ACCESS CONTROL SYSTEM WITH INTEGRATED PROCESS OF TWO-FACTOR AUTHENTICATION BASED ON NEURO-FUZZY NETWORKS

Ekaterina Lobaneva[a], Alexey Lazarev[a]

[a] *Branch of the National Research University Moscow Power Engineering Institute in Smolensk, 1 Energeticheskiy proyezd, Smolensk, 214013, Russia*

Abstract

The paper presents the developed access control system, which allows to identify, control and manage clients by applying two-factor authentication algorithms. The access control system is based on the use of artificial neural fuzzy networks of deep training in combination with the algorithms of two-stage verification.

Keywords 1

Access Control System, two-factor authentication, neuro-fuzzy algorithms, fuzzy model, ARM processor

1. Introduction

In the modern world, the factor of ensuring safety in various spheres of human life takes a priority place. Almost any organization must first of all ensure the availability of a security system at the target object, and the security segment must correspond to the proper level and provide an opportunity for convenient management of the organization and employees in the optimal mode [1, 2]. The main idea described and implemented by the authors allows to solve the main problem of ensuring security at critically important objects by means of the implemented hardware and software access control system [3].

The conditions for ensuring increased security at the facility allow to initialize the priority disadvantages inherent in systems in the studied market: the use of standard authentication algorithms in the systems makes it vulnerable to a number of attacks, and the integration of biometric systems is often ineffective in small and medium-sized businesses.

At the planning stage of the implemented access control system, the main criteria were identified that the hardware and software components of the system must meet. The ARM processor is used as a hardware component due to its cost-effective use in small organizations and the possibility of integrating its own software algorithms. The software component is based on the use of a multi-level system for generating one-time codes through the use of neuro-fuzzy algorithms based on static parameters of the operating system.

2. Technical description of the problem

Based on the task described above, the main software and hardware requirements can be identified. First, the access control system should provide the possibility to check the validity of the generated sequence without active Internet connection. The existing technologies use a dynamic sequence generation algorithm based on analyzed data tied to the current time parameter and the Unix static time parameter, however, the existing algorithm is not sufficiently secure, since the integration of a

Russian Advances in Fuzzy Systems and Soft Computing: selected contributions to the 8-th International Conference on Fuzzy Systems, Soft Computing and Intelligent Technologies (FSSCIT-2020), June 29 – July 1, 2020, Smolensk, Russia

EMAIL: lobaneva94@mail.ru (A. 1); anonymous.prodject@gmail.com (A. 2);
ORCID: 0000-0002-0695-5145 (A. 1); 0000-0003-3252-0409 (A. 2)

neural network, represented by a multilayer perceptron and a recurrent neural network, makes it possible to create a unique generation algorithm sequences at regular intervals without the possibility of cloning by third parties. Secondly, the algorithm must ensure the transfer of a unique client identifier, for this purpose the integration of the software component with the PostgreSQL relational database is used [4, 5]. The process of transferring a unique identifier is implemented by generating a character string based on static parameters of the hardware component of the server control unit. Thirdly, the server part of the system must provide transmission of a numerical sequence in encrypted form, using the AES encryption algorithm [6]. The required parameter represented by the private key in this case is dynamic and is generated by a recurrent neural network within a certain time value range.

The software component of the system is based on the Python programming language due to its multiplatform support and the absence of the need to consume significant software and hardware resources, which makes it possible to adapt the operation of the software module to the mobile ARM processor [7]. The developed software platform provides the following functionality:

- generation of dynamic sequences based on time range;
- organization of software component management using a web interface located on the server side of the hardware system;
- validation of the sequence on the server without an active Internet connection;
- ability to parallelize the sequence generation process on several devices to ensure correct interaction.

