DOI 10.29042/2019-5264-5268

Helix Vol. 9 (4): 5264- 5268

Encryption Based on Neural Networks

[1]Alexey G. Isavnin, [2]Izida I. Ishmuradova, [3]Anton N. Karamyshev, [4]Denis M. Lysanov, [5]Irina I. Eremina
[1-5] Kazan Federal University
Email: IIIshmuradova@kpfu.ru

Received: 23rd July 2019, Accepted: 10th August 2019, Published: 31st August 2019

Abstract
Due to the active development of information technology, information is of great value to date. Any sphere of public life can be described by information, loss or modernization of which can lead to large losses. Thus, information becomes strategic resource of the state and business of all levels that are interested in its preservation. In addition to the natural risks of information loss (failure of technology, natural disasters, etc.), there is also the desire of criminal structures to illegally kidnap or update information. The problem of protecting information is extremely urgent nowadays. Companies require often applications with the most enhanced protection for sending messages. In this article, we implement a scheme for sending encrypted messages between several parties.On the basis of neural networks, we implement symmetric encryption with the distribution of keys. A scheme for synchronizing neural networks using a tree of a parity machine will be constructed.

Keywords
Neural Networks, Synchronization, Encryption, Tree of a Parity Machine, Key Distribution

Introduction

Due to the active development of information technology, it is of great value to date. Any sphere of public life can be described by information, loss or modernization of which can lead to large losses.
When transmitting information over the network, it is processed at each node, according to the rules specified in the data transfer protocol. Also, each node (router, switch, computer) is a system in which various rules of information processing can be present.
Thus, when information is transferred within a computer network, a dual problem arises: on the one hand, to ensure the security of information in a single system with unified rules for processing information, and, on the other, to monitor the integrity, confidentiality and availability of information in the aggregate of individual systems with different rules for processing it[1].
There are many modern applications working on the safety standards of industrial control systems that require the safe transfer of sensitive data. It is possible to create a connection using cryptographic protocols. This is done for the fastest and most efficient encryption method using symmetric encryption algorithms. The structure of symmetric encryption is clearly shown in Figure 1.

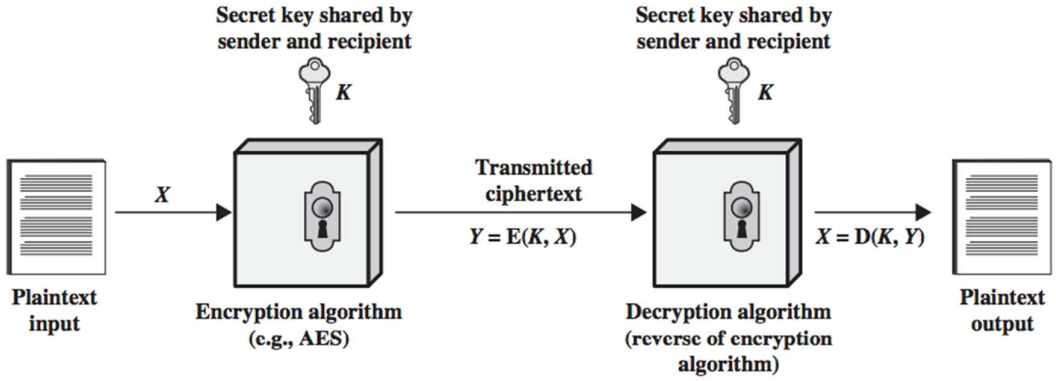

Figure 1: Structure of Symmetric Encryption Algorithms

Nowadays, one of the most important tools for automating network and communication security is encryption. The meaning of encryption is that data is transferred to the domain so that they become more robust to listen to the network. As we know, the two most basic methods of encryption are symmetric and asymmetric encryption. But here there are flaws. Symmetric encryption algorithms have an important key distribution problem. Two sides must have the same secret key to establish a symmetric secure connection. In order to exchange keys or other information, complete confidentiality is necessary to ensure that no one else can access the keys or their copies. This is the problem. There are some solutions to the problem of providing a key exchange scheme, but they, unfortunately, have some vulnerabilities. Therefore, it is very important to find a security key exchange scheme.
This article suggests the analysis and implementation of a key exchange scheme based on neural networks. The neural network is used to build an effective encryption system using constantly changing keys. Nowadays, a huge number of researchers are very interested with this method. Neural networks allow for a very powerful and general structure of the

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

representation of a nonlinear mapping from several input variables for several output variables. The neural network is considered as one of the most suitable choices for the functional forms used for encryption operations. Thus, attackers lose the ability to gain access to the key exchange protocol [2].

