UDC 519.7

O.M. Glazok, c.t.s.
(National Aviation University, Ukraine)

The use of the extended shear register PRNG in a neural session key exchange protocol

A neural network based cryptographic protocol for session key exchange and the corresponding architecture of the system is proposed. The system includes a pseudorandom number generator based on the scheme of extended shear register built on the nonbinary elements and a tree parity machine.

Applications of neural networks in cryptography include encryption and decryption, message digests and digital signatures generation, cryptanalysis attacks, as well as key exchange [1,2]. Key exchange is one of the basic problems of modern cryptography. In today's information environment, it is a common problem to establish via open communication channels a secure virtual communication channel between two parties who have never met before and do not have a reliable secret channel for transmitting classified information during key negotiation. In order to solve this problem, the parties must agree on session keys which will be known only to legitimate participants in the exchange, using only an insecure communication channel. By default, one can assume that information transmitted over public channels is available for listening to an attacker who will try to obtain matching keys. The Diffie-Hellman key exchange protocol is now widely used to solve this problem. Neural network key exchange protocols are considered as a possible safe replacement for the Diffie-Hellman protocol [3].

The currently developed neural key exchange protocols are mostly based on the synchronization of two tree parity machines. The tree parity machine is a special type of multi-layer feedforward neural network. A tree parity machine with architecture parameters – the numbers K and N – consists of $K*N$ input neurons, K hidden neurons and one output neuron (Fig. 1).

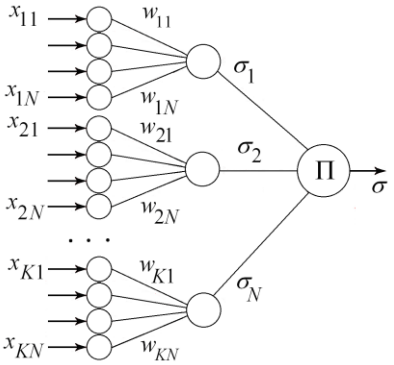

Fig. 1. The basic structure of a tree parity machine.

Inputs to the network (denoted as x_{ij} in the Fig.1) may take one of the three possible values: –1, 0 or 1. The output value of each hidden (i-th) neuron is calculated as a function of its inputs:

$$\sigma_i = sign\left(\sum_{j=1}^{N} w_{ij} x_{ij}\right), \quad (1)$$

where x_{ij} are input values of the j-th neuron; w_{ij} are the weights of connections between input and hidden neurons. These weights are taken as integer coefficients which may take positive, negative or zero values, but bounded with a constant L common for the entire network: $\forall i, j : |w_{ij}| \le L$.

Finally, the output of neural network is computed as the multiplication (product) of all values produced by hidden elements with a correction:

$$\sigma = \prod_{i=1}^{K} c(\sigma_i); \quad (2)$$

$$c(\sigma_i) = \begin{cases} -1, \sigma_i = -1 \\ -1, \sigma_i = 0 \\ +1, \sigma_i = +1 \end{cases} \quad (3)$$

The outputs σ_i of the hidden neurons, being the values of the signum function, may take three possible values: –1, 0 or 1 In order to avoid the loss of value in the product (2), the outputs of hidden neurons (1) are corrected according to (3); namely, if for some i the sum (1) turns out to be 0, it is changed to –1. Due to this, the output (2) of the tree parity machine is either +1 or –1, but never becomes 0. A binarized variant of tree parity machines, called “permutation parity machines”, is also used. In a permutation parity machine, the weights between input and hidden neurons are binary values, e.g. 0 or 1. The output values of the hidden neurons are also binary.

The goal of the proposed protocol is generation of a secret session key common for the two parties A and B who communicate over an insecure channel in such a way that the malefactor who listens their negotiations was not able to retrieve the result (the session key). In order to implement the protocol the two parties should keep the shared secret (a master key) which is used in calculations but is not transmitted over the communication channel. An important element of the scheme is the pseudorandom number generator (PRNG) which produces a group of values for each round of the protocol. It is proposed to use in this scheme a PRNG based on the scheme of extended shear register. The classical shear register is constructed of a series of a simple elements such as D-triggers that may keep one of the two binary states (0 or 1). The register feedback is implemented as a set of taps which take values from some elements of the register. These values are added modulo 2 which gives the feedback value. The extended shear register is built of the elements that may take a set of states. Because the states are not binary now, simple addition modulo 2 is replaced with the function whose inputs and the output are elements of the set of states. Such a function may be one and the same for all the register, repeating several times, or one can use a set of different functions. Thus, the feedback value of the extended shear

register depends on the positions of taps and the set of mapping functions used. A classical shear register turns out to be a partial case of an extended one. The structure of an extended shear register is shown in the Fig.2.

