

**Igor Halenár**

**Using Neural Networks for Detection of  
Anomalous Traffic in Automation Networks**

# **Scientific Monographs in Automation and Computer Science**

Edited by  
Prof. Dr. Peter Husar (Ilmenau University of Technology) and  
Dr. Kvetoslava Resetova (Slovak University of Technology in  
Bratislava)

**Vol. 7**

**USING NEURAL NETWORKS FOR  
DETECTION OF ANOMALOUS  
TRAFFIC IN AUTOMATION  
NETWORKS**

Igor Halenár



Universitätsverlag Ilmenau  
2012

## Impressum

### **Bibliographic information of the German National Library**

The German National Library lists this publication in the German national bibliography, with detailed bibliographic information on the Internet at <http://dnb.d-nb.de>.

Author's acknowledgement to Gabriela Chmelíková for translation.

This scientific monograph originated from the author's dissertation thesis defended at the Slovak University of Technology in Bratislava, Faculty of Materials Science and Technology in Trnava.

#### **Reviewers:**

Prof. Dr.-Ing. habil. Peter Husar  
Prof. Ing. Juraj Spalek, PhD.  
Ing. Augustín Gese, CSc.

#### **Author's contact address:**

Ing. Igor Halenár, PhD.  
Slovak University of Technology in Bratislava  
Faculty of Materials Science and Technology in Trnava

Ilmenau Technical University / University Library

### **Universitätsverlag Ilmenau**

Postfach 10 05 65  
98684 Ilmenau  
[www.tu-ilmenau.de/universitaetsverlag](http://www.tu-ilmenau.de/universitaetsverlag)

### **Production and delivery**

Verlagshaus Monsenstein und Vannerdat OHG  
Am Hawerkamp 31  
48155 Münster  
[www.mv-verlag.de](http://www.mv-verlag.de)

**ISSN** 2193-6439 (Print)  
**ISBN** 978-3-86360-049-5 (Print)  
**URN** urn:nbn:de:gbv:ilm1-2012100213

---

Titelfoto: [photocase.com](http://photocase.com)

## **Abstract**

Opening of local communication means of technological devices towards networks available to the public, supervision of devices, and remote technological device administration are the characteristics of modern automation. As a result of this process, the intrusion of unwanted elements from the Internet into control networks occurs. Therefore, in communication and control networks we have to build in active means to insure access to individual technological process components.

This contribution is focused on the insurance of control system data communication via neural network technologies in connection with classical methods used in expert systems. The solution proposed defines a way of data element identification in transfer networks, solves the transformation of their parameters for neural network input, and defines the type and architecture of a suitable neural network. This is supported by experiments with various architecture types and neural network activation functions and followed by subsequent real environment tests. A functional system proposal with possible practical application is the result. The determination of parameters for unambiguous data packet identification, proposal and transformation of the data for neural network input are the most significant outcomes of the contribution. The final experimentally verified proposal itself, as well as subsequent neural network implementation, represent a suitable way for utilization of such a network type in the field. The integration scheme proposal of the dedicated system into real production structure and the execution of the conversion of data elements representing the data packet parameters to be used as an input into the proposed neural

network are the main contributions of related research. In addition, software able to carry out conversions for a random file size with continuous communication in the data network recorded was created. Determining data transfer parameters capable of specifying unambiguously the data packets' parameters, and subsequently the neural network proposal, able to detect and control the communication network operation according to parameters selected by us, are essential parts of this contribution as well.

### **Key words**

automation, neural network, security, communication, firewall

## **LIST OF ABBREVIATIONS**

IPS	Intrusion prevention system
TCP	Transmission control protocol
IDS	Intrusion detection system
UDP	User datagram protocol
ACL	Access control list
IPSec	Internet protocol security
DOS	Denial of service
IP	Internet protocol
DNS	Domain name system
SSH	Secure shell
SSL	Secure sockets layer
ICMP	Internet control message protocol
IT	Information technology
HTTP	Hypertext transfer protocol
HTTPS	Secure hypertext transfer protocol
FTP	File transfer protocol
MAC	Media access control
NN	Neural Network

## INTRODUCTION

Production process controls are being replaced practically everywhere by discrete automated control systems. This dynamic development is rapidly increasing the importance of automation control systems and processes. The information transfer to the control system has to be as immediate as possible; a direct on-line connection is preferred. To work properly, the control system itself needs the information on results and feedback. All these parts necessarily require properly working communication channels capable of transferring the essential information quickly and reliably. The more complex the control system is, the higher the requirements for its control are. In the case of extensive systems, the control computer is utilized as a control system. This way it can be used for controlling even very complex systems in real time. The control process is based on current information on the values of input and controlled quantities, on the state of individual system parts and other relevant information. Communication channels represented by the file of protocols as well as physical transfer media are used to transfer the values.

In building and operating the communication networks, their functionality is commonly emphasized. If the characters of their utilization or legislation do not require meeting security aspects, the information security is understood as the extension of the system functionality and not as its part. This is caused by the unawareness as well as by the fact that the security measures lower the system functionality and make its use more complicated.



It is also obvious that the effort to control complex and accelerating processes properly necessarily leads to the requirement for faster and better information transfer.

This contribution focuses on the proposal and solution to the system utilizing artificial intelligence methods to investigate control systems' communication issues and data transfer. The application of neural networks in cooperation with expert control of data transfer represents one of the possible solutions leading to quality communication channels of control systems.

## **1. DATA TRANSFER IN NUMERIC SYSTEMS CONTROL**

Communication systems control represents a large field of issues. To ensure the control and communication of automation systems via communication interface, the utilization of many technical and program means is necessary. The control structure of this communication together with technical and program means is defined by the **network architecture**. One of the best solutions is to implement the attempt to classify the issue into the hierarchically arranged layers. There are many decomposition methods representing various layer models of network architecture. Such models provide higher flexibility in terms of changes in network architecture. At present, mostly TCP/IP Protocols based communication is utilized. In the case of conventional calculation systems, it is a commonly used protocol; however, this is not true for control systems. The elements of technological processes such as PLC, regulators, action members, sensors, etc. commonly utilize industrial protocols based on other communication ways than classical IT informatics (18). The current trend is to interconnect the industrial network types and classical Ethernet technology which results in various super-structural technologies for TCP/IP Protocols dealing with the compatibility of automated and classical calculation systems, however, these bring errors and risks to automation networks as well (72).

### **1.1 Communication standards of control and production systems**

To control the automation systems in terms of transfer infrastructure, it is necessary to meet higher quality parameters than by a common communication in computer networks (30). It is mainly the possibility of

high-speed communication and determination transfers in time-critical input-output information for processes and systems control. Classical communication interfaces included in IEC61158 or in S50.02 ISA Standards (Actual Sensor interface, CAN-bus, Controlnet, etc.) are proposed directly for the needs of control (21). Nevertheless, the separateness of the system represents a disadvantage. The disadvantage can be eliminated by the implementation of new protocols for automation networks communication (72). Then, by the interconnection of automation and classical IT networks we will acquire the following:

- integration to generally available networks with classification to Internet/Intranet, the possibility of distant configuration,
- higher accessible speeds, the possibility of transferring larger data packages,
- the possibility of addressing and controlling more devices at larger distances,
- the possibility of developing homogenous communication networks, i.e. automation and data networks in one protocol,
- the subsequent possibility of developing MES systems, online control, firmware update, errors correction, etc.

### ***1.1.1 Automation networks protocols encapsulation***

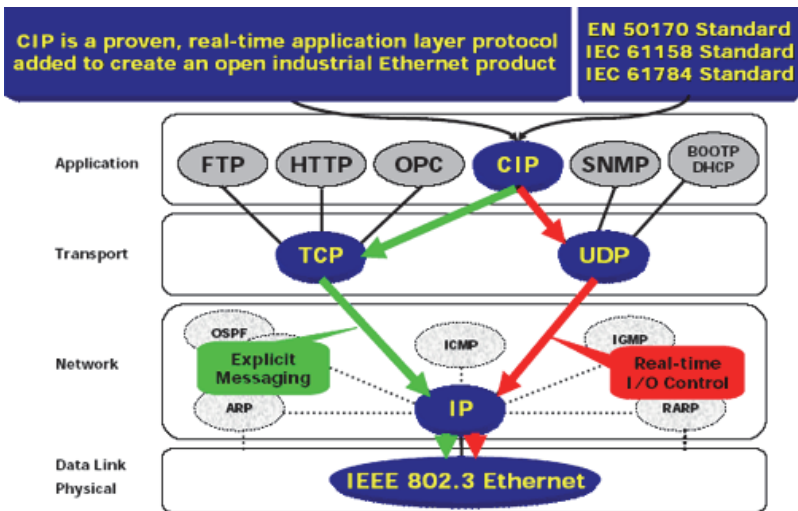
The mechanism of data transfer in automation networks in renowned companies (Siemens, Rockwell) is usually proprietary. Nevertheless, the utilization of ICP/IP Protocol model is the same. Well introduced 1-4 Ethernet technologies standards provide the common base. This means that IEEE 802.xx technologies (Layer 1), CSMA/CD Access method (Layer 2),

IP (Internet Protocol, Layer 3) and TCP and UDP Protocols (Layer 4) are utilized for data transfer. Besides common elements, some application layer (Layer 7) protocols were also accepted, such as Hypertext Transfer Protocol (HTTP), File Transfer Protocol (FTP) and SNMP Simple Network Management Protocol (26). In general, the basic disadvantage of the classical IT communication is network reaction time, e.g. standard reaction time of the Ethernet is approximately 100ms. In local networks with lower number of devices it is about 20ms. Therefore, in specific cases UDP Protocol instead of TCP Protocol is used for technological processes control (reaction 10 ms), and technology of direct MAC addressing in local segments (1ms) (33).