This article contains the following information: in the second section we consider in detail and describe the scheme for synchronizing artificial neutron networks using trees of parity machines.The third section describes the system for implementing the synchronization scheme and establishes a secure connection between two different parties.Further some results on experiments are summed up. The fourth section is devoted to the results and proposals for improving the model proposed by us.

Methods

Structure of synchronization of artificial neural networks using trees of parity machines.

By studying each other, it is possible to synchronize artificial neutron networks (ANNs).In order to achieve this, networks receive shared resources and share the results.Synchronization occurs with the possibility of reinforcement training. In order to establish a key exchange channel, it is possible to use ANN synchronization.Such a process is shown in Figure 2.

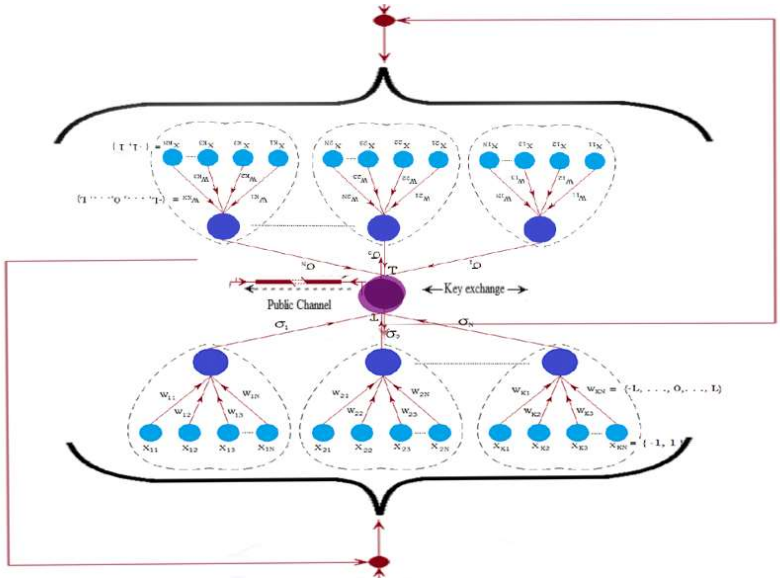

Figure 2: Channel for Key Exchange Using ANN Synchronization

In order for the synchronization to be performed, we used a special type of ANN tree, thus performing parity checking [3].Each side has its own side of the parity tree control machine.In other words, this is a special type of a multilevel neural network using feedback.The structure of such a network can be seen in Figure 3.

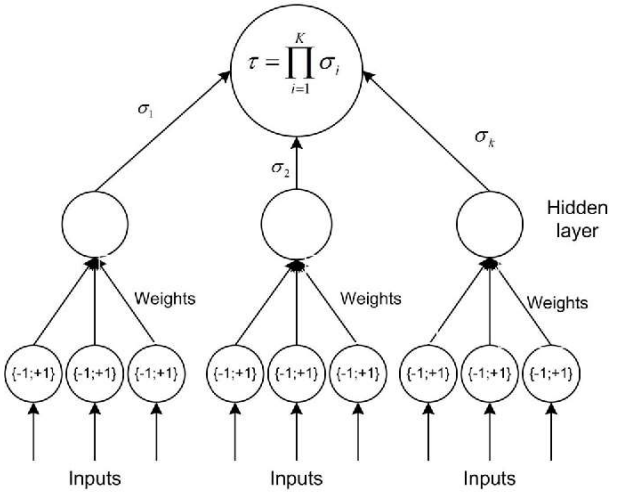

Figure 3: The ANN Structure for the Two Sides of the TPM.

As part of artificial neural networks the following: 1 input neuron, K – hidden neuron in the hidden layer and some more connections with each perception on the input layer.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

The total number of entries $X_{k,\check{n}}$ will be $K \times N$ where $1 \le k \le K$ $1 \le n \le N$ The inputs are randomly selected from the set $X_{k,n} \in \{-1; 1\}$.

