A Key Agreement Protocol based on Spiking Neural P systems with Anti-Spikes

Mihail-Iulian Pleșa*[1], Marian Gheoghe[2], Florentin Ipate[1], and Gexiang Zhang[3]

[1] Department of Computer Science, University of Bucharest, Bucharest, Romania
`mihail-iulian.plesa@s.unibuc.ro`
[2] School of Electrical Engineering and Computer Science, University of Bradford, Bradford, UK
[3] School of Control Engineering, Chengdu University of Information Technology, Chengdu 610225, China

Abstract. Spiking Neural P systems, SN P systems for short, have found various applications over time. Perhaps the most important application to date is in the area of artificial intelligence where SN P systems are significant models of the third generation of neural networks. Another application of SN P systems that has not been researched much is cryptography. SN P systems can be used as computational devices on which various cryptographic algorithms can be implemented. Many of the machine learning algorithms that are applied in cryptography are based on neural networks which can be implemented using SN P systems. In this paper, we propose a new type of SN P system called Anti Spiking Neural Tree Parity Machine. The system is inspired by the way in which a Tree Parity Machine works and is constructed using SN P systems with anti-spikes. Based on the new system we propose a novel key agreement protocol that allows two parties to communicate over a public channel and obtain a secret shared key. We perform multiple experiments in which we show the efficiency of our protocol and its security.

Keywords: Spiking Neural P system · Anti-spikes · Tree Parity Machine · Cryptography · Key agreement protocol.

1 Introduction

Spiking neural P systems are a subclass of P systems that is inspired by how the biological neurons work and communicate [8]. Many types of SNP systems have been developed over time. For example, there are SNP systems with astrocytes [16], with communication on request [15], with polarizations [24], with colored spikes [20] or asynchronous systems [3]. In this work, we use a variant of the SN P system with anti-spikes [14]. Although we are using the original model of SN P with anti-spikes there are several other variants of this system such as systems without the annihilating priority [23] or systems with multiple channels [21].

The main application of an SN P system is in the construction of modern machine learning algorithms, especially in the design of specific classes of neural networks [4]. Although these types of neural networks are mostly used to solve classification problems, they can also be used for cryptographic purposes [19, 25]. For example, in [6] the authors proposed a new implementation of the RSA algorithm using SN P systems.

One special class of cryptographic algorithms is the one that performs the key agreement. This kind of algorithm allows two parties to communicate over a public channel and establish a shared secret key. Most key agreement algorithms are based on theoretical assumptions from number theory such as the hardness of integer factorization or the hardness of the discrete logarithm problem [7]. These problems can be solved in polynomial time using a quantum computer [18]. For this reason, researchers are investigating new constructions for key agreement algorithms. One such construction is the Tree Parity Machine (TPM) [11]. A TPM is a three-layer neural network and the protocol for key agreement is based on the phenomenon of synchronization of two neural networks [12].

1.1 Related work

There are many papers regarding key agreement protocols based on the synchronization of neural networks.

In [13] the authors proved that given the fact that the inputs are in the range $[-L, L]$ the synchronization time increases by L^2 while the probability of the attacker succeeding decreases exponentially by L. In [1] and [2] there are multiple strategies proposed for improving the security of the TPM-based key agreement protocol. The main idea is to obfuscate the outputs exchanged by the two legitimate parties so that an attacker cannot access them. The disadvantage of this approach is that it requires that the two parties have access to a common secret until the protocol is run.

In [9] the authors proposed a method for verifying the synchronization status of two TPMs. A public multivariable polynomial P is considered. The weights of the TPM represent the inputs for the polynomial P. Both parties compute and publish the value of the polynomial using the weights of their TPM. If the values are equal then the two TPMs are fully synchronized. The paper also shows that an attacker could not infer the weights based on the output of the polynomial using brute force or genetic searching algorithms.

In [5] the authors used complex numbers to represent the inputs, the weights, and the output of the TPM. The idea is to use the imaginary and real parts of a complex number to form the shared key. In this way, one run of the protocol will produce two keys. In [17] the authors tested more than 15 million combinations of TPM parameters to determine the most suitable ones to generate 512 bits RSA keys. For each combination, the authors analyzed the impact on the synchronization time and the security.