Commonly utilized accesses are as follows:

- Ethernet/IP

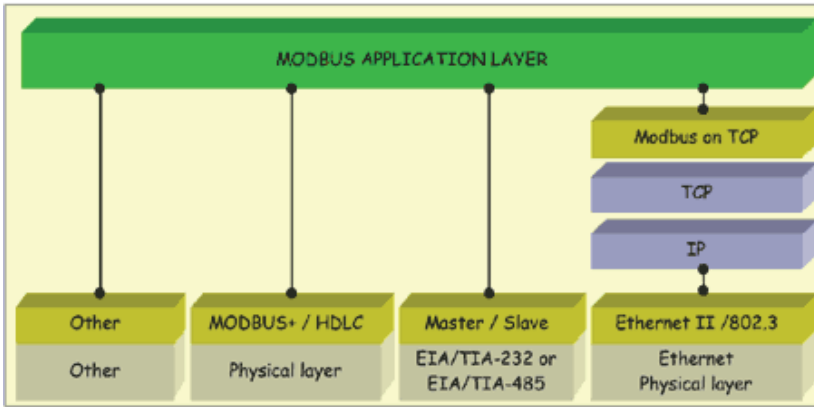
Industrial Ethernet Protocol developed by Rockwell Automation is specified as IEC 61158 (28). For encapsulation of packages TCP/IP Communication System is used, while the super-structural Layer 7 is built by CIP Protocol. This Protocol directly supports automation interface of DeviceNet, and ControlNet. The position of the aforementioned CIP Protocol within seven-layer RM OSI model and positions of TCP and UDP Protocols is illustrated in Figure (28).



*Fig. 1 CIP encapsulation in TCP/IP*

- Modbus-TCP

The protocol superstructure for Modbus automation networks control developed by Modicon Company represents the superstructure of TCP/IP Protocols and corresponds to the seventh (application) RM OSI layer. Figure 2 (*Fig. 2 Modbus encapsulation*) shows the position of Modbus-TCP Protocol to RM OSI model position. In the model shown it represents the seventh (application) layer and utilizes all protocol services of the third and fourth layers represented by TP and IP Protocols (32).



*Fig. 2 Modbus encapsulation*

- ProfiNET

ProfNET is utilized mainly in devices used by Siemens Company (Symatic NET) and it is standardized via IEC 61158 and IEC 61784. It is based on the Ethernet technology complemented by the mechanisms for Real-time Control of processes. Especially services of SNMP, HTTP and TELNET from TCP/IP Protocols are utilized. (40)

Besides the aforementioned standards, there are more automation protocols standards (e.g. CC-Link, EtherCat, SERCOS III, Versatile Automation Random Access Network, IEEE 1588, etc.) (69). The mentioned protocols, and Profinet (Symatic) in particular, are the most common.

### *Note to automation networks security*

The common sign of modern automation is to open local communication means of technological devices towards publicly available networks, or possibly the compatibility of communication protocols. On-line supervision of devices and remote administration of technological devices result in the intrusion of unwanted elements from the Internet to control networks, which leads to the requirement of considering the unauthorized access to the network as well. Therefore, it is necessary to implement security means for the access to individual technological process components in communication and control networks.

According to PA Consulting Group Agency, the average amount of damage caused by viruses in automation networks is approximately €1.5 mil. The incidents and damages (registered by CERT Institute from 2000 to 2006) show an increase of 50-100% a year. Protection against these incidents shall therefore be a part of all automation and production systems strategy proposals (34).

## **1.2 Reference structure model of communication protocols**

The reference model of network architecture of open systems interconnection is accepted as an international standard. It is defined by ISO (International Standards Organization), registered as ISO 7498 Standard, or RM OSI Standard (*Fig. 3 ISO/OSI model*) (19). The model comprises specifications of individual layers and freely describes the interfaces among the layers, especially the communication and transport protocols origin as

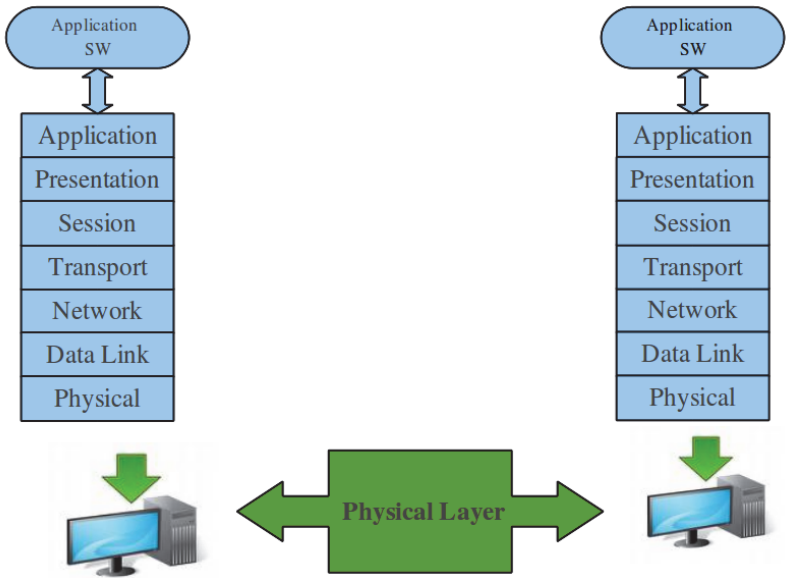
independent ISO standards, or standards of other institutions (IEEE, CCITT, etc.) (83).

The three bottom layers are network dependent layers, since their functions depend on the specific type of the network. These layers can see only a part or all actual network topology. The three upper layers belong to application-oriented layers, for which the network type is transparent and they are used as program support for application processes. The interface among these groups is made by the transport layer.

An **entity** is a process unit, which is indicated according to its layer affiliation (application entities, presentation entities); the entities on the same level are indicated as peer entities. Entities in  $N$  layer implement the services used by  $N+1$  layer. In this case, the  $N$  layer is called *a service provider* and subsequently the  $N+1$  layer is called *a service user*. To exchange the control and data information, the established buffers indicated as *SAP* (service access points) are used, e.g. for communication of two entities from neighbouring layers (83).

The entire model, as well as its specifications, is described in sources (19), (24).



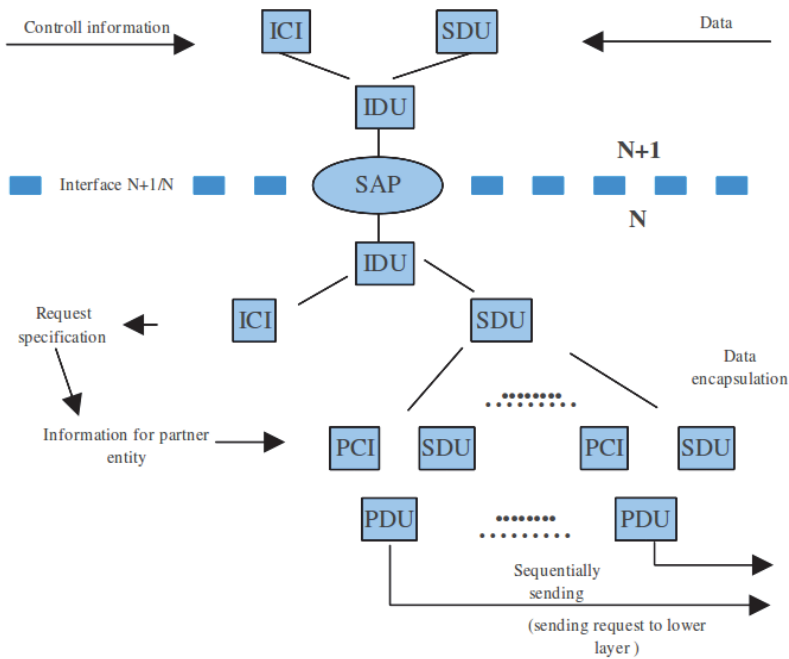


**Fig. 3** ISO/OSI model

A communication unit comprises several parts:

- Interface Control Information ICI
- useful data – Service Data Unit SDU

Together they build an *Interface Data Unit* (IDU). Data flows transported among subjects within RM OSI are illustrated in Fig. 4 (*Fig. 4 Data flow in ISO/OSI model*).



*Fig. 4 Data flow in ISO/OSI model*

### 1.3 Protocols of RM OSI 3rd and 4th layers

The protocol can be defined as a communication regulation for systems requiring carrying out the data exchange via some communication channel (24). The main point of this contribution is to propose a system with a neural network module serving to ensure the systems communication via TCP/IP network. Obviously, the most commonly used protocols in this case are TCP, IP and UDP Protocols. The perfect comprehension of communication via these protocols is crucial. According to the assignment

of communication protocols to individual ISO/OSI model layers, IP Protocol (network layer) is on the lowest level and ensures the processes related to addressing within the network.

The description of individual areas in the IP Dataprogram Package structure is not necessary. They are strictly given and can be obtained from related standards (37, 38). Nevertheless, it is necessary to know the function and importance of individual areas, namely in the case of directing packages, or in the case of identifying their origins and targets. Regarding the data, the basic filtration of transferred data is carried out. The areas of *source IP address* and *destination IP address* can also be one of the input data for learning and then for subsequent deciding on neural network.

Since it is the basic transport protocol, practically the whole communication within the networks based on TCP/IP is executed under this protocol (84). Source and destination address in the headings of TCP and IP Packages is one of the significant characteristics utilized by the evaluation of data packets' correctness and security.

## **2. SECURITY OF CONTROL SYSTEMS IN DATA TRANSFER**

At present, not only is performance important, but data transfer security and secure data access are significant as well. The protection of confidential data is possible in some specific places – interface of private/public side network, remote access, or remote administration of devices, servers' security, etc. Methods of protection correspond to the place of implementation and usually in other ways, which can be combined.