The scales between the input and hidden neurons take the following:

$$W_{k,n} = \{-L, -L+1, ..., L\}$$

At the time when parity was introduced, the value L = 100 was selected.
To calculate the output value of each hidden neuron, it is necessary to sum the weighted sum of input neurons and weights.To calculate the output, signum activation was used.

$$\sigma_k = \operatorname{sgn}\left(\sum_{k=1}^{N} w_{kn} x_{kn}\right) \quad (1)$$

$$\operatorname{sgn}(x) = \begin{cases} -1, \ if \ x \le 0 \\ 1 \ , \ if \ x > 0 \end{cases} \quad (2)$$

The product of the value of hidden neurons is the output of the neural network, represented in the formula:

$$\tau = \prod_{k=1}^{\ddot{E}} \sigma_k \quad (3)$$

The output of the neural network allows both sides to establish the most secure channel. On each side are their own parity controls and artificial neural networks, as shown in Figure 3. In the first step, weights with random values are initialized. The second step generates a random input vector for parity checking. In the third step, the values of the hidden neurons and output neuron are calculated. The fourth step is comparing the values from two tree parity machines. At the fifth step both values are checked, if they are different, then it is necessary to return to the second step. The sixth step is obtaining identical values on different parity checks. And then in this case the Hebbian weights rules are applied. Both partners use parity with the same structure, and the parameters K, L, N are public. All the conditions of artificial neural networks are kept in strict confidence and begin with randomly chosen weights. In the process of synchronization, only the common outputs are transmitted through one common channel. And that is why for each participant of the process, the internal state of its own parity control is known [4].
To preserve the security of the key exchange protocol, a mandatory condition is the preservation of sensitive information [9].When the synchronization is completed, both process participants will be able to use the weight vectors as a shared secret key.

Results and Discussion

In this section, we develop software for teaching parity and creating the most secure channel between both participants. With the help of Python and Ruby, software was created.
The parity control algorithm allows the two parties to create a secure connection. All third-party addresses on the server with the implementation of the parity-learning algorithm. Public parameters K, N, L are also used here. As soon as the training begins, the two parties involved send the output to the server. In case the data does not match, the server informs the participants about it by sending a message. Then each side again has to assign new random output values. When the outputs are matched, both sides must send the hash of the memory hash to the server. If suddenly these values were unequal, the server again notifies the participants about this, giving the command to change the matrix of the weights by the training rule. In the case where the hashes are equal, the training of artificial neural networks ceases, and the hash or weight is further used as a key. Such a scheme can be clearly seen in Figure 4.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

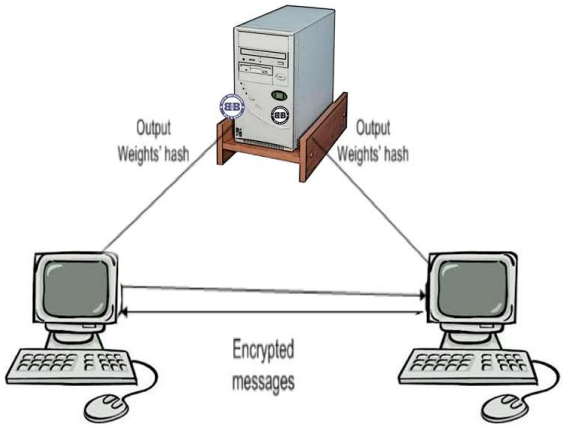

Figure: The Scheme of TPM Training

Sometimes it happens that the server acts as a second party. In this case, the problem will be that the comparator of the two parity checks becomes the protection of one server [5].

During the writing of the article, some experiments related to the training of parity control were carried out. In the experiments, the above schemes and various training rules were used.The results of the experiments are given in Table 1.

Name of the training rule	Hebbian	Anti - Hebbian	Random - walk
Number of attempts	40	20	30
Number of errors	3	10	17
Percent of errors (%)	13,3	50	57

Table 1: Results of Experiments

In the course of the experiment it was revealed that the use of the Random-walk training rule is most inefficient, but using the Hebbian rule on the contrary will help improve the results.

The process of learning in synchronization is very complex and very unique.This importance and uniqueness is obtained in connection with the fact that it is very difficult to achieve full equality of all weights without cycles. The progress of the parity learning process is shown in Figure 5.

```
[[-52, -37, 6, -59], [-52, -37, 6, -59]]
new inputs
[[-79, 7, -17, -27], [-79, 7, -17, -27]]
new inputs
new inputs
new inputs
[[-54, -35, 8, -61], [-54, -35, 8, -61]]
[[-81, 5, -15, -25], [-81, 5, -15, -25]]
new inputs
new inputs
[[-56, -33, 10, -63], [-56, -33, 10, -63]]
[[-83, 3, -13, -23], [-83, 3, -13, -23]]
new inputs
new inputs
[[-58, -31, 12, -65], [-58, -31, 12, -65]]
[[-85, 1, -11, -21], [-85, 1, -11, -21]]
new inputs
new inputs
[[-60, -29, 14, -67], [-60, -29, 14, -67]]
[[-87, -1, -9, -19], [-87, -1, -9, -19]]
new inputs
new inputs
[[-62, -27, 16, -69], [-62, -27, 16, -69]]
[[-89, -3, -7, -17], [-89, -3, -7, -17]]
new inputs
new inputs
[[-64, -25, 18, -71], [-64, -25, 18, -71]]
[[-91, -5, -5, -15], [-91, -5, -5, -15]]
new inputs
new inputs
[[-66, -23, 20, -73], [-66, -23, 20, -73]]
[[-93, -7, -3, -13], [-93, -7, -3, -13]]
```