In [22] the authors generalize the original architecture by allowing the hidden layer to have an arbitrary number K of neurons and the input neurons to have non-binary values. By allowing non-binary input values, the time required for full synchronization decreases. The paper also studies the security of the protocol from the perspective of a Man-in-the-Middle attack. An attacker who intercepts the inputs and the outputs exchanged by the legitimate parties can synchronize a third TPM given enough information. Since the two TPMs have a low synchronization time, the chances of an attacker succeeding decrease. The paper shows a series of experiments that investigate the effects of the TMP parameters on the synchronization time and the security of the protocol. The authors also propose the use of entropy to measure the quality of the output key.

In [10] the authors proposed a new TPM architecture in which the inputs, the weights, and the outputs are vectors. In this way, a single run of the protocol will produce multiple keys. The paper studies the efficiency and the security of the proposed protocol. It also proposes some mathematical formulas for the synchronization time and the probability that the attacker will be successful depending on the parameters of the model.

1.2 Our contributions

In this paper, we propose a new type of SN P system with anti-spikes that is inspired by the construction of a TPM. We use this new system, to propose a new key agreement protocol that does not base its security on any number-theoretical problem. We conduct experiments that analyze various aspects of the protocol and compare the novel protocol with its classical counterpart based on TPMs.

The rest of the paper is organized as follows: Section 2 describes the new SN P systems with anti-spikes, called the Anti-Spiking-Neural-Tree-Parity-Machine P system, ASNTPM P system for short. Also in Section 2, we describe the proposed key agreement protocol. Section 3 presents the experimental results regarding the efficiency of the protocol, its security, and the cryptographic quality of the generated key. Section 4 is left for the conclusions and further directions of research.

2 A key agreement protocol based on ASNTPM P systems

This section introduces a new SN P system named Anti-Spinking Neural Tree Parity Machine (ASNTPM P system). The system is inspired by the construction of a TPM and is used to implement a new key agreement protocol. Experimental results show that the new protocol based on ANSTMP-systems substantially improves synchronization time.

2.1 ASNTPM P system

An ASNTPM P system can be viewed as a three-layer neural network. The input layer is partitioned into k groups each containing n neurons. Each group is connected to a neuron from the hidden layer and all neurons from this layer are connected to a single output neuron. The network can be visualized in Fig.1.

Definition 1. *An Anti-Spiking Neural Tree Parity Machine P system is defined as the following construct:*

$$\Pi = (O, \{\sigma_{input_1}, \sigma_{input_2}, ..., \sigma_{input_{kn}}\}, \{\sigma_{hidden_1}, \sigma_{hidden_2}, ..., \sigma_{hidden_k}\}, \sigma_{output}, N, K, L, syn_0, f)$$

where:

1. $O = \{a, \overline{a}\}$ is an alphabet formed by two symbols:
   (a) The symbol a denotes a spike
   (b) The symbol $\overline{a}$ denotes an anti-spike
2. $\sigma_{input_{ij}}$ is an input neuron formed by the folowing tupple $(n_{ij}, \overline{n}_{ij}, R_{ij})$:
   (a) n_{ij} denotes the number of spikes from the neuron
   (b) $\overline{n}_{ij}$ denotes the number of anti-spikes from the neuron
   (c) R_{ij} is a finite set rules of the following forms:
      i. Firing rules: $b^c \rightarrow b^{c*w_{ij}}$ where $b \in \{a, \overline{a}\}$, $1 \leq c \leq L$ and w_{ij} is a positive integer that will be defined below .
      If at the moment t a neuron has c spikes or anti-spikes it will fire, consuming either c spikes or c anti-spikes and sending c spikes or c anti-spikes to the hidden neuron $\sigma_{hidden_i}$with which it is connected. At moment $t = 0$, there are no spikes or anti-spikes in the neuron i.e. $n_{ij} = 0$, $\overline{n}_{ij} = 0$.
      ii. Annlihilation rule: $a\overline{a} \rightarrow \lambda$
      This rule indicates that at any moment t, an input neuron cannot contain spikes and anti-spikes simultaneously. If a spike and an anti-spike are present in an input neuron, they will annihilate each other instantaneously.