Considering the function principle of the Internet network, the utilization of the Internet as a transfer medium is quite suitable. Protocols used for communication originated directly from remote data transfer. Gradually, with network development and increase of users' techniques and connected client stations, the necessity of transferred data protection occurred. At present, communication security can be understood as a whole system of protection elements (administration of passwords, access levels, communication coding, security gates, hidden networks, certification authorities, etc.) serving to avoid unauthorized access to the system or personal computer network. We have a number of standards and protocols at our disposal; and by applying the protocols, we can avoid the majority of attacks.

The security of systems is frequently not taken as a complex. This means that there is usually quite a strong protection by the input to the local system from the Internet; however, the communication itself is executed in the unsecured (unencoded) channel. We provide the possible attacker space for data acquisition when they can bug the communication. After the

analysis of the Ethernet frameworks of such bugged communication, it is possible to acquire all the data transferred, including passwords.

It may seem that these issues relate only to the field of computers and computer technology, understood as a personal computer, however, it is essential to realize that modern production lines and systems (e.g. presses, machining centres, etc.) are directly connected to the Internet and their producer is, via remote access, directly capable of modifying the parameters and recording the operational software. We can imagine that damaged software in a machining centre can result in production failure, which can bring enormous economic loss.

## **2.1 Categories of communication systems attacks**

In any system, not only the computer system, such a situation can occur (sooner or later it almost always happens), that functionality will be damaged by an attacker's activity to change the system properties in favour of other parties. Obviously, the system is damaged, or if the system works comparatively well and on standard, besides functions required by us it carries out other activities as well. In case it did not come to the change of system behaviour, it could have come to the system data change or misuse. Both cases represent a considerable problem to be taken care of.

Communication can be destroyed in two ways. First, by damaging communication transfer channels either by physical damage of transfer, or by the modification of transferred packets. Secondly, damage can occur by overloading the communication network nodes via appropriate frequency of meaningless requirements and orders. The system usually denies such requirements, nevertheless, regarding the quantity, and then it is not capable

to carry out specific activities (DOS attack). Considering the servers' performance and technologies that allow splitting the load on multiple servers, or possibly filtering and modifying the transfer channel zone width, such an attack from only one computer is practically impossible. Distributed DOS, i.e. dDOS, is widely spread. In this case, the attack is executed from the number of calculation systems connected to the network. Protection is practically impossible.

The amount of attacks on information and control systems is significant; therefore, the field of research and investigation is large as well. In general, it is possible to divide the infiltration attempts into communication networks into two basic groups, *active and passive attacks*.

Passive attacks on communication networks basically only monitor the network operation. Their defectiveness lies in the misuse of acquired data in further activities. In this case, as the title also suggests, no modification of transferred data happens.

Active attacks represent the attempt to change the transferred data and provide the possibility of communication set damage. They are the following attacks:

- *eavesdropping, replay attack*
- *Man-in-the-Middle attack*
- *password attacks*
- *Integrity attack*
- *Identity spoofing, IP address spoofing*
- *Attack on accessibility/availability (Denial Of Services, Distributed DOS).*

The result of such damage of the communication infrastructure means practically destroying communication means. The impact of such an attack is enormous. Systems are without control, supervision and hence the consequences can frequently be fatal.

Basically, according to the main method used, it is possible to classify these technologies into *IP Protocol Misuse* and *TCP Protocol Misuse*.

**Misuse of IP Protocol:**

- Pretended IP addresses
- Misuse of TTL

**Misuse of TCP Protocol:**

- SYN Flooding
- Backscatter

**Misuse of ARP Protocol**

- ARP Flooding
- MAC Flooding

**Misuse of ICMP**

- Smurfing
- Pretended ICMP packages

**Misuse of UDP**

- ‘Fraggle’ attack

**2.1.1 *Special techniques of data transfer***

- Covert Channels

This method represents a simple but efficient mechanism for data exchange among the systems without any recognition of the activity by the

firewall or IDS system. The technique is based on the principle of utilizing such communication TCP Protocol ports, which are not usually blocked via security systems (TCPP53, UDP/53, etc.). Apart from the aforementioned, for such a communication, it is possible to use common packages as well, while the covered data are transferred directly in “harmless” data represented by certain changes in TCP or UDP packages headings.

This data transfer technique is mostly resistant to all standard firewalls and IDS systems. The revelation of covered channels is possible only by constant analysis of network communication. However, a large amount of data can be transferred in the network, therefore, considering real conditions such an analysis is almost impossible.

A possible solution is the application of a device controlled by a neural network (called an intelligent firewall), which can process the behaviour formulas of the communication set in transfer. The process is obviously demanding and requires calculations and it is not possible to guarantee the revelation of all hidden communication.

- Out Of Band Communication

It is a special technique of data exchange, in which apparently unrelated data in information systems are utilized, e.g. the change of access rights of a specific file. If there are changes continuously being carried out (denial/prohibition of registration, permission/authorization of registration in the file, change of property right, etc.), by monitoring these changes, it is possible to transfer data. These data exchange techniques are very complex and it is practically impossible to reveal them, due to the fact, that for such a data exchange transfer various hardware and software elements can be



combined. They can also very “exotic” combinations, e.g. the aforementioned change of access rights together with the change of processor frequency, or possibly with occupying or not occupying some address in the memory, etc.

## **2.2 Ways of industrial systems security**

All systems connected to the computer network are exposed to the misuse by an unauthorized person via the Internet. In this case, we should not consider only “personal computers”, in which an office agenda is carried out. It is necessary to include the control computers of technological processes, production lines, etc. as well. In such cases, the protection of computer networks against danger takes on a completely different dimension. In general, we can divide the network protection into two main parts. The first one is the protection of exchanged data, the other one the protection of systems (80).

- The protection of exchanged data means the process by which we replace the commonly used communication protocols by their “safer” format. Therefore, it is suitable to use safe HTTPS instead of HTTP. Similar situations are possible in practically all communications via networks (76).

The protection of systems against an attack represents a self-contained file of means and activities ensuring the information systems security against attacks from the outside of the network. Firewall, honeypots and IDS systems are the basic elements.

### **2.1.2 Firewall**

It represents the basic security element. It checks the network traffic among zones with various degrees of trust (Internet – LAN).

They can be classified according to used technology as follows:

- IP filter – the simplest firewall. It is a “blocker” of Internet communication. In this type, there are defined rules on communication permission or prohibition on individual ports. The operations on the network, that are not prohibited, are therefore permitted. Low flexibility of such a solution is a disadvantage, as well as the fact that, due to security, it is necessary to forbid a large amount of input-output ports.
- Status firewall – or status IP filter is a more advanced technology of network security. This device checks the data flow and communication of higher layer protocols of ISO/OSI model, i.e. TCP and UDP Protocols. The standard regime of such a device permits any inside-out network communication out, and forbids outside-in Internet communication. Hence, it permits only the packages related to inside-out communication.
- Proxy server – superstructure of the status firewall. It is a program, which carries out the filtration on the application layer directly for specific applications. For example a standard port 110 (downloading mails, POP Protocol) can be accessible from the network only by the program working with electronic mail. The communication via other program is prohibited in this port.

Firewalls are frequently utilized also by more sophisticated security devices, e.g. IPS, to prepare provisions against the attack found.

### **2.1.3 IDS**

Intrusion Detection System (IDS) can be defined as a file of tools, methods and sources helping reveal, record or announce the attempts on network intrusions and attacks (72).

IDS, i.e. intrusion detection systems operate on the network layer and usually they are passive systems which are not aimed at attack prevention. The primary purpose of IDS developing is to reveal the attack, not to execute appropriate measures to avoid it. Systems trying to avoid attacks are called Intrusion Prevention System - IPS. Regarding development, some of the functions have grown up to prevention measures against attacks, and then some IDS become active. They were renamed as Intrusion Detection and Prevention Systems – IDPS (70). Any use of prevention functions should be indicated in IDS records.

Patterns of abnormal behaviour are many; however, in general, they include unneeded, harmful or illegal activities occurring within the system. We distinguish two main methods of intrusions detection, i.e. misuse detection and anomaly detection – AD (43).

According to the principle of activity, intrusions detection can be divided into two groups:

- statistical anomaly detection
- patterns comparison.

### 3. ANALYSIS OF CURRENT NEURAL NETWORKS TECHNOLOGIES

Artificial neural networks attempt to model the abilities to process information in the form of a nervous system. Due to fast computer technologies development, they are largely utilized not only in experimental tasks, but in practice as well. Application tasks such as information processing, patterns classification, optimization issues in the field of control, and prediction tasks exist in various fields of industry and it is possible to execute them via neural networks.

Neural network is a massive parallel processor with the tendency to preserve experimental knowledge and its further utilization (5). It imitates the human brain in two aspects:

- knowledge is collected in neural networks in the course of learning
- inter-neural connections (synaptic scaling) are used on the basis of knowledge

*Neuron* is the basic element in neural networks (7). Compared to human neurons, it is possible to develop (simulate) much faster neurons than the human ones. Nevertheless, the issue is the simulation of such an amount of neurons and their connections, which is similar to human brain. The number of neurons in the human brain is approximately  $10^{11}$  to  $10^{14}$ , while there are  $10^3$  to  $10^4$  of their connections to each neuron. It is a massive parallel structure, which is not possible to simulate in reality with current computer technologies.