3. Proposed solution

To solve the above problem, an access control system based on the ARM processor was developed, with the software component divided into several modules. Separateness in the use of multi-level system is explained by the factor of increasing the level of security by crossing the output data of the layers in the process of the algorithm. Thus, at the modeling stage, the following modules were identified that ensure the correct operation of the two-factor authentication system:

- module “building the sequence input parameters” – the initial part of obtaining input values of the multilayer perceptron;
- module “generation of a numerical sequence based on multilevel perceptrons” – an algorithm for obtaining a new sequence based on the input parameters of the previous stage of value generation is applied;
- module “sequence encryption based on AES algorithm” – encryption of the sequence obtained on the previous layer is performed by generating a key based on a recurrent neural network;
- module “expert system for sequence assessment” – an analytical comparison of the output parameters of the client's neural network, represented by the hash function, with the provided hash function from the server side is performed.

The modules developed in the software component of the project are presented in Figure 1.

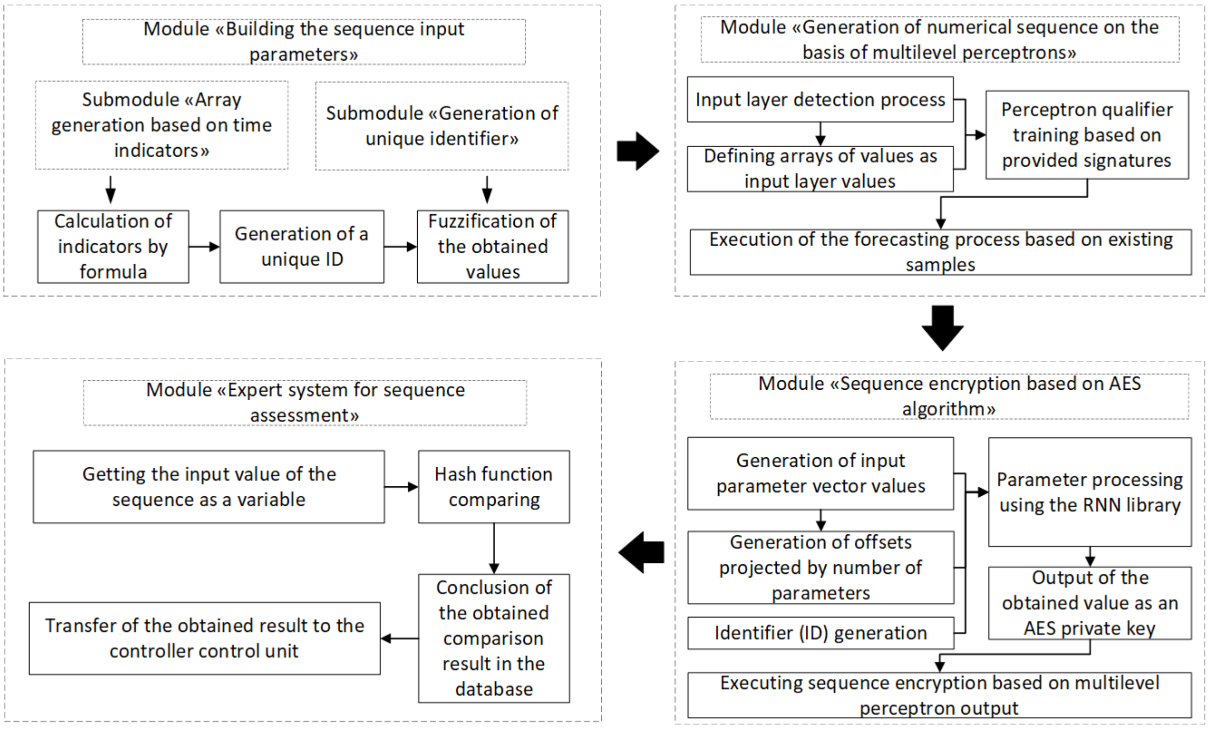

Module «Building the sequence input parameters»
Submodule «Array generation based on time indicators»
Submodule «Generation of unique identifier»
Calculation of indicators by formula
Generation of a unique ID
Fuzzification of the obtained values
Module «Generation of numerical sequence on the basis of multilevel perceptrons»
Input layer detection process
Defining arrays of values as input layer values
Perceptron qualifier training based on provided signatures
Execution of the forecasting process based on existing samples
Module «Expert system for sequence assessment»
Getting the input value of the sequence as a variable
Hash function comparing
Conclusion of the obtained comparison result in the database
Transfer of the obtained result to the controller control unit
Module «Sequence encryption based on AES algorithm»
Generation of input parameter vector values
Generation of offsets projected by number of parameters
Identifier (ID) generation
Parameter processing using the RNN library
Output of the obtained value as an AES private key
Executing sequence encryption based on multilevel perceptron output