   Here, $1 \leq i \leq K$ and $1 \leq j \leq N$.
3. σ_{hidden_i} is a hidden neuron formed by the folowing tupple $(n_i, \overline{n}_i, R_i)$:
   (a) n_i denotes the number of spikes from the neuron
   (b) $\overline{n}_i$ denotes the number of anti-spikes from the neuron
   (c) R_i is a finite set of rules of the following form:
      i. Firing rules: rule of the form $b^c \rightarrow b; d$ where $b \in \{a, \overline{a}\}$, $1 \leq c \leq L^2$. If at the moment t the neuron has c spikes then it will fire consuming c spikes and sending 1 spike to the output neuron. If at the moment t the neuron has c anti-spikes then it will fire consuming c anti-spikes and sending 1 anti-spike to the output neuron. At moment $t = 0$, there are no spikes or anti-spikes in the neuron i.e. $n_i = 0$, $\overline{n}_i = 0$.
      ii. Annlihilation rule: $a\overline{a} \rightarrow \lambda$
      This rule indicates that at any moment t, a hidden neuron cannot contain spikes and anti-spikes simultaneously. If a spike and an anti-spike are present in a hidden neuron, they will annihilate each other instantaneously.

   Here, $1 \leq i \leq K$.
4. σ_{output} is the output neuron formed by the folowing tupple $(n_{out}, \overline{n}_{out}, r_{out})$:
   (a) n_{out} denotes the number of spikes from the neuron
   (b) $\overline{n}_{out}$ denotes an anti-spikes from the neuron

(c) r_{out} is an annihilation rule of the form $a\overline{a} \to \lambda$. The rule indicates that the output neuron cannot hold spikes and anti-spikes simultaneously. At moment $t = 0$, there are no spikes or anti-spikes in the neuron i.e. $n_{out} = 0$, $\overline{n}_{out} = 0$.

The output of the system is the number of spikes or anti-spikes from this neuron.

5. syn_t is the set of synapses at the computational step t. A synapse if defined by the triplet $(\sigma_i, \sigma_j, w_{ij})$ meaning the existence of a synapse between the neuron σ_i and the neuron σ_j. Here, σ_i and σ_j can be input, hidden or output neurons. The weight on the synapse, $w_{ij} \in \mathbb{Z}^+$ has the role to amplify the spikes or the anti-spikes passing through the synapse e.g. if the neuron σ_i fires sending c spikes or anti-spikes to the neuron σ_j and the weight on the synapse is w_{ij} then the neuron σ_j will receive $c * w_{ij}$ spikes or anti-spikes.
syn_0 represents the set of synapses at moment $t = 0$. Initially, the weights between the input and the hidden neurons are randomly chosen from the set $\{1, 2, ..., L\}$. The weights between the hidden and the output neurons are always 1.
6. N is the number of input neurons connected to a single hidden neuron.
7. K is the number of neurons hidden neurons.
8. L represents the maximum value of a weight i.e. $0 < w_{ij} \leq L$.
9. The learning function f has the role of updating the weights on the synapses according to (1):

$$syn_{t+1} = f(syn_t) \tag{1}$$

The input of the system is defined by the vector $X = (x_{11}, x_{12}, ..., x_{KN})$, $-L \leq x_{ij} \leq L, x_{ij} \neq 0$, $\forall 1 \leq i \leq K, 1 \leq j \leq N$. If $x_{ij} < 0$ then the input neuron σ_{ij} will receive from the environment $|x_{ij}|$ anti-spikes. If on the other hand, $x_{ij} > 0$ then the input neuron σ_{ij} will receive from the environment x_{ij} spikes.

The input neurons fire sending all the spikes or all the anti-spikes to the hidden neurons. The number of spikes or anti-spikes is amplified by the corresponding weight of the synapse. After the annihilation rule is applied the maximum number of times in each hidden neuron, they send one spike or one anti-spike to the output neuron. After the annihilation rule is applied the maximum number of times in the output neuron, it can be in one of the following states:

1. The output neuron is empty if the number of spikes received from the hidden neurons is equal to the number of anti-spikes.
2. The output neuron contains one spike if the number of spikes received from the hidden neurons is greater than the number of anti-spikes.
3. The output neuron contains one anti-spike if the number of spikes received from the hidden neurons is smaller than the number of anti-spikes.

The output of the system is the state of the output neuron. The system evolve by the application of the learning function f which modifies the weights of the synapses between the input and the hidden neurons.