In the implementation and study of neural networks, we distinguish three basic activities:

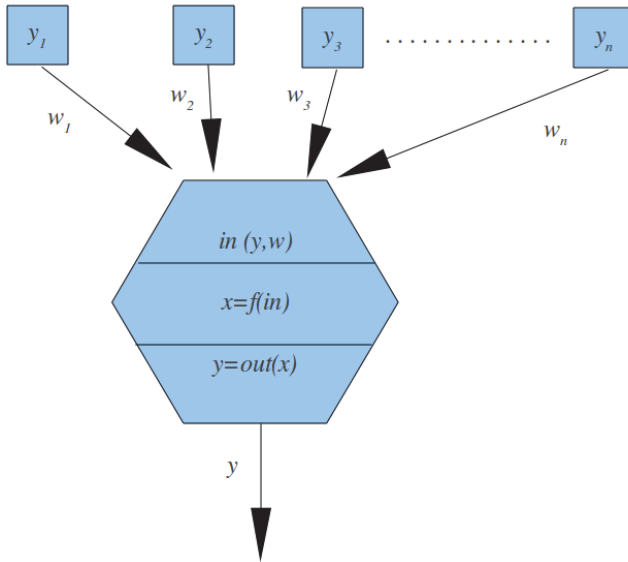
- theoretical analysis of neural networks – mathematical analysis of neural networks activities and their analysis from the point of dynamic systems
- Simulation of neural networks – simulation of neural networks via computer systems. The field includes the phase of neural networks and the process called neural network learning. The process of learning, however, demands quite a lot of calculations.
- Implementation of neural networks – implementation of learnt neural network into a result form suitable for practical use (neural chip).

### **3.1 Neural structure**

The neuron is the basic element as well as the basic process unit in neural networks (6).

Neurons are built by five main parts:

- input
- neuron threshold
- activation function
- neuron output function
- synaptic scaling.



**Fig. 5** *Mathematical model of neuron*

Regarding the synapses flow, we can divide the neurons into:

- pre-synaptic (source) – are located in the direction in front of the synopsis
- post-synaptic (destination) – in the direction behind the synopsis.

To indicate the synaptic scaling we can use indication  $w_{ij}$ , where “ $i$ ” indicates post-synaptic neuron and “ $j$ ” indicates pre-synaptic neuron. It is the synopsis coming out of “ $j$ ” neuron and directing towards “ $i$ ” neuron.

The neuron’s input is the function of individual inputs coming from pre-synaptic neurons. If we consider it is the sum of inputs with specific

scaling, then we can express the input into  $i$ -the neuron having  $N$  of pre-synaptic neurons by the following relation:

$$in_i = \sum_{j=1}^N w_{ij} o_{uj} + \Theta_i \quad [1]$$

where  $w_{i,j}$  is synoptic scaling and  $o_{uj}$  are neuron outputs, while  $\Theta_i$  is  $i$ -neuron's threshold (5).

The aforementioned equation can be rewritten as:

$$in_i = \sum_{j=0}^N w_{ij} o_{uj} \quad [2]$$

where  $w_{i0} = \Theta_i$  and  $o_{u0} = 1$  or  $-1$ . The threshold is the neuron's input from the surroundings (i.e. not from other neurons). That is in case there are no inputs into the  $i$ -neuron investigated from other neurons  $j = 1, \dots, N$ , then  $\Theta_i$  threshold represents the input into the neuron. Neurons having such an input are called sigma neurons (5).

Neural network has to be considered as a dynamic system, i.e. dependant on time. We can talk about the neuron state in time  $t$ , or in time  $t+1$ . Neuron's activation function is the function of the neuron input  $in_i(t)$ . The state of the neuron is defined by the variable  $x_i$  such as follows

$$x_i = f(in_i) \quad [3]$$

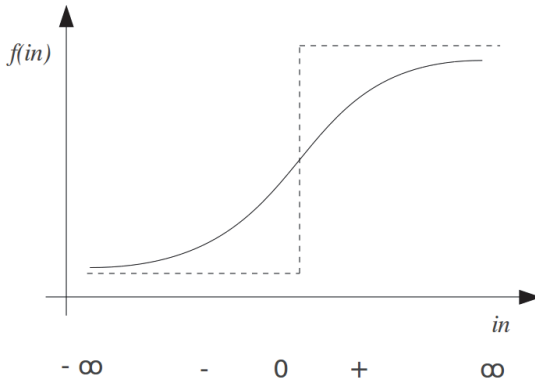
We call the function  $f(\ )$  as the activation function of the neuron. Basically, the neuron's output/axon is represented by the calculated value from the application of the activation function to sum of inputs, weighed by synoptic scaling.

It is possible to build an even more complex connection, where the combinations of inputs represent the input to the neuron (e.g. in the form  $w_{ij}y_iy_j$ ). Nevertheless, they are rarely used in practice due to demanding complex calculations.

Scaling  $w_{ij}$  is an adjustable part of neural networks. They are acquired by the process called neural network learning (or neural network training). Neural network training is carried out by the training set of data comprising combinations of input vectors and required output/axon vectors. There exist, however, networks with necessary learning (6).

From the mathematical point of view we can say that the neurons behave as a function (6). The neurons change their input values  $in_i(t)$  (in time  $t$ ) to limited output values via the activation function.

Sigmoidal function (*S curve*) is the most frequently used.

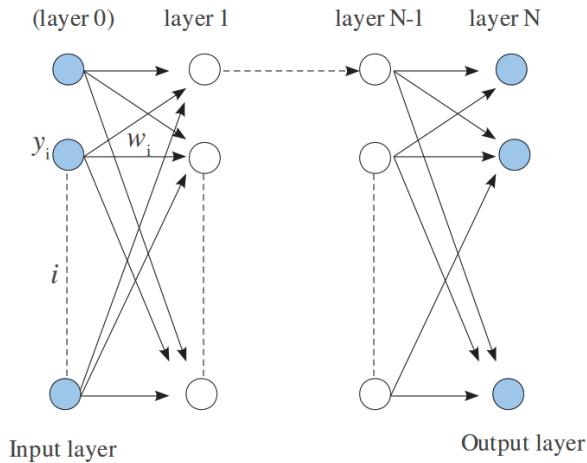


**Fig. 6** Sigmoid function



### 3.1.1 Topology of neural networks

In general, neural networks are built by neurons laid in layers, so that we can connect the outputs of one layer's neurons to the inputs of the output layer. We can describe the structure of these networks by a random oriented graph via the apices being represented by the neurons and by the edges representing their connections.



**Fig. 7** NN topology

In such a neural network type the layers are called as follows (5):

- Input layer ( $0$  layer), where there is usually data processing carried out. It is for the input data distribution from the surroundings to other neurons in other layers.

- Covered layer ( $I \sim N-1$  layers), in which neurons receive the input data from other neurons through threshold connections and their outputs continue to further possible layers.
- Output layer ( $N$  layer) is similar to the covered layer, however, the output goes out from the neural network.

Neurons classified into individual layers have the layer characteristics.

We can further divide the neural networks due to the direction of the signal spread by the synapses, and mainly according to the connections orientation:

- *Feed-forward* neural networks, where the signal is spread only in one direction, i.e. the input layer has only outputs and the output layer only receives data from lower neurons.
- *Recurrent neural networks*, where the output neurons represent also the input neurons and subsequently the layers are input and output at the same time. In this case it is very difficult to distinguish the input and output layers.
- Similarly, in neural networks the signal can spread among individual neurons by different ways.

#### **4. SUBJECT MATTER FORMULATION**

Information technologies represent complex relations, and therefore it is not possible to eliminate the danger of control systems damage by an unauthorized, incorrect input either from the inside or outside networks. Various technologies are used to protect the systems, e.g. Firewall, IDS, IPS, antivirus systems. The technologies develop and, also, people develop the systems which are able to deal with the issues and evaluate the data without being supervised physically by a person.

One possible way is to develop a system capable of executing complex calculations similar to human brain operations. There are various expert systems. They have one disadvantage: they are able to deal with various issues in the field and they can prove the solution as well; however, in the case of a false answer, it is necessary to modify the rules from the operator so that the further operation results in a correct solution. These systems are low in flexibility in situations where input information comprises a certain degree of uncertainty.

Hence the neural networks represent a shift in the systems development. They are able to learn and, if trained correctly, they bring excellent results. Their information, however, is a distributed form of arrangement, represented by neuron connection synoptic scaling (7). This property means that even a detailed analysis cannot determine how the neural network came to the result. Development of a hybrid system using the strengths of both types could be the solution (10).

Neural networks, or possibly hybrid systems, seem to be an optimal solution to the need of replacing an expert when defining the new rules by the change of inputs. The theory of neural networks is elaborate; there are several sources available. In general, they can be used for classification and functional approximation or mapping the functions by using a lot of training data, where the systems with clearly defined rules (e.g. expert systems) fail (7). There are many applications based on the utilization of various types of neural networks. Despite this fact, there are only few functional and really applicable solutions connecting the neural network directly with IDS or IPS systems (14, 15). The first devices of the kind were published in 1986 (16). As far as their principle is concerned, they were based on the utilization of statistical methods and they were looking the anomalies up in the data flows. Denning and Neumann introduced their system model called *MIDAS* (29) in 1988; it was an expert system made with the use of LISP language. The neural networks implementation in the anomalies evaluation was first published by Lunt in 1993 (36). He developed a system for anomalies detection called “*Next-generation Intrusion Detection Expert System (NIDES)*”. As the system title suggests, it was mostly an expert system. In 1991 in the University of California they developed a prototype called “*Distributed intrusion detection system (DIDS)*” (25). Nevertheless, this was again an expert system.