Figure 1: Scheme of modules of the developed system

3.1. Number sequence generation and post-processing

The process of two-factor device authentication is generally based on the application of static and dynamic time parameters in Microsoft Windows, Linux operating systems, as well as all systems based on the Linux kernel. The algorithm for calculating a static index serving as a key with a limited time action, in general, is calculated based on the static value of the starting point of time by UTC and the current value of time in the system [8, 9]. In addition to the temporary value, the calculation of the unique identifier of the hardware component of the system is added. The parameter of the unique identifier in this case is static and is bound to the hardware value of the ARM processor component, and the variable obtained in the process of generating the device identifier is modified on the basis of the basic parameters of the string array sampling. Based on a high value of the system security index, the generation of a shared key is carried out at the expense of the current activity of the Linux kernel: the concept of using identical hardware components in client and server devices allows us to conclude that the hardware calculation index will be the same [10]. The main indicator is also the RAM ID, which allows you to use a unique static variable for calculating the time string of characters in the future. Thus, the obtained formula for generating a time value based on time parameters is presented below (1).

$$TFA(ID\_PROC\ ) = \left(\frac{\left[\dfrac{T_1 - T_0}{T} \right]}{SHA_1 \left[R \left(ID\_\ PROC \begin{pmatrix} CPU_{IMPLEMENTER} \\ SCSI_{PRODUCT} \\ SCSI_{PHYSICAL_{ID}} \end{pmatrix} \right) \right]} \right) \cdot \frac{\log(E^{TIMEOUT})}{KERNEL\_\ ID}, \qquad (1)$$

where TFA – is a unique indicator valid at the time of execution of the function during the time interval TIMEOUT; ID_PROC – static indicator of the function of the hardware component of the device; T_1 – current time value; T_0 – static indicator of the origin of time in UTC; TIMEOUT – action parameter of the current indicator of the generated function; SHA_1 – function of generating one-dimensional crypto-protected array; R – function for refining a sample of static system parameters; $\text{CPU}_{\text{IMPLEMENTER}}$ – static CPU offset; $\text{SCSI}_{\text{PRODUCT}}$ – processor static character string; $\text{SCSI}_{\text{PHUSYCAL\_ID}}$ – static CPU identifier; KERNEL_ID – the numeric value of the version of the Linux kernel; MEMORY_ID – internal memory identifier.

The peculiarity of the integration of the multilayer perceptron (MLP) is expressed in the possibility of specifying functional layers as input parameters of the neural network [11, 12]. The value obtained as a result of executing the model of the software component based on the above formula is interpreted by the fuzzifier as the object of the first input node in the process of generating a sequence based on a multilayer perceptron [13, 14]. The second layer uses the values of the indicators of the hardware support of the device. The final layer is used to specify a specific parameter, called the internal memory offset of the hardware device.

The classifier training process is the submission of training samples. Due to the fact that one-dimensional arrays represented by values in the interval «[0,1]» are fed to the input as training signatures, the value of the target sequence obtained in the previous step is converted into an interval binary array with a breakdown by the sequence restriction of 8 characters. The use of such an approach allows one to perform partial training of the neural network, and then, at the output, obtain the predicted value used in subsequent modules [15]. In other words, as a result of network training, the output parameter can be a sequence of characters represented by a one-dimensional array of required length (Figure 2).