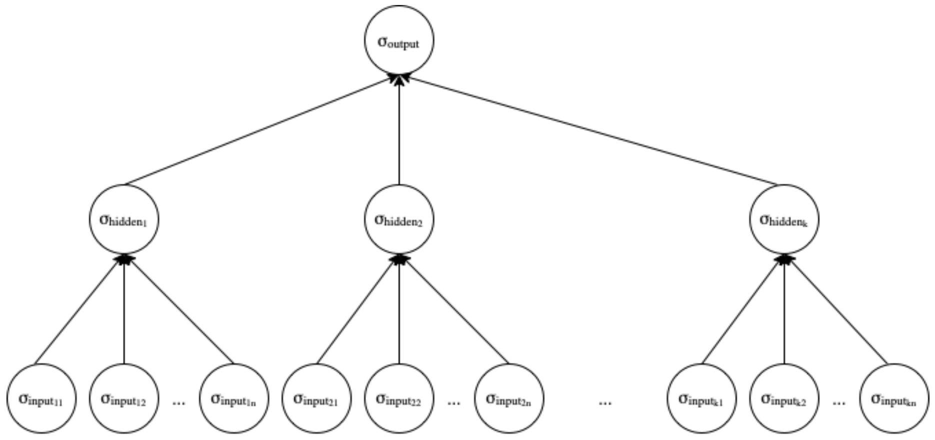

Fig. 1. A visual representation of an ASNTPM P system

2.2 The novel key agreement protocol

The goal of the protocol is for two parties, named Alice and Bob, to compute a secret shared key. All the communications take place over a public channel so that any message between Alice and Bob can be intercepted by a third malicious actor, named Eve. We suppose that Eve can read any message but it cannot alter it. The protocol is secure if Eve cannot compute the secret key between Alice and Bob.

Each of the two participants has a local ASNTPM P system to which only he has access. The main idea of the key agreement protocol is to synchronize the two P systems using the learning function. Two ASNTPM P systems are synchronized if all the synapses have the same weights. The shared secret key between Alice and Bob will be these weights. The protocol consists of the following steps:

1. Both participants agree over a public channel on the parameters N, K and L of the ASNTPM P system.
2. Both participants exchange over a public channel the input vector $X = (x_{11}, x_{12}, ..., x_{KN})$, $-L \leq x_{ij} \leq L, 1 \leq i \leq K, 1 \leq j \leq N$. Both ASNTPN P systems will receive input from the environment according to the vector X.
3. Each participant computes the output of their ASNTPM P system and publishes the result.
4. If the outputs are the same i.e. both Alice and Bob read the same number of spikes or the same number of anti-spikes in the output neuron of their ASNTPM P system, then each of them will update the weights of their system according to the learning function f.
5. The steps 2-4 are repeated until both ASNTPM P systems are synchronized.

Let $O_{Alice}(\sigma)$ and $O_{Bob}(\sigma)$ be the number of spikes contained in the σ neuron of Alice's respectively Bob's ASNTPM P system. Similarly, let $\overline{O}_{Alice}(\sigma)$ and

$\overline{O}_{Bob}(\sigma)$ be the number of anti-spikes contained in the σ neuron of Alice's respectively Bob's ASNTPM P system. The output of the two ASNTPM P systems are the same if the expression defined in (2) is true:

$$[[O_{Alice}(\sigma_{output}) = O_{Bob}(\sigma_{output})] \vee [\overline{O}_{Alice}(\sigma_{output}) = \overline{O}_{Bob}(\sigma_{output})]] \quad (2)$$

The learning function is described by the following algorithm, where $X \in \{Alice, Bob\}$:

```
for $i = 1;\ i \leq K;\ i = i + 1$ do
    if $[[O_X(\sigma_{hidden_i}) = O_X(\sigma_{output})] \vee [\overline{O}_X(\sigma_{hidden_i}) = \overline{O}_X(\sigma_{output})]]$
     then
        for $j = 1;\ j \leq N;\ j = j + 1$ do
            if $O_X(\sigma_{output}) > 0$ then
                $w_{ij} = |w_{ij} + O_X(\sigma_{input_{ij}})|$
            else
                if $\overline{O}_X(\sigma_{output}) > 0$ then
                    $w_{ij} = |w_{ij} - \overline{O}_X(\sigma_{input_{ij}})|$
                end
            end
        end
        if $w_{ij} > L$ then
            $w_{ij} = L$
        end
    end
end
```

Algorithm 1: The learning function

This learning function is an adaptation of the Hebbian and Anti-Hebbian learning rules used in classical TPM constructions.

3 Experimental results

In this section, we present the experimental results regarding the novel key agreement protocol. The experiments aim to evaluate the efficiency of the new protocol compared to the classical TPM based protocols, the security of the protocol against a Man-in-the-middle attack, and the cryptographic quality of the generated shared key.