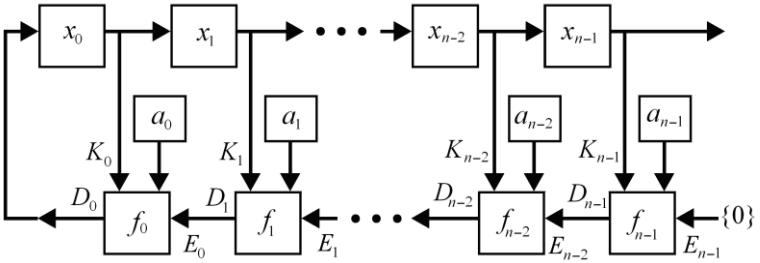

Fig. 2. The structure of an extended shear register.

In the Figure 3 there is an example of the results of research of dependence of the maximal periods of the output series of the extended shear register on its length.

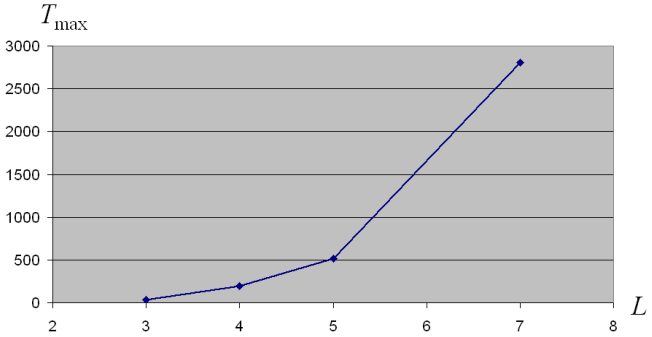

Fig. 3. The dependence of maximal periods of the output number series ($T_{\max}$) of the extended shear registers with eight possible states of a cell on the length of the register L .

The configuration and initialization data completely determine the PRNG output; these data include the part of a shared secret S_1 and the nonce values N_{A1} and N_{B1} which are generated correspondingly by A and B at the beginning of the operation, and, if the parties agree on that, may be substituted by the new nonce values at any round. The input values of the tree parity machine at each round of the protocol operation are formed in the input block which performs conversion of the four groups of parameters: 1) the current values from the output of the pseudorandom number generator (PRNG); 2) the nonce values N_{A2} and N_{B2} provided by the parties of the communication and refreshed for each step of operation; 3) the part of a shared secret S_2 ; 4) the secret (private) nonces P_A and P_B that are generated by each party at the

beginning of the process but not transferred to the other party. In addition, another group of PRNG output values is used for changing the network structure. In the simplest variant, the weights of the network may be determined at each round as $w_{ij} = v_{ij}^{curr} b_{ij}$, where v_{ij}^{curr} are the current values of the network coefficients sought during the network learning, b_{ij} are binary values (0 or 1) produced by the PRNG.

Implementation of the protocol starts with choosing by each party its corresponding part of the coefficients set $\left\{v_{ij}^0\right\}$ and generation of the private nonce values – P_A by the party A and P_B by the party B. Then the parties produce the public nonce values $N_{A1}, N_{A2}, N_{B1}, N_{B2}$. The PRNG is configured and initialized. Then the inputs x_{ij} for the tree parity machine are determined. For the beginning of operation, each party chooses random values for those weight coefficients that remain unknown. Based on the known inputs and the weight coefficients of the network, the output value σ of the tree parity machine is determined. Now the parties may perform a step of mutual learning for their neural networks according to one of known procedures, with the goal of finding the common set of the coefficients v_{ij}^1 and the unknown private nonce of the other party, so that A tries to find the unknown half of values from $\left\{v_{ij}^0\right\}$ and P_B, B tries to find the unknown half of values from $\left\{v_{ij}^0\right\}$ and P_A. The malefactor who may eavesdrop their communication has to solve the problem of finding the complete set of values $\left\{v_{ij}\right\}$, P_A and P_B. Solving this problem requires having more information than solving the two above mentioned partial problems, therefore the key exchange protocol achieves its goal and finishes before the malefactor succeeds.

Conclusions. Experiments show that the proposed protocol performs its main function successfully. The possible directions of the further research include: the investigation of choice of the mapping function in the extended shear register; the investigation of possibilities of coordination of the register structure with the structure and parameters of the neural network; scaling the protocol for the number of parties greater than 2.

References

1. Солодовников В., Евдокимов А. Анализ криптостойкости нейросетевого алгоритма симметричного шифрования // Новые информационные технологии в автоматизированных системах. – 2016. – No. 19. – С. 263-269.
2. Kinzel W., Kanter, I. Neural Cryptography // Proc. of the 9th int. conf. on neural information processing (ICONIP'02) . – 2002. – Vol. 3. – Pp. 1351–1354.
3. Singh, A., Nandal, A. Neural Cryptography for Secret Key Exchange and Encryption with AES // Int. J. of Advanced Research in Computer Science and Software Engineering. – 2013. – Vol. 3, Issue 5. – Pp. 376–381.