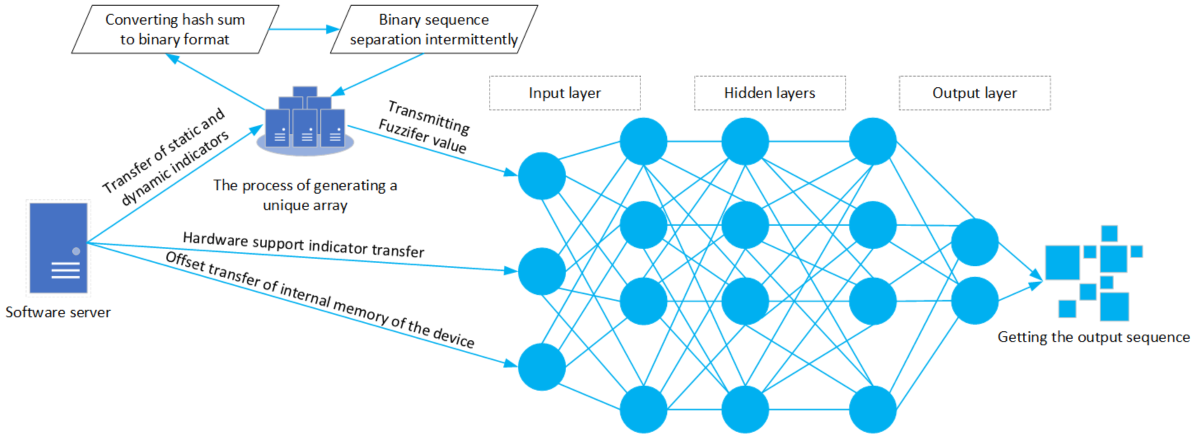
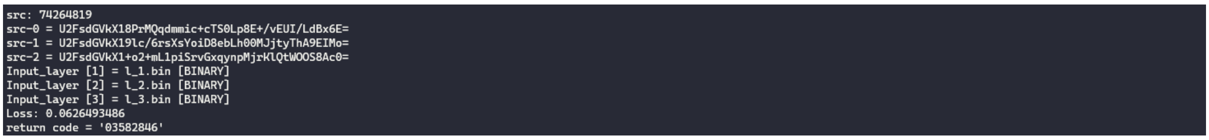

Converting hash sum to binary format
Binary sequence separation intermittently
Transmitting Fuzzifer value
Transfer of static and dynamic indicators
The process of generating a unique array
Input layer
Hidden layers
Output layer
Hardware support indicator transfer
Offset transfer of internal memory of the device
Software server
Getting the output sequence

Figure 2: The process of predicting a value by a multilayer perceptron

As a result of processing the digital sequence, a limited string array presented in Figure 3 will be fed to the input of the next module, shown in Figure 3. The subprocess of introducing restrictions is necessary, since as a result of encryption of the final sequence, the array transmitted from the client device to the server device should not exceed 64 bytes.



```
src: 74264819
src-0 = U2FsdGVkX18PrMQqdmmic+cTS0Lp8E+/vEUI/LdBx6E=
src-1 = U2FsdGVkX19lc/6rsXsYoiD8ebLh00MJjtyThA9EIMo=
src-2 = U2FsdGVkX1+o2+mL1piSrvGxqynpMjrKlQtWOOS8Ac0=
Input_layer [1] = l_1.bin [BINARY]
Input_layer [2] = l_2.bin [BINARY]
Input_layer [3] = l_3.bin [BINARY]
Loss: 0.0626493486
return code = '03582846'
```

Figure 3: Multilayer Perceptron Processing Testing

3.2. Sequence encryption based on AES algorithm

The process of encrypting the output parameter of a multilayer perceptron is performed on the basis of the symmetric AES algorithm, and as a private key - generation of a temporary key value based on recurrent neural networks [16]. Prevention of unauthorized access attempts is carried out on the basis of the fact that several vector values are fed to the recurrent neural network and a new value is obtained at the output that takes the value of the private key in AES encryption [17]. Selection of the use of a recurrent neural network is expressed in the possibility of setting variable intervals of input parameters of the network, while the type of the used network is «many to one» [18].

The input parameter, which is the private key in the AES cryptographic method, is a string parameter, which is 256 bits, and the dimension of the key does not essentially determine the size of the output value. The main vector parameters in this case are the parameter of the string output obtained at the multilayer perceptron output and the vector of dynamic value calculation based on the time index from Formula 1. Based on the above factors, the formula for a hidden layer of a recurrent network represented by a vector is reflected in (2). So, the calculated formula of output value RNN of a network will have an alternative type (3).

$$z_i = \tan z(G_{iz}i_t + G_{zz}z_{t-1} + l_z), \tag{2}$$

were z – recurrent network hidden layer; i – the input parameters described above; t – recurrent function state stage; l – offset parameter; G – weight parameter used in the calculations.

$$q_t = G_{zq}z_t + l_a, \tag{3}$$

were q – network RNN output parameter; G – weight parameter used in the calculations; z – hidden layer of recurrent function; t – recurrent function state stage; l – offset parameter.