3.1 The efficiency of the proposed protocol

The efficiency of the protocol is reflected by the synchronization time, i.e. the number of computation steps required by the two ASNTPM P systems to synchronize i.e. we count how many times the steps 2-4 from the protocol are executed. We tested multiple variations of the parameters N and K of the ASNTPM P systems but we maintained the L limit constant to 256. Since the inputs are

Table 1. The efficiency of the protocol

K	**N**	**Syncronization time**			
		Minimum	**Maximum**	**Mean**	**Standard deviation**
4	8	10	67	31.62	12.26
4	16	15	135	39.02	16.39
8	16	23	178	65.45	23.72
4	32	22	95	49.99	17.44
8	32	23	162	77.30	25.37
16	32	37	193	112.78	32.68
4	64	22	178	60.15	23.03
8	64	38	182	86.39	29.78
16	64	61	218	128.64	36.48
32	64	72	392	188.32	54.15

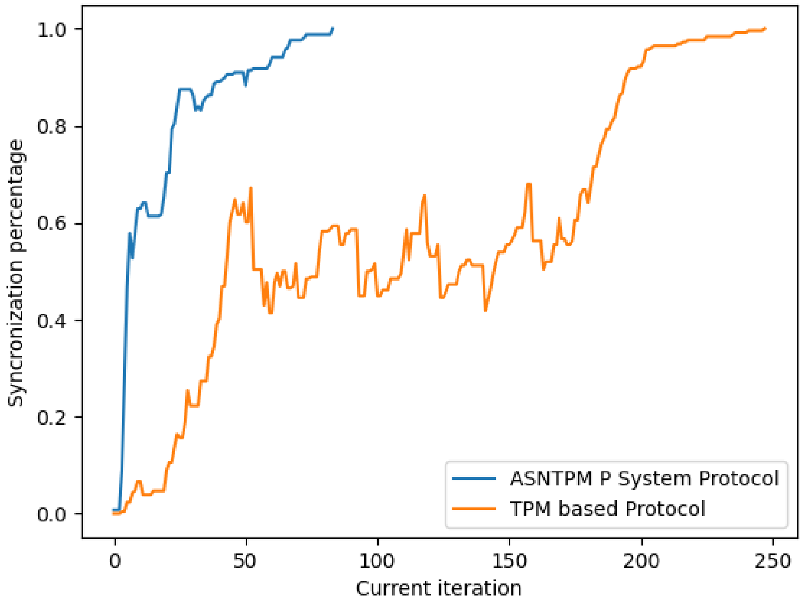

Fig. 2. Efficiency comparison between TPM and ASNTPM P system

randomly generated, we conducted 100 simulations for each set of parameters. The results are shown in Table 1.

Table 1 shows that by increasing the K parameter and holding N constant, there is an increase in the synchronization time. Similarly, increasing the N parameter and holding the K constant, increases the synchronization time.

A comparison between the novel protocol based on ASNTPM P system and the classical protocol based on TPM is presented in Table 2. The comparison is made concerning the mean and the standard deviation of the synchronization time.

Table 2. The efficiency of the protocol

K	**N**	**Syncronization time**			
		TPM		**ASNTPM P system**	
		Mean	**Standard deviation**	**Mean**	**Standard deviation**
4	8	138.49	64.23	31.62	12.26
4	16	135.16	63.81	39.02	16.39
8	16	247.18	80.78	65.45	23.72
4	32	138.77	52.06	49.99	17.44
8	32	278.16	89.87	77.30	25.37
16	32	450.63	90.65	112.78	32.68
4	64	149.59	53.45	60.15	23.03
8	64	319.74	102.59	86.39	29.78
16	64	519.39	122.25	128.64	36.48
32	64	727.02	130.50	188.32	54.15

The protocol based on ASNTPM P systems is much more efficient than the classical protocol based on TPM. Also, Table 2 shows that the standard deviation is lower in the case of the novel protocol. To better visualize the difference in efficiency between the two protocols, Figure 2 shows the synchronization percentage after each iteration computational step when $K = 8$ and $N = 32$.