This contribution is aimed at proposing a dedicated system capable of securing the systems within the communication network against threats from outside. The connection of classical firewall functionality and advantages of neural networks represents the solution including the strengths of both. Obviously, all over the world there are systems to prevent

attacks on local networks (IDS, NIDS, and IPS). The efforts to implement various techniques to define possible attacks, as well as predict them, are clear (11), (12), (13). These methods are mainly determined for prevention, or possibly for detection. They mostly do not have the possibility of directly influencing the running communication. The security of networks is ensured via dedicated filtering devices – firewalls, however, the devices have no possibility to react directly and dynamically on possible changes in the input conditions. The issue can be solved by the connection of these two technologies. Nevertheless, the need of an operator to define the rules remains their disadvantage, as both of them are expert systems.

The proposed system should be able to eliminate attacks on data networks via an active filter represented by a firewall, controlled by IDS system in connection to the neural network. The efficiency of the device depends on the neural network type, therefore, it is necessary to select a suitable neural network type. The entire product should operate on a dedicated industrial system.

The aims of this contribution can be summarized as follows:

- analyze the security of control systems communication,
- analyze possible neural networks utilization by the data transfer validation and select a suitable neural network type,
- propose a functional security system model,
- via suitable tools, execute the model system of communication network security.

## **5. POSSIBILITIES OF NN USE BY SYSTEMS COMMUNICATION VALIDATION**

Traditional computers ensuring security of data and automation networks, and subsequently the security of industrial, information and production systems (firewalls) are based on more or less complex files of security rules (68). The data entering such a system are processed as an input and output based on one or more rules from the base of rules. The base of rules is structured as a file of sequenced logic operators (Boolean). In cases where the base of rules becomes larger, and/or more complex, it naturally requires a larger processor performance and more system sources. In practice, it frequently means that the file of rules represents a compromise between the efforts to cover all the possible firewall states by rules and by required obtained data penetrability (74). It means that we usually sacrifice security in favour of higher penetrability. Besides, a frequent intervention of operation personnel in the administration or modification of the base of rules is necessary. Obviously, even when it is most spread, the base of rules works in the same way of linear processing. In addition, besides various limitations in all classical systems, e.g. limited capacity of storing space, limited possibilities of logical mathematics, and need of demanding complex calculations in extremely large bases of rules, in the case of classical firewalls the systems are still static. Their security level corresponds proportionally to the operator's abilities and knowledge. Since they are static firewalls, their automated adaptation and learning from data flows coming through the data network are not possible. Subsequently, the systems do not allow performing of comparisons and analysis of data

flows and avoid the pitfalls and security threats, or possibly simplify the work of administrators.

Therefore, it is necessary to dedicate certain sources and means for devices' development; we can call them firewalls with artificial intelligence. Such a device represents the connection of a functional neural network and classical firewall. The system is then capable of learning from the data flows coming through it, or it is possibly able to adapt to changed input conditions, and hence ensure higher security requirements. It would also be useful to include the requirement of combining various methods of risks analysis into the development of the intelligent security system method, which would provide a higher degree of functionality and security than classical firewalls. In the use of neural networks, it is possible to develop an independent intelligent security network system – an intelligent firewall comprising the knowledge of possible security risks and threats and a system capable of dynamically adapting to possible malfunctions and attacks of the communication system, or possibly more complex forms of intrusions/infiltrations (exploits, out of band communication, analysis of covered communication channels).

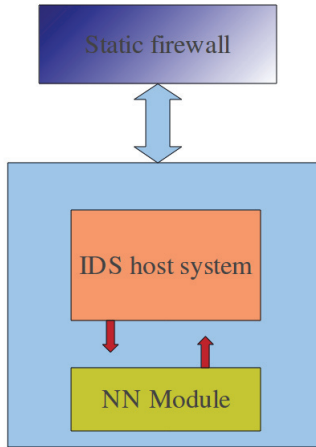
The system, with such a heuristic analysis of data flows, should optimally eliminate the shortcomings of classical devices, as well as should be able to adapt to the changed input conditions in computer communication. This contribution is aimed at proposing and implementing a system offering the advantages of neural networks integration into a classical IDS security model. The proposal and implementation of a suitably selected neural network and its integration into the proposed data transfer/transport system is the basic element.

## 5.1 System proposal

The proposal of the entire system consists of more parts. Individual steps of the functional model implementation to ensure the communication via neural networks correspond to the stages of the implementation. Necessary steps for practical implementation of the proposed system (*Fig. 8 NN system design*) can be summarized as follows:

- necessity (by some suitable way) to implement the main core of the security system,
- propose, carry out and implement the functional model of the neural network, meeting the requirements for possible implementation as proposed,
- select a suitable element of active security of the protected network, which is able to dynamically change the parameters of communication set data transfer on the basis of the neural network module's outputs,
- implement the solution by suitable software.





**Fig. 8** *NN system design*

The field connecting the IDS and NS module is a very simple software element best placed on a dedicated system. The component indicated as a static firewall can then be placed on another system, either on the dedicated system or the system with aggregated communication functions.

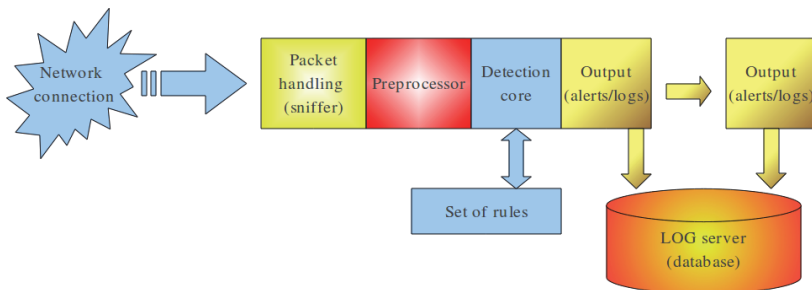
## **5.2 Proposal of host system**

Regarding the complexity and extent of the subject matter, the development of the system itself is beyond the contribution framework. It is efficient to choose an existing system as the basis and implement the security via neural network as a complement, or possibly a superstructure of the system. Obviously, such a system shall meet the following requirements:

- open system –with the ability to program extending modules,
- ability to capture packets – IP and TCP/UDP packets represent the basis for data transfer in the network, therefore it is necessary to grab the packets in the communication network,
- record of packets –an important point due to the possibility of repeating the learning cycles of a neural network,
- existence of communication interface – to ensure the active filter control of the data transfer in a network (firewall).

There are more systems to detect security deviations in the communication network, however, only a few of them are practically utilized. There are particularly the following five software solutions: Snort (90), Untangle (91), Bro NIDS (92), Prelude Hybrid IDS (93), OSSEC HIDS (94), Flowmatrix NBAD (95).

Regarding the aforementioned criteria we select IDS Snort licensed by GNU GPL as a host system (96). The system architecture is modular and is illustrated in the following scheme (*Fig. 9 Architecture of Snort IDS*) (96).



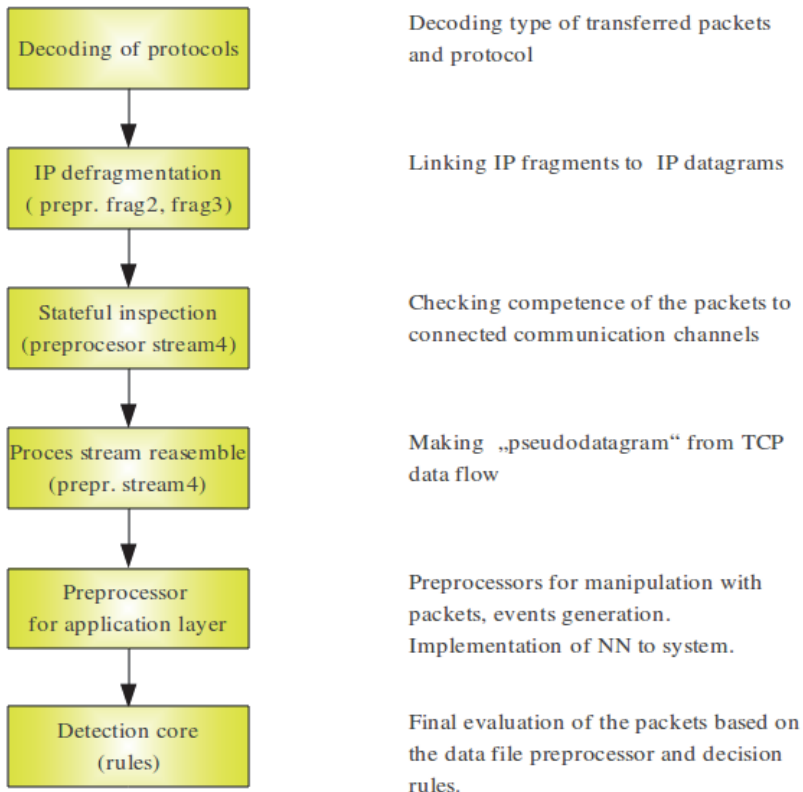
**Fig. 9** Architecture of Snort IDS

The application itself is a variable tool. It allows working in IDS regime, regime of grabbing packets or IPS mode. For us it is a host application for the module proposed.

The ability to implement their own evaluation modules of the network operation is available in the part of a *pre-processor*. By a suitable complementary module, it is possible to investigate and evaluate anomalies within the network communication, and subsequently shift the anomalies found as an input to the detection system with pre-set action for the packets in question.

There are three practical regimes of the application mentioned. It can work in the following three modes:

- *sniffer* mode – by which the data of the monitored communication are displayed in the screen, or possibly stored in a file. This operation regime is suitable for neural network learning or for data acquisition for neural network learning. In the simulated regime we can use the stored data for multiple simulation of the network system operation. Activation of the system is possible by the order on the operator's console  
*/snort -vde*      (*./snort -vde -l \$HOME/log – with storing*)
- NIDS mode – basic operating mode, by which the monitoring of the system network operation with the reaction based on the set of defined rules is carried out  
*./snort -vde -l \$HOME/log 147.175.133.0/24 -c snort.conf*
- *inline* mode – for the cooperation with firewall and *IP tables* filtration system. (63).