Software implementation is possible with the help of several software solutions - the extensive set of functionalities provides the software solution TensorFlow with the joint use of Keras neural network library, which allows to automate the process of machine learning in order to obtain reliable output results, but in this implementation the software component is based on the PyTorch library due to hardware limitations on the part of devices based on ARM processors [19 - 20]. Application of the library is based on the use of the input values of the numeric sequence as the values of the tokenizer, in this case the numeric sequence will be the output value of the previous module, and as an additional parameter dynamic parameter for calculating the time value [21]. Operation of the processor unit of neural network training also requires a static offset parameter, which is a floating-point value; it is suggested to use the entropy mean as the main input parameters of the neural network. The actual value of the argument in this case depends on the complexity of the input sequence being submitted, and, accordingly, taking into account the input index of 8 characters long, the average value of the complexity with a higher digit deviation is used, because in the presence of a dynamic factor in the generated value it is possible to obtain 8^{10} combinations. So, as a result of training the neural network, a unique value will be supplied to the output, which in this case is called the AES private key (Figure 4).

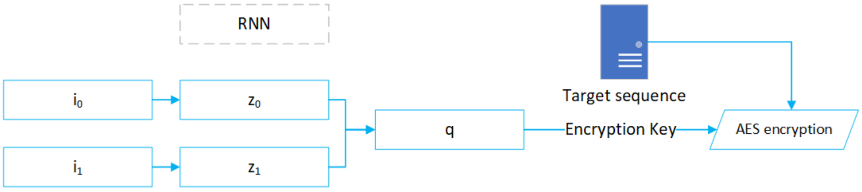


Figure 4: The process of AES key generation by a recurrent neural network

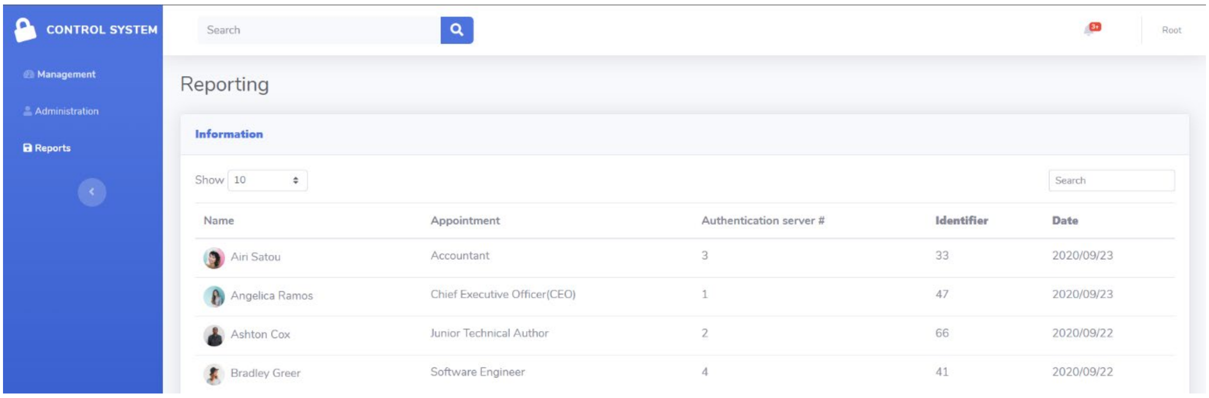
3.3. Expert system of analytical sequence comparison

The basic concept of the operation of access control systems is based on the fact of checking a unique fingerprint on the server side, in general, such systems check a static key written in the corresponding section of the information storage device. In the case of the implementation of an intelligent access control system based on the use of neural networks, the generated fingerprint is valid for a certain period of time. Also, the factor of identifying a unique client device is superimposed on the factor of generating a dynamic key, in this regard, the fact of writing a fingerprint into the database is not possible.