3.2 The cryptographic quality of the key

The purpose of the protocol is for two participants to compute a secret shared key over a public channel. The key is then used in various cryptographic operations e.g. encryption, authentification, etc. The quality of the key is measured by its entropy. A low entropy level means that the key has little embedded randomness and therefore an attacker can deduce it. In the case of our protocol, the key is formed by the values of the weights which are positive numbers bounded by the parameter L of the ASNTPM P system. Thus we can view the system as a source I that generates symbols $k_1, k_2, ..., k_r$ with probabilities $p_1, p_2, ..., p_r$, $0 \leq k_i \leq L$, $0 \leq p_i \leq 1$, $\forall i, 1 \leq i \leq r$. The entropy of the source I measured in bits is defined in (3):

$$H(I) = -\sum_{i=1}^{r} p_i log_2 p_i \tag{3}$$

We make 100 simulations of the protocol for each configuration of the parameters K and N of the ASNTPM P systems. Table 3 shows the mean entropy of the generated keys for both the novel and the classical protocol.

Table 3. The mean entropy of the key

K	**N**	**Mean entropy of the key**	
		ASNTPM P system	**TPM**
		Mean	**Mean**
4	8	4.86	4.82
4	16	5.87	5.79
8	16	6.84	6.79
4	32	6.86	6.78
8	32	7.85	7.77
16	32	8.82	8.77
4	64	7.87	7.76
8	64	8.85	8.76
16	64	9.83	9.76
32	64	10.83	10.76

The two protocols produce keys with almost the same level of entropy although in our case, the entropy was slightly better. It can also be observed that increasing the parameters K and N also increases the entropy of the key. Figure 3 displays the distribution of the key elements when $K = 16$ and $N = 64$. A key used for cryptographic purposes must have a distribution as close as possible to the uniform distribution which is the case of the key produced by our protocol.

3.3 Security against Man-in-the-Middle Attack

Since all the information needed to synchronize the two ASNTPM P systems are transmitted over a public channel, an attacker named Eve can try to synchronize his system with the systems of the two legitimate participants Alice and Bob. The attack methodology is described by the following protocol:

1. Eve intercepts the parameters exchanged by Alice and Bob in step 1 of the protocol and instantiates its ASNTPM P system.
2. Eve intercepts the input vector X transmitted in step 2 of the protocol and initializes the input neurons of its ASNTPM P system in the same way as Alice and Bob.
3. Eve computes the output of its ASNTPM P system and intercepts the results announced by Alice and Bob.

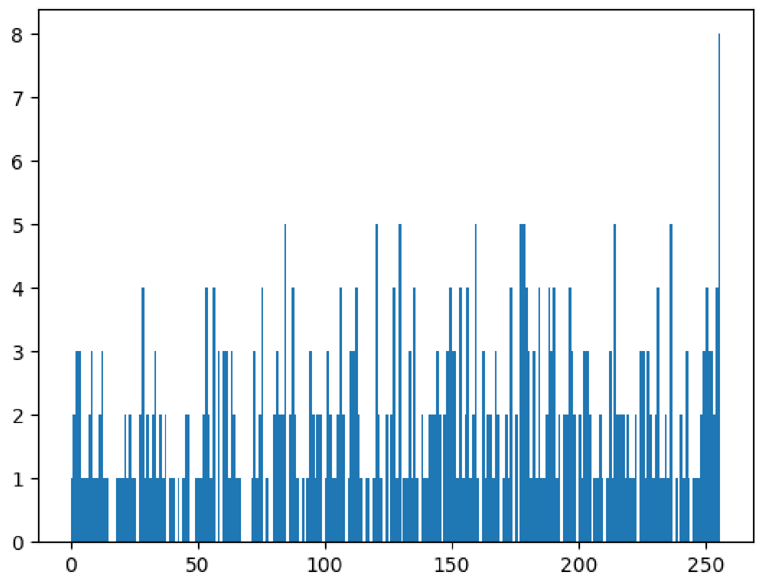

Fig. 3. The distribution of the key

4. If Eve's output is the same as the outputs produced by Alice and Bob then she will update the weights of his system according to the learning function f described in section 2.2.
5. The steps 2-4 are repeated until the ASNTPM P systems of Alice and Bob have been synchronized and they interrupt the exchange of information over the public channel.

The attacker has success if he manages to synchronize his ASNTPM P system with the systems of Alice and Bob simultaneously with them. The efficiency of the protocol directly affects its security. A protocol that performs fewer computational steps will not give the attacker a window of time large enough for him to synchronize simultaneously with the legimite participants. Figure 4 shows that at the moment when Alice and Bob's ASNTPM P systems are synchronized, Eve managed to synchronize his ASNTPM P system for only 0.4%. The experiment was performed with $K = 16$ and $N = 64$.