**Fig. 10** Packet processing algorithm

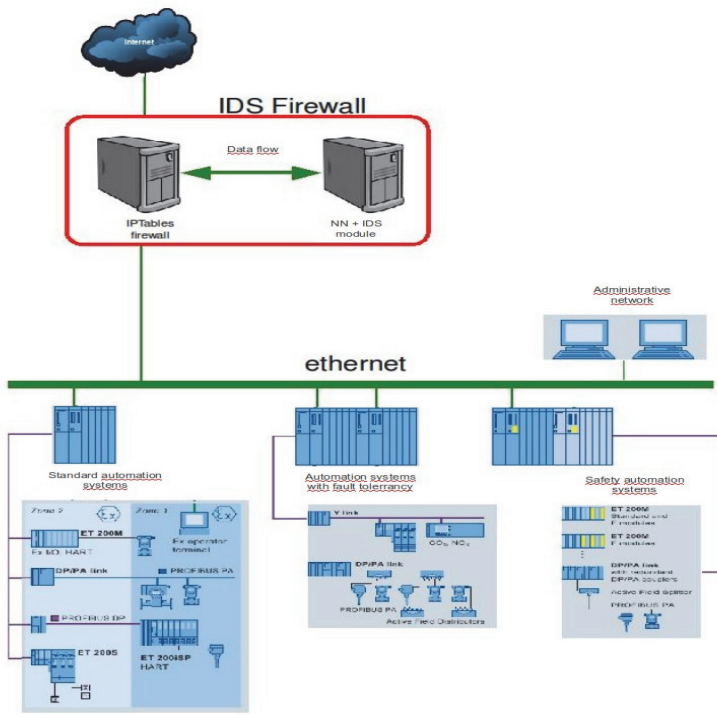
For the intended utilization, all operation regimes are important. The modes mentioned in points one and three are, however, the most significant ones. In the *sniffer* regime they are suitable for the development of sets for learning as well as for neural network testing, and in *inline* regime, for active control of communication data transfer.

The packet is the basic information unit in the proposed system. It enters the system through the interface of communication monitoring (*sniffer*), which captures the packets and ends them for further processing. Monitored data in the network can be redirected to the detection core together with the pre-processors, or into the database on the disc for later testing or possible analysis.

The analysis is represented by an insert module of the neural network (pre-processor). In this case, when an anomaly in the monitored data flow is detected, the output from the NS module is represented by a relevant announcement. By this core action, the communication can be blocked via *IPtables* filter kernel, or a constant rule can be defined in the expert database of rules.

The entire detection system scheme comprises two basic components - IDS + firewall. The following figure illustrates a practical application of the proposed system (*Fig. 8 NN system design*) into the area production environment (*Fig. 11 Implementation in real environment*).

A suitable system platform able to connect all the necessary software modules is the prerequisite of the function in the detection system of this kind. Noting that it is a specialized, dedicated system, a high degree of modularity and the ability to modify the program functionality of the entire system are necessary. Therefore, it is suitable to develop the system on UNIX platform, Debian Linux in particular. This system also provides the possibility of data network penetrability control via *IPtables* core module (82). The module can be inserted as a module into the system core.



*Fig. 11 Implementation in real environment*

We can summarize the configuration of the required software means as follows:

- operation system: Debian Linux, kernel 2.6.28-16-generic
- host IDS: Snort 2.8.5.1
- data flow system control: Netfilter IPTables

### ***5.2.1 Determination of parameters for transfer anomalies detection***

Anomaly in the data network operation is represented by a situation in which the operation communication parameters deviate from the standard behaviour of the transport infrastructure. Network anomalies can occur for various reasons: either as physical malfunctions, e.g. network, or as the result of unauthorized network intrusions disturbing the standard transfer conditions. The method of transfer misuse known as “*Misuse detection*”, the transferred packets is described by an unambiguous pattern (signature) (44). Signatures suitably identify the data transferred and, regarding their occurrence, we can define the data flow correctness. For correct validation of data transfers the monitoring of individual fields’ values in TCP and IP protocols is primarily appropriate; this can simply identify the packets correctness.

The data packet content can also be included in the main identification parameters. In the data, we can find misuse signatures for ISO/OSI Model higher levels, either random misuses ( e.g. misuses in data blocks, incorrect control orders for devices), or targeted misuses (e.g. attempts to control devices in the network, unauthorized data acquisition, DOS attacks, etc.). Because of the huge amount of used protocols and sub protocols and their combinations, it is suitable to choose determining parameters by the selection of data packets parameters. Considering the preliminary analysis of possibilities found in references (51) as well as regarding the results of (35) and (36), we can propose that in this case we will specify the data packets properties by the following parameters:

- ID protocol - protocol type connected with a packet
- System source port - TCP/UDP number of source system port
- Destination port - TCP/UDP number of destination system port
- Source address - IP address of source system
- Destination address - IP address of destination system
- ICMP type - ICMP type of a packet
- Length of transferred data - size of packet data in bytes
- FLAGS setting - flags in protocol heading
- TCP window size - window size parameter

### 5.3 Selection of neural network algorithm

The neural network is the main core of the operation as well as it is the core of the entire system. Despite little research in the field of neural networks utilization in data transfer misuses detection, there are these practical applications (1, 35, and 46). It is mainly an effort to replace the statistic methods commonly used for anomalies evaluation in data transfer by a neural network. Further utilization of neural networks in practice is described in publication (49), where it is used to detect virus attacks on computer systems. A Kohonen network with one input layer is the type of neural network here.

With the application and selection of a suitable neural network type, it is necessary to realize the implementation of any neural network has both advantages and disadvantages. The flexibility of neural networks is an



advantage; in addition, the neural network is able to analyse incomplete data. Non-linearity of data flows in communication networks is another aspect influencing the selection. Since the neural network output is expressed as probability, neural network outputs can subsequently work as a certain prediction. Because neural networks can improve their abilities by learning, the output information could then be used to generate various actions in cases where a prediction is an alert of an attack attempt.

The disadvantages of a neural network application are basically identical as those of other applications. Particularly, the issue of a “black-box”(7), where we cannot determine the network’s behaviour, as well as its output precisely, is a disadvantage. In contrast to classical expert systems, for neural networks, it is essential to set the scaling of synapses. The network accuracy cannot be determined in the course of learning. Only after the process of learning stops is the network able to produce significant results. The process of a network’s learning represents another disadvantage of neural networks application. To train the neural network, a large set of training data is necessary. In this case, the training data represent an amount of sequences of data packets attacks on communication systems. The set should be selected properly in order to represent the needed functions statistically. Nevertheless, it is quite demanding to acquire such a set of data.

Research in the field of neural networks application by anomalies detection in data transfers/ transports is carried out in two main directions, or we can say that two basic types of neural networks are utilized for the purpose. Regarding the sources (44) we can consider a neural network with or without controlled learning. In references (52), (53), and (54) the

possibilities of applying *Self-organized maps* are described (Kohonen neural network). According to the aforementioned sources, such networks achieve more balanced results. They can also detect unknown misuses; however, their susceptibility to fail in comparison to other network types in the number of known anomalies detections is higher. From the point of view of time demands, it represents an algorithm, which is quite time-consuming in the process of structure learning. By changing the input vector, all the scaling data should be recalculated together with the determination of new winning neurons.

Neural network with controlled learning represent another neural network type able to be utilized. The feed forward neural network is the basic network type. The architecture is simple and suitable to be used in various applications.

Key properties are as follows:

- possibility of the network to learn from input signals for required output (controlled learning),
- easy adaptation to input values change,
- necessity of a lower amount of calculations (in contrast to SOM network).

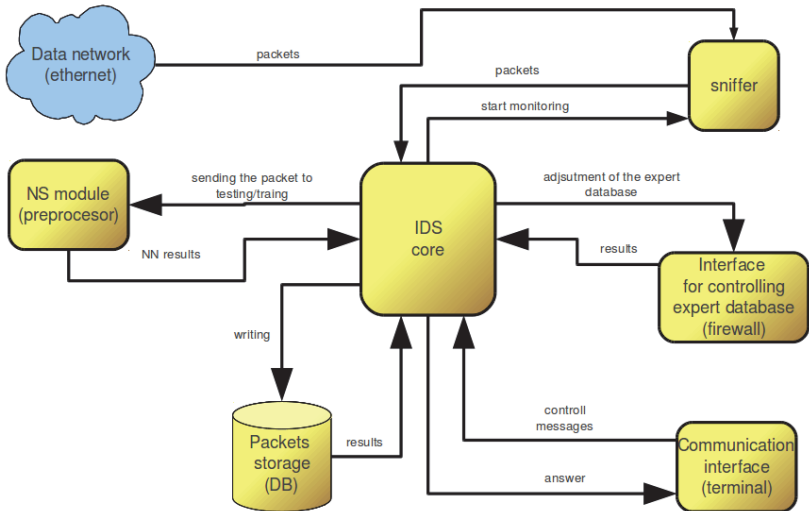
There are many types of internal arrangements of neural networks, either with or without controlled learning. According to sources available and utilization required, we select a **free-forward neural network** with learning with backward spread of misuse. This neural network type provides sufficient flexibility and applicability for a large scale of tasks, where NS technologies can be utilized.

## 5.4 Implementation proposal and learning of network

The propose system of non-standard data packets detection consists of three basic parts:

1. neural network module,
2. host IDS system providing interface and support to system modules,
3. expert database of rules in connection with the system so that it can actively intervene the data frameworks transferred/transported.