The solution to the above problem is the ability to use a relational database with the corresponding data tables, on the basis of which the fingerprints will be compared. The encryption process described in subsection 3.2 was designed taking into account the factor of the impossibility of reverse decryption of the original sequences - using the reverse decryption process will violate the security of the protocol, therefore, the fingerprint is compared using the hash function transmitted from the AES encryption process. Due to the use of a dynamic key generation system, the process of storing the hash function in the database is not possible – as a result of processing the initial sequence, it is impossible to calculate the final value.

It is possible to eliminate the problem of analytical comparison of data on a client device and a server by parallelizing the processes of generating a numerical sequence, including processing sub-processes using a multilayer perceptron and further encryption using the AES algorithm based on recurrent neural networks. The developed hardware and software solution allow you to start the key generation process both on the client device and on the server. It is proposed to use the PostgreSQL server as the database system used, since the use of the MySQL database server is not possible due to the limited hardware resources of devices based on the ARM processor [5, 22].

The set of databases uses several tables – «Web-admin», which initializes users who manage the web-based administration interface; «Users» – table of unique identifiers of user keys; «Static» – a table for saving logs of user authorization attempts, as well as possible unsuccessful authentication attempts [23]. This set of tables will allow controlling the process of client authentication by the administrator of the access control system – viewing the logs of authentication information is provided by the web interface shown in Figure 5 [24].



Reporting

Information

Name	Appointment	Authentication server #	Identifier	Date
Airi Satou	Accountant	3	33	2020/09/23
Angelica Ramos	Chief Executive Officer(CEO)	1	47	2020/09/23
Ashton Cox	Junior Technical Author	2	66	2020/09/22
Bradley Greer	Software Engineer	4	41	2020/09/22

Figure 5: Web interface for viewing authentication reports

The algorithm shown in Figure 6 demonstrates the general software process of interaction between the client and server hardware of the device. The sequence transmission process is carried out on a simplified prototype of the signature processing modules at frequencies of 433/315 MHz. Due to the unprofitable consumption of energy resources of the signal processing modules existing on the market, the hardware component of the devices interacts on integral components in conjunction with additional discrete components.