Table 4 shows a comparison between the mean synchronization percentage of Eve when the attack is mounted against our protocol and when the attack is mounted against the classical protocol. The results confirm that a lower synchronization time i.e. the case of our protocol implies a lower synchronization percentage of the attacker. For each set of parameters, 100 simulations were made.

Table 4. The syncronization percentage

K	**N**	**Mean syncronization percentage of the attacker**	
		ASNTPM P system	**TPM**
		Mean	**Mean**
4	8	0.05 %	0.68 %
4	16	0.02 %	0.40 %
8	16	0.01 %	0.45 %
4	32	0.01 %	0.53 %
8	32	0.02 %	0.40 %
16	32	0.02 %	0.30 %
4	64	0.02 %	0.34 %
8	64	0.02 %	0.30 %
16	64	0.03 %	0.34 %
32	64	0.03 %	0.32 %

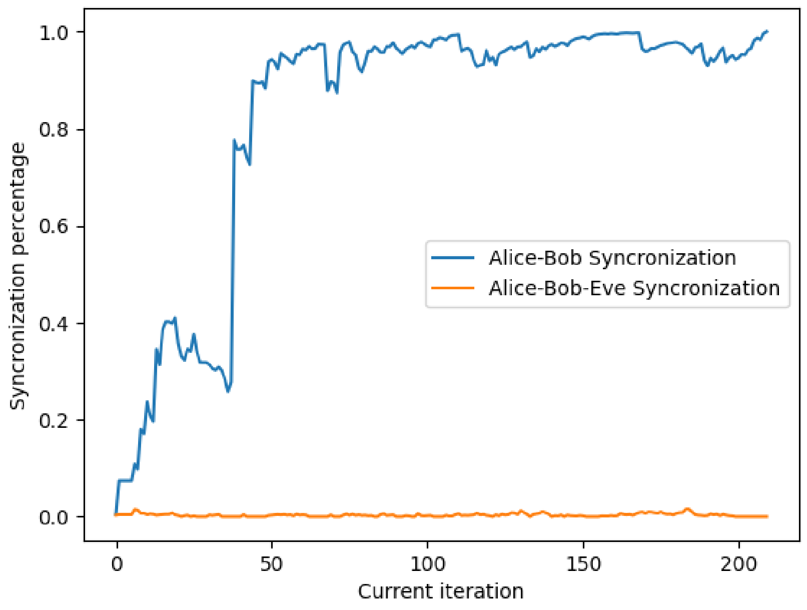

Fig. 4. The attack

4 Conclusions and further directions of research

This article presented a new P system called Anti Spiking Neural Tree Parity Machine (ASNTPM P system). Based on the new system, we proposed a novel key agreement protocol. We conducted multiple experiments that studied the efficiency of the new protocol, the cryptographic quality of the shared secret key, and the security against Man-in-the-Middle attacks. Our protocol is more efficient than the original key agreement protocol based on TPMs [11].

We have shown experimentally that security against Man-in-the-Middle-Attacks of the protocol is related to the synchronization time i.e. the number of steps executed by the protocol until the weights of both systems coincide. We measured the synchronization percentage of the attacker when the protocol finishes executing and showed that our new protocol is much more secure than the classical one.

A further direction of research is the extension of the experiments. Feature work might try several combinations of parameters for the ASNTPM and determine which one is the most efficient or the most secure. In this paper, we tried the most used parameters from the literature on TPM. In future work, perhaps a theoretical connection can be established between the parameters of the model, its efficiency, and its security.

Another direction of research is to study other connections between the field of cryptography and SN P systems. Given the fact the SN P systems are Turing complete, they can be used to implement various cryptographic primitives such as hash functions, symmetric and asymmetric ciphers, MACs, or cryptographic primitives based on elliptic curves.