The block scheme of the entire system can be illustrated as follows:



*Fig. 12 Block scheme of the proposed system*

The data packet, entering the system through interface of the device for monitoring the data transfer, is the basic transfer element of the system proposed (sniffer). Interface (eth0) record the packets and sends them for further processing. For recording the packets, the data interface has to be switched in a promiscuous mode. In a subsequent step, the interface will send the packet either to the neural network module, or, for recording of the packet, given in local hard disc database.

#### ***5.4.1 Application equipment for neural networks implementation***

The issue of the proposal and implementation, or possible neural networks simulation, is covered by a large scale of application equipment. The majority of software is taken as a simulation program, in which neural network development is possible. Subsequently, based on the inputs and outputs, the learning process is carried out by setting and recording the synaptic scaling parameters. There are many applications of the kind, for Windows platform we can mention Matlab & Simulink (96), MemBrain NN Editor + Simulator (98). Stuttgart Neural Network Simulator is perhaps the best known representative (99). For other platforms (UNIX, Mac OS X) we can consider e.g. GENESIS Simulator (100).

All mentioned programs are suitable for proposing and testing the parameters. In the case of practical implementation of the model, however, we have to often utilize the proprietary implementation using a suitable programming language. Regarding the efficiency and possibility of source program transfer/transport for various platforms, it is mostly C Language in various modifications based on expected final implementation.

Considering the expected utilization, for the proposal and modelling we shall come out of the file of libraries for neural networks simulation available in the form of source codes in C++ language, gcc 4.1 and Tcl/Tk 8.4. Utilized source codes are freely available in a student version on the website (102).

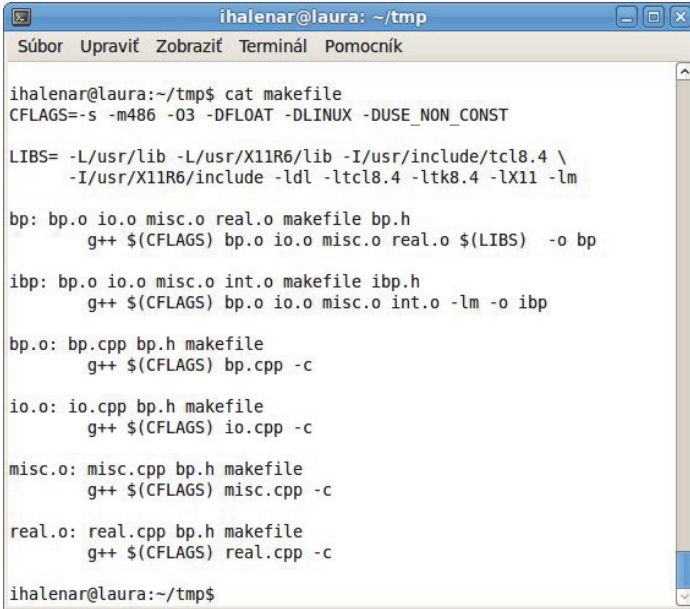
The possibility of implementing the libraries by the functions for neural networks operation in their own program is an advantage. On the other hand, compilation demands and following development of the source code are disadvantages. Nevertheless, for the proposed implementation it is necessary (*Fig. 12 Block scheme of the proposed system*). In this case, the neural network will be part of the whole system of monitoring and data transfer control.

The package of software available (102) comprises the following files of source codes and libraries:

- bp.cpp – represents the main program and program routines for developing neural networks data fields
- bp.h – functions for the program for working with scaling in the floating comma mode
- io.cpp – input-output and formatting functions
- misc.cpp – functions for manipulation with patterns
- real.cpp – procedures for operating the scaling values in floating comma mode
- bp.tcl – framework for graphical infrastructure.

For compilation, we utilize standard system libraries and C Language compiler (GNU Compiler Collection) (103) available directly in

depositories of the Linux Debian operation system, particularly gcc in version 4.x.



```
ihalenar@laura: ~/tmp
Súbor Upraviť Zobrazíť Terminál Pomocník

ihalenar@laura:~/tmp$ cat makefile
CFLAGS=-s -m486 -O3 -DFLOAT -DLINUX -DUSE_NON_CONST

LIBS= -L/usr/lib -L/usr/X11R6/lib -I/usr/include/tcl8.4 \
      -I/usr/X11R6/include -ldl -ltcl8.4 -ltk8.4 -lX11 -lm

bp: bp.o io.o misc.o real.o makefile bp.h
    g++ $(CFLAGS) bp.o io.o misc.o real.o $(LIBS) -o bp

ibp: bp.o io.o misc.o int.o makefile ibp.h
    g++ $(CFLAGS) bp.o io.o misc.o int.o -lm -o ibp

bp.o: bp.cpp bp.h makefile
    g++ $(CFLAGS) bp.cpp -c

io.o: io.cpp bp.h makefile
    g++ $(CFLAGS) io.cpp -c

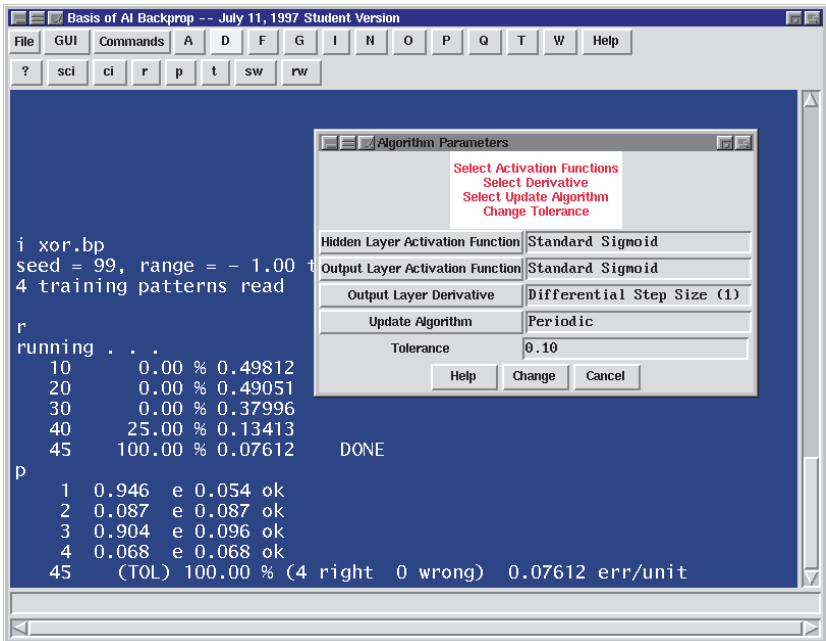
misc.o: misc.cpp bp.h makefile
    g++ $(CFLAGS) misc.cpp -c

real.o: real.cpp bp.h makefile
    g++ $(CFLAGS) real.cpp -c

ihalenar@laura:~/tmp$
```

*Fig. 13 Makefile for gcc*

It is possible to compile the program either in text version, where the operation runs in the order line through the standard input (stdin), or compile in graphic version using TCL/Tk framework is possible as well (104). Installation packages Tcl/Tk in version 8.4 are commonly available in the system depositories. After the compilation with Tcl/Tk package and using „bp.tcl“ file , the system for modelling looks as follows:



*Fig. 14 Communication interface for NN design*

### **5.4.2 Preparation of input data**

The acquisition of input data for neural network learning and testing is one of the most important steps. There are more ways to obtain an appropriate sample of data sufficiently representing the transport network relations.

First, in the long term, we can monitor the operation in communication and control data network. From these data we can filter a sample sufficiently representing the properties required by us (various malfunctions/misuses, type of transfer packets, various services, etc.) They

are real data of an existing communication system, which is an advantage. On the other hand, time consumption and the fact that even after a long time no necessary transfer packets types occur in the monitored network at all is a disadvantage.

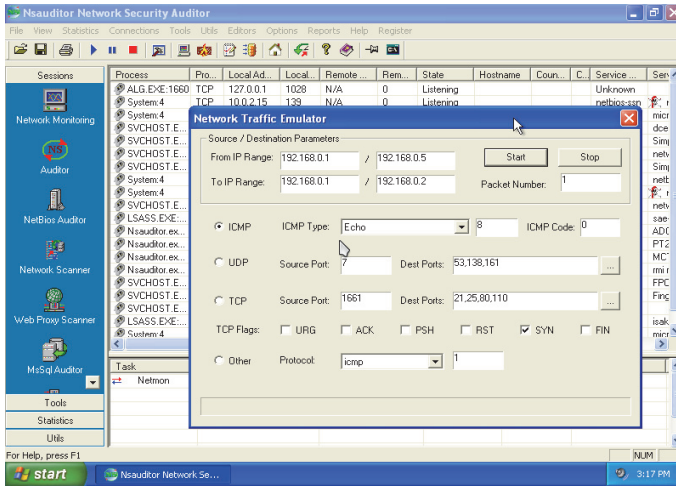
Development of communication network model, in which we utilize an independent system as a data operation generator is another possibility. We can generate multiple required states and we are not bound to random processes in real operation as in the first possibility, which is an advantage. Nevertheless, this is an artificial process different from the real operation in real communication network.

The third possibility is to obtain examples of recorded communication from on-line sources. (104). In this case, they are real records from real production systems. It is possible to “replay” these records by a specialized software (pre unix - tcp replay) in our network so that it seems it is a malfunction/misuse (or possible security failure) directly in the home network. However, they have to accommodate the local network parameters and regarding the differences in transport networks structure the utilization of the entire database available is limited.

### **Generating input data**

For data acquisition in neural network experiments we use the combination of the first and second possibility as aforementioned. For generation, I selected the system by Nsasoft Company called NS Auditor (105). The application operates with various functionalities, such as network audit, network monitoring, netbios audit, MS SQL server audit, packets filtration, etc.

