]. N. Prabakaran, P. Vivekanandan, " A New Security on Neural Cryptography with Queries", International Journal of Advanced Networking and Apllication (IJAIA), Vol.2, No. 1, 60-69, 2011.
[6]. Wolfgang Kinzel, IdoKanter, "Neural Cryptography", Proceedings TH2002 Supplement, Vol. 4, 147-153, 2003.
[7]. Einat Klein, Rachel Mislovaty, Idokanter, Andreas Ruttor, Wolfgang Kinzel, "Synchronization of neural network by mutual learning and its application to cryptography", International Proceeding of: Advances in Neural Information Processing System 17, Neural Information Processing System NIPS, 2004.
[8]. R. M. Jogdand, Sahana S. Bisalapur, "Design of an efficient neural key generation", International Journal of Artificial Intelligence & Application (IJALA), Vol. 2, No. 1, 60-69, 2011.
[9]. Pratap Singh, Havir Singh, "Cryptography in Structure adaptable digital neural networks", National monthly refereed journal of research in science & technology, Vol. 1, Issue. 12, 35-44,2012.
[10]. William Stalling, " Cryptography and Network Security: Principles and Practicle, (5th Edition), Prentice Hall, 2010.
[11]. M. Arvandi, A. Sadeghian, "On the use of Recurrent Neural Networks to Design Symmetric Cipher", IEEE Computational Intelligence Magazine, pp. 42-53, May 2008.
[12]. Khalil Shihab, " A back propagation Neural Network for computer Network Security", Journal of computer science 2(9): 710-715, 2006.
[13]. Behrouz A. Forouzan, " Cryptography and Network Security", Tata McGraw-Hill, Special Indian Edition, 2007.
[14]. Meghdad Ashtiyani, Soroor Behbahani, Saeed Asadi, Parmida Moradi Birgani, " Transmitting Encrypted Data by Neural network", 2007 IEEE International Symposium on Signal Processing and Information Technology, pp. 385-389, 2007.
[15]. Seref S. Neclao, " Neural Solution for Information Security", Politeknik Dergisi, Journal of Polytechnic, Vol. 10, No. 1, 21-25,2007.