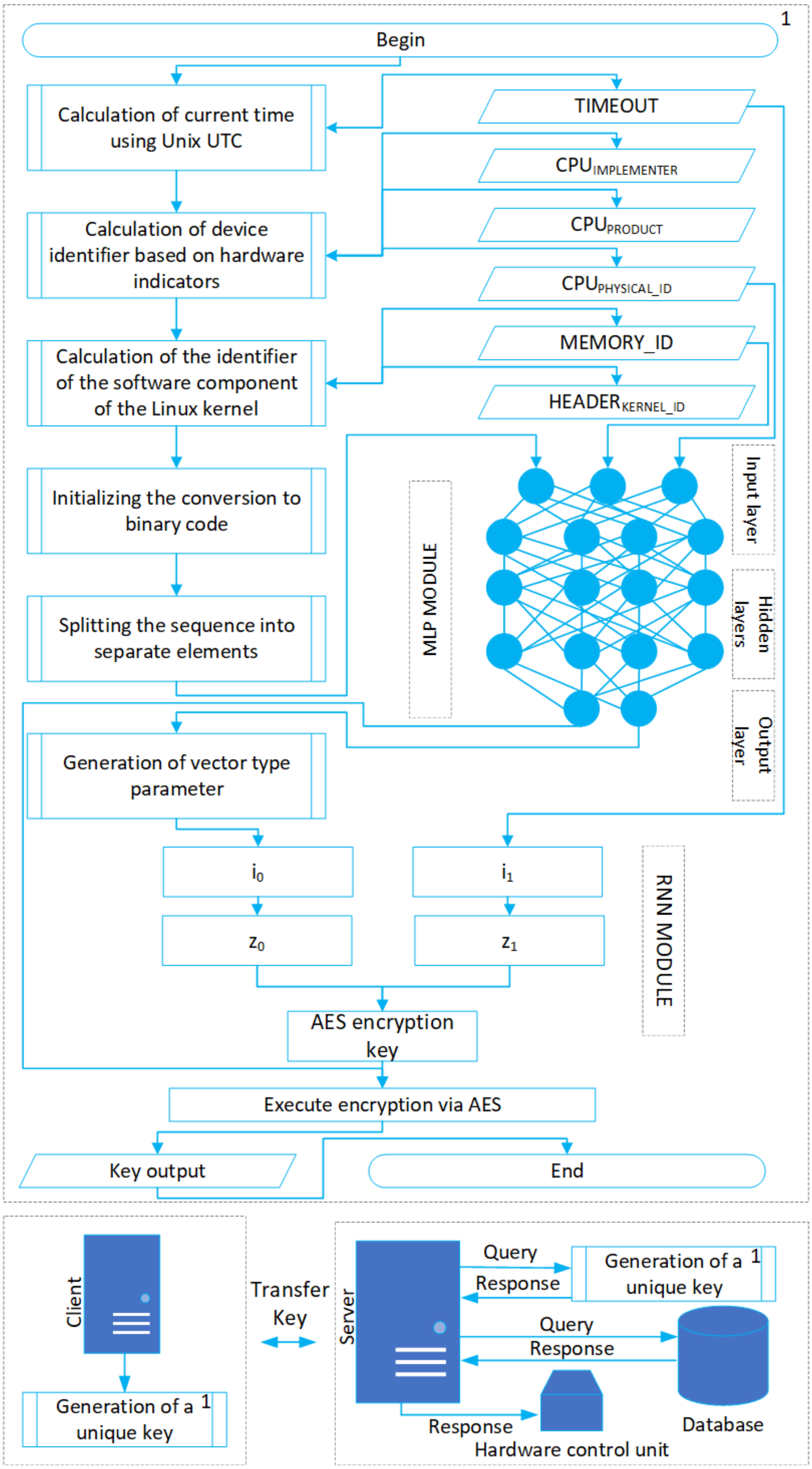

1
Begin
Calculation of current time using Unix UTC
TIMEOUT
CPU IMPLEMENTER
CPU PRODUCT
Calculation of device identifier based on hardware indicators
CPU PHYSICAL_ID
MEMORY_ID
Calculation of the identifier of the software component of the Linux kernel
HEADER KERNEL_ID
Input layer
Initializing the conversion to binary code
MLP MODULE
Hidden layers
Splitting the sequence into separate elements
Output layer
Generation of vector type parameter
i0
i1
z0
z1
RNN MODULE
AES encryption key
Execute encryption via AES
Key output
End
Client
Generation of a unique key 1
Transfer Key
Server
Query
Response
Generation of a unique key 1
Query
Response
Database
Response
Hardware control unit

Figure 6: Algorithm of interaction of the software and hardware complex under development

4. Conclusion

The developed software solution makes it possible to automate the access control process, while using the processes of multithreaded key processing with the help of neural technologies - multilayer perceptron modules and recurrent neural networks in parallel processing by applied encryption tools provide the highest level of security without the possibility of restoring the original sequence. The main method of identification data transmission is carried out using application hardware, represented by a set of application hardware modules. In this case, the use of software solution on the hardware component of processors ARM family allows you to avoid additional economic costs and increase the profitability of the product, which makes it possible to install it in small and medium-sized businesses.

Thus, the application of multi-level system of target data processing at the software level allowed to create a system of user identification based on the generation of dynamic sequences in certain time intervals without the ability to restore the generated sequence, which provides the highest level of security and ensures sales efficiency in certain areas of the market.

5. Acknowledgements

The reported study was funded by RFBR, project No.20-37-90062.

6. References

[1] S. Jetty, S. Rahalkar, Securing Network Infrastructure: Discover practical network security with Nmap and Nessus 7, Birmingham, Packt Publishing, 2019, 62-85.

[2] R. E. Smith, Elementary Information Security. 3rd. ed., Burlington, Jones & Bartlett Learning, 2019.

[3] S. Chin, S. B. Older, Access Control, Security, and Trust: A Logical Approach (Chapman & Hall/CRC Cryptography and Network Security Series), 1st. ed., United Kingdom, Chapman and Hall/CRC, 2010.