References

1. Allam, A.M., Abbas, H.M.: Improved security of neural cryptography using don't-trust-my-partner and error prediction. In: 2009 International Joint Conference on Neural Networks. pp. 121–127. IEEE (2009)
2. Allam, A.M., Abbas, H.M.: On the improvement of neural cryptography using erroneous transmitted information with error prediction. IEEE transactions on neural networks **21**(12), 1915–1924 (2010)
3. Cavaliere, M., Ibarra, O.H., Păun, G., Egecioglu, O., Ionescu, M., Woodworth, S.: Asynchronous spiking neural P systems. Theoretical Computer Science **410**(24-25), 2352–2364 (2009)
4. Chen, Y., Chen, Y., Zhang, G., Paul, P., Wu, T., Zhang, X., Rong, H., Ma, X.: A survey of learning spiking neural P systems and a novel instance. International Journal of Unconventional Computing **16** (2021)
5. Dong, T., Huang, T.: Neural cryptography based on complex-valued neural network. IEEE transactions on neural networks and learning systems **31**(11), 4999–5004 (2019)
6. Ganbaatar, G., Nyamdorj, D., Cichon, G., Ishdorj, T.O.: Implementation of RSA cryptographic algorithm using SN P systems based on HP/LP neurons. Journal of Membrane Computing **3**(1), 22–34 (2021)
7. Hoffstein, J., Pipher, J., Silverman, J.H., Silverman, J.H.: An introduction to mathematical cryptography, vol. 1. Springer (2008)
8. Ionescu, M., Păun, G., Yokomori, T.: Spiking neural P systems. Fundamenta informaticae **71**(2-3), 279–308 (2006)
9. Javurek, M., Turčaník, M.: Synchronization of two tree parity machines. In: 2016 New Trends in Signal Processing (NTSP). pp. 1–4. IEEE (2016)
10. Jeong, S., Park, C., Hong, D., Seo, C., Jho, N.: Neural cryptography based on generalized tree parity machine for real-life systems. Security and Communication Networks **2021** (2021)
11. Kanter, I., Kinzel, W., Kanter, E.: Secure exchange of information by synchronization of neural networks. EPL (Europhysics Letters) **57**(1), 141 (2002)
12. Klein, E., Mislovaty, R., Kanter, I., Ruttor, A., Kinzel, W.: Synchronization of neural networks by mutual learning and its application to cryptography. Advances in Neural Information Processing Systems **17** (2004)
13. Mislovaty, R., Perchenok, Y., Kanter, I., Kinzel, W.: Secure key-exchange protocol with an absence of injective functions. Physical Review E **66**(6), 066102 (2002)
14. Pan, L., Păun, G.: Spiking neural P systems with anti-spikes. International Journal of Computers Communications & Control **4**(3), 273–282 (2009)
15. Pan, L., Păun, G., Zhang, G., Neri, F.: Spiking neural P systems with communication on request. International journal of neural systems **27**(08), 1750042 (2017)
16. Pan, L., Wang, J., Hoogeboom, H.J.: Spiking neural P systems with astrocytes. Neural Computation **24**(3), 805–825 (2012)
17. Salguero Dorokhin, É., Fuertes, W., Lascano, E.: On the development of an optimal structure of tree parity machine for the establishment of a cryptographic key. Security and Communication Networks **2019** (2019)
18. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM review **41**(2), 303–332 (1999)
19. Song, T., Pan, L., Wu, T., Zheng, P., Wong, M.D., Rodríguez-Patón, A.: Spiking neural P systems with learning functions. IEEE transactions on nanobioscience **18**(2), 176–190 (2019)

20. Song, T., Rodríguez-Patón, A., Zheng, P., Zeng, X.: Spiking neural P systems with colored spikes. IEEE Transactions on Cognitive and Developmental Systems **10**(4), 1106–1115 (2017)
21. Song, X., Wang, J., Peng, H., Ning, G., Sun, Z., Wang, T., Yang, F.: Spiking neural P systems with multiple channels and anti-spikes. Biosystems **169**, 13–19 (2018)
22. Stypiński, M., Niemiec, M.: Synchronization of Tree Parity Machines using non-binary input vectors. arXiv preprint arXiv:2104.11105 (2021)
23. Wang, X., Song, T., Zheng, P., Hao, S., Ma, T.: Spiking neural P systems with anti-spikes and without annihilating priority. Rammian Journal of Science and Technology **20**(1), 32–41 (2017)
24. Wu, T., Păun, A., Zhang, Z., Pan, L.: Spiking neural P systems with polarizations. IEEE transactions on neural networks and learning systems **29**(8), 3349–3360 (2017)
25. Zhang, G., Zhang, X., Rong, H., Paul, P., Zhu, M., Neri, F., Ong, Y.S.: A layered spiking neural system for classification problems. International Journal of Neural Systems p. 2250023 (2022)