Figure 5: Progress of the Learning Process of Parity

Further in Table 2 the results revealed in the course of experiments on some possibilities of training in parity control are presented.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495

The kind of training	Changing the weight	Changing the inputs	Changing the weights, and the inputs
Number of learning	32	13	5
Number of tries	50	50	50
Percent of learning (%)	64	26	10

Table 2: Results of Training in Parity Control

The experimental results showed the following: 64% of the parity check paths were studied from the weight change.This suggests that in this case a person who tried to hack the system will be defeated and cannot do it.

Summary

Often, problems arise in connection with the violation of TPM, creating at the same time difficult processes that require unconventional approaches to their solution. We also considered how important it is to protect networks and our confidential information from intruders. Therefore, we provided protection for the server and getting rid of the hash-transfer of the scales. In the course of the experiments, it was revealed how important balancing of weights is, and this is the main part of the training. In the future, there is an opportunity to calculate the number of steps, where two neural networks are fully synchronized. Such an action will leave the hash confidential, and use the remote server as a comparator of output values.

Implementing our system, we included a permanent update of the key, when any message was sent between two users, session keys were used [6]. Also, it would be appropriate to destroy the hash-exchange with the server. Doing this would be useful for removing the constant synchronization of the two NNs.

As an improvement, in the future we would like to establish the possibility of setting inputs with a certain sequence, and not randomly [10]. For example, as a binary set $\{-1;+1\}$, a fingerprint could be represented. During such important processes as authentication and encryption, a fingerprint would help us as a secret key. Based on fingerprints, you could enter a protocol with a digital signature, which is also a big plus.

Conclusions

In symmetric encryption algorithms, we were able to examine and analyze one of the new ways to solve the key exchange problem. This method is based on artificial neural networks on the TPM synchronization. We used a remote server acting as a comparator to apply this method for messaging. We conducted experiments and tested our system, and also brought forward our arguments and suggestions for improving further work. Such actions will allow to get rid of comparison keys on the server, make the system much safer and to prevent unpleasant incidents with intruders [7,8].

Acknowledgements

The work is performed according to the Russian Government Program of Competitive Growth of Kazan Federal University.

References

[1] S. Pissanetzky, Sparse matrix technology, London Academic Press, 1984.
[2] Ishmuradova I.I., Ishmuradova A.M. Stochastic modeling of economic activity of costs on Innovation of the organization of the Republic of Tatarstan, in the formation of business processes // RevistaPublicando – 2017. – Vol. 4 – No 12. (1) –P. – 545-559.
[3] Karimov S.A., Sibaeva G.R., Eremina I.I., Karamyshev A.N, Method of introducing the multidimensional concept of authorization SAP BW// Journal of Advanced Research in Dynamical and Control Systems – 10–13 Special Issue, – P. 536 – 540.
[4] Makhmutov I.I., Isavnin A.G., Karamyshev A.N.,Sych S.A., Classification approach in determination of knowledge in context of organization// Academy of Strategic Management Journal, – Volume 15, – Issue Special Issue, –1 January – 2016, – P. 40-46.
[5] R.L. Ackoff, The art of problem solving, John Wiley & Sons, 1978.
[6] T. T. Soong, Fundamentals of probability and statistics for engineers, John Wiley & Sons, 2004.
[7] Miftakhova A.R., R.Sibaeva, G.,Lysanov D.M., Karamyshev A.N. A development of an online monitoring system of the public transport// Astra Salvensis, Supplement No. 2/2017. – P. 545-556.
[8] W.H. Greene, Econometric Analysis, 7th ed. Prentice Hall, 2011.
[9] Makhmutov I.I, Murtazin I.A, Isavnin A.G. Karamyshev A.N., Methods and models of outsourcing// International Journal of Economic Perspectives. - 2017. - Vol.11, Is.3. - P.1620-1632.
[10] Eremina, Irina I.; Gazizov, Ilnaz F., Accounting and analysis of inventories of materials and production of companies// DILEMAS CONTEMPORANEOS-EDUCACION POLITICA Y VALORES – Issue 6.

  Helix E-ISSN: 2319-5592; P-ISSN: 2277-3495