[4] S. Juba, A. Volkov, Learning PostgreSQL 11: A beginner's guide to building high-performance PostgreSQL database solutions, 3rd. ed., Birmingham, Packt Publishing, 2019, 54-71.

[5] H. Schönig, Mastering PostgreSQL 12: Expert techniques to build scalable, reliable, and fault-tolerant database applications, 3rd ed., Birmingham, Packt Publishing, 2019.

[6] W. Stallings, Cryptography and Network Security: Principles and Practice (7th Edition), 7th. ed., London, Pearson, 2016, 339-419.

[7] S. Harris, D. Harris, Digital Design and Computer Architecture: ARM Edition, 1st. ed., Burlington, Morgan Kaufmann, 2015.

[8] M. Kerrisk, The Linux Programming Interface: A Linux and UNIX System Programming Handbook, 1st. ed., San Francisco, No Starch Press, 2010, 185-513.

[9] E. Nemeth, G. Snyder, T. R. Hein, B. Whaley, D. Mackin, UNIX and Linux System Administration Handbook (5th Edition), 5th. ed., United States of America, Addison-Wesley Professional, 2017, 901-948.

[10] R. Hertzog, J. O'Gorman, Kali Linux Revealed: Mastering the Penetration Testing Distribution, Illustrated ed., New York, Offsec Press, 2017.

[11] V. Sze, Y. Chen, T. Yang, Efficient Processing of Deep Neural Networks (Synthesis Lectures on Computer Architecture), United States, Morgan & Claypool Publishers, 2020, 26-48.

[12] S. O. Haykin, Neural Networks and Learning Machines (3rd Edition), London, Pearson, 2008, 47-66.

[13] Y. Goldberg, G. Hirst, Neural Network Methods in Natural Language Processing (Synthesis Lectures on Human Language Technologies), Burlington, Morgan Kaufmann, 2017.

[14] A. Dorzhigulov, A. James, Deep Neuro-Fuzzy Architectures. 2020. URL: https://www.researchgate.net/publication/332301666_Deep_Neuro-Fuzzy_Architectures. doi:10.1007/978-3-030-14524-8_15.

[15] J. M. Keller, D. Liu, D. B. Fogel, Fundamentals of Computational Intelligence: Neural Networks, Fuzzy Systems, and Evolutionary Computation (IEEE Press Series on Computational Intelligence). 1se ed. Hoboken, New Jersey: Wiley-IEEE Press, 2016.

[16] J. Aumasson, Serious Cryptography: A Practical Introduction to Modern Encryption, San Francisco, No Starch Press, 2017, 106-131.

[17] H. Delfs, H. Knebl, Introduction to Cryptography: Principles and Applications (Information Security and Cryptography), 3rd. ed., Berlin, Springer, 2015.

[18] J. Loy, Neural Network Projects with Python: The ultimate guide to using Python to explore the true power of neural networks through six projects, 1st. ed., Birmingham, Packt Publishing, 2019.

[19] A. Gulli, A. Kapoor, S. Pal, Deep Learning with TensorFlow 2 and Keras: Regression, ConvNets, GANs, RNNs, NLP, and more with TensorFlow 2 and the Keras API, 2nd ed., Birmingham, Packt, 2019.

[20] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems 2nd Edition, 2nd. ed., Newton, O'Reilly Media, 2019.

[21] M. Avendi, PyTorch Computer Vision Cookbook: Over 70 recipes to master the art of computer vision with deep learning and PyTorch 1.x, 1st. ed., Birmingham, Packt Publishing, 2020.

[22] H. -J. Schönig, Mastering PostgreSQL 12: Advanced techniques to build and administer scalable and reliable PostgreSQL database applications, 3rd ed., Birmingham, Packt, 2019.

[23] A. Beaulieu, Learning SQL: Generate, Manipulate, and Retrieve Data, 3rd ed., United States of America: Addison-Wesley Professional, 2010, 41-52.

[24] B. Frain, Responsive Web Design with HTML5 and CSS: Develop future-proof responsive websites using the latest HTML5 and CSS techniques, 3rd ed., Birmingham, Packt Publishing, 2020.