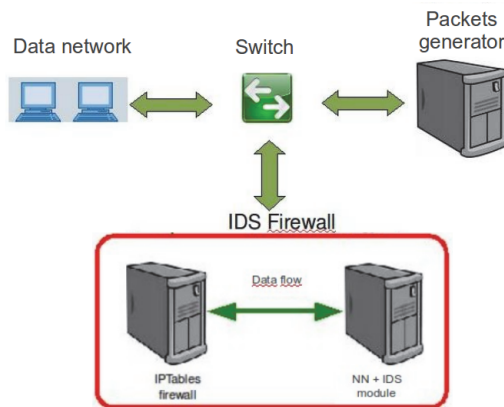




*Fig. 15 Data flow generator*

In our testing environment (*Fig. 16 Recording of packets*) we utilize it mainly for data flow emulation from the specific source to the destination address in a specific port, as well as for the emulation of non-standard network communication.

Simultaneously, with data communication generation, the data operation record module will be integrated and the output will be redirected to the packets database on the hard disc. Since at the same time the testing system will be connected to the external network, it will result in the database comprising standardized network communication, real network communication as well as random generated special events from the emulator.



**Fig. 16** Recording of packets

After storing the record of data communication into a database, it is necessary to extract the aforementioned values from the sum of data. Regarding that the activity should be automated as much as possible, it is suitable to develop a software conversion bridge that would be able to supply suitable input data directly into the neural network.

Data modification should be executed in stages. Step one is to extract required values from the entire data package. Then we can afford to ignore the service communication records (ARP, RARP communication) as well as the irrelevant data. The following step lies in an appropriate representation of some elements.

Particularly, they are the following parameters:

ID\_PROTOCOL, where we carry out the following transformation:

ID_PROTOCOL	Substitution
TCP	6
UDP	17
ICMP	1

TYPE\_ICMP, where we carry out the following transformation:

TYPE_ICMP	Substitution
ECHO	1
REQUEST	2
NULL	0

FLAGS, where we carry out the following transformation:

FLAGS	Substitution
No_flags	1
RST	2
FIN	3
PSH	4
URG	5
SYN	6
Other	7

GNU AWK script language was selected as a suitable programming language for operation with chains (106). It is a standard language on the UNIX/LINUX platform, and it is located directly in the installation repositories. The implementation into the system is executed by a simple order:

```
sudo apt-get update && apt-get install gawk
```

From the volume of data monitored for the proposed system we can select only ICMP, TCP and UDP packets. It is essential to carry out the whole data transformation process as simple and fast as possible. Therefore, we implement programs for AWK in executable script for system Bourne Shell. By simple redirecting of data flows within the system we can subsequently process huge volumes of the communication recorded.

### **ICMP transformation (using AWK)**

```
BEGIN {fill = 0 }
{
if ( index ($0,"(17)") ) next
else
dlzka = $NF
typ = substr($15,2,1)
split ($18,za,".")
split ($20,ca,".")
if (index ($0,"") ) sluzba = 0
if (index ($0,"echo") ) sluzba = 1
if (index ($0,"unreachable") ) sluzba = 2
split (ca(4), cabd, ":")
```

```

    printf("%s,%s%s%s%s,%s,%s%s%s%s,%s,%s,%s,%s,%s\n",typ,
za(1), za(2), za(3), za(4), fill, ca(1), ca(2), ca(3), cabd(1),
fill, fill, fill, dlzka, sluzba)
}

```

## TCP transformation (using AWK)

is executed similarly as ICMP.

```

BEGIN {fill = 0, conv_flag = 7}
{ i= 1
while ( i < NF) {
hodnota = index ($i, "win")
if (hodnota == "1") okno = $(i+1)
++i
}}
{ split ($17,dlzka,"") )
typ = substr($15,2,1)
split ($18,z_a_p, ".")
split ($20,c_a_p, ".")
split (c_a_p(5),cport, ":")
split ($21,flags, ",")
if (index (flags(1), ".") conv_flag = 1
if (index (flags(1), "R") conv_flag = 2
if (index (flags(1), "F") conv_flag = 3
if (index (flags(1), "P") conv_flag = 4
if (index (flags(1), "URG") conv_flag = 5
if (index (flags(1), "S") conv_flag = 6
printf("%s,%s%s%s%s,%s,%s%s%s%s,%s,%s,%s,%s,%s\n",typ,z_a_p
(1), z_a_p(2), z_a_p(3), z_a_p(4), z_a_p(5), c_a_p(1),
c_a_p(2), c_a_p(3), c_a_p(4), cport(1), okno, conv_flag,
dlzka(1), fill)
}

```

## TCP transformation (using AWK)

```
BEGIN {fill = 0}
{ split ($17,dlzka,"" ) }
{ typ = substr($15,2,2) }
{ split ($18,z_a_p,".")}
{ split ($20,c_a_p,".")}
{ split (c_a_p(5),cport,":")}
{printf("%s,%s%s%s%s,%s,%s%s%s%s,%s,%s,%s,%s,%s\n",typ,z_a_
p(1),z_a_p(2),z_a_p(3),
z_a_p(4),z_a_p(5),c_a_p(1),c_a_p(2),c_a_p(3),c_a_p(4),cport(1),
fill,fill,dlzka(1),fill)
}
```

Such a way of data extraction from the entire database file of stored network communication is essential for the fast generation of necessary data for neural network. Optimally, it is possible to achieve automated network learning without direct intervention. By unification of all three transformations we get a table, which should be extended by one parameter important for training learning of the neural network (class).

TABLE OF INPUTS FOR NN

Table 1

ID PROTOCOL	SOURCE IP	SOURCE PORT	TARGET IP	TARGET PORT	SIZE OF TCP WINDOW	FLAG	PACKET SIZE	ICMP TYPE	CLASS
1	1471751309	0	147175134254	0	0	0	44	1	1
6	62240183148	80	147175134254	2190	8514	4	1500	0	1
17	19416092	33239	233104778	65535	0	0	1500	0	2
6	147175130212	2246	19512215120	80	65535	1	40	0	1
17	13015680151	4262	2242127254	9875	0	0	224	0	1
1	147175130212	0	10254247189	0	0	0	36	2	2

Regarding the supposed neural network utilization as a classifier, it is sufficient to classify the patterns into two groups. In this case, it will be enough to divide the packets into correct ones (CLASS = 1) and those that will be subject to further testing (CLASS = 2). This is subsequently executed by the file of expert rules (*Fig. 12 Block scheme of the proposed system*). It is evident that we can develop another neural network module, which will operate only with incorrect packets. Or, possibly we can extend the neural network classification into more classes. Nevertheless, regarding the variability of input data and investigated issue complexity, this approach would represent a more complex proposal extremely demanding in terms of a training set and processor calculation time. Finally we convert the tables into a format suitable as a neural network input. Since it is a server system

operating on a terminal console and taking the operation speed and simplicity into account, they will be text values separated by a space.

```
1 147175128234 0 255255255255 0 0 0 8 1 1
17 955215335 15994 14717513511 35420 0 0 61 0 1
1 14717510832 0 14717512943 0 0 0 44 1 1
6 14717513337 43066 147175116158 41981 266 4 362 0 2
6 14717513337 43066 147175116158 41981 266 1 52 0 1
17 92557475 21608 147175131126 24854 0 0 90 0 1
1 14717510832 0 14717512925 0 0 0 44 1 1
17 10254244120 138 10254255255 138 0 0 229 0 1
6 24212211 50178 147175141249 20437 8192 6 48 0 2
```

### ***5.4.3 Neural network implementation***

Network architecture represents the connection of neurons and setting of scaling and threshold parameters. Considering the sources studied (7), the neural network with one covered layer (3-layer) and a sufficient number of covered neurons is always able to simulate binary or continuous functions with the accuracy required. In our case we consider nine input and two output neurons of the covered layer. Solutions available in sources, however, are quite unclear and complicated for practical use. I tried more network types empirically with variously sufficient results.

Data transformation selected 5000 patterns from the running communication in the testing environment. Four groups comprising 200 patterns for network learning and groups comprising 100, 200 and 3000 patterns for testing were selectively made of the amount. This division was necessary due to experimental verification of achieved results for neural



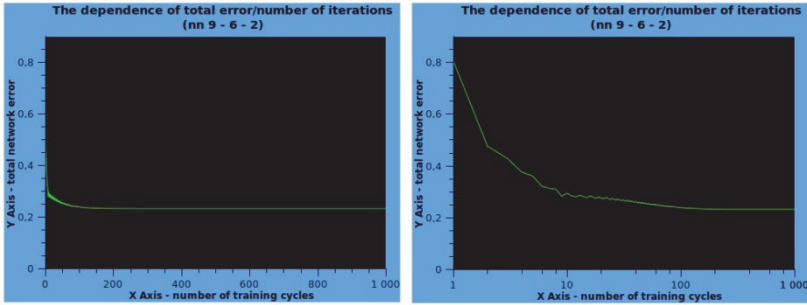
networks with various numbers of neurons in the covered layer and different activation function for covered and output layers. Standard sigmoid activation function was used in all cases as a covered layer activation function:

$$x_i = f(in_i) = \frac{1}{1 + e^{-cin_i}} \quad [4]$$

Besides the sigmoid function also the linear activation function was used in various experiments as an activation function of output neurons:

$$x_i = f(in_i) = in_i \quad [5]$$

The same networks with other activation function on the output layer had different results, which resulted in a disadvantage for utilizing the linear activation function. As an example, we can show the following graphs (*Fig. 17 Learning NN 9-6-2 in linear and logarithmic scale*), where the introduced network is trained in the same conditions as the network in figures (*Fig. 18 Learning NN 9-6-2 in linear and logarithmic scale*). The only difference is that the linear function was used as an activation function of the output layer. After 1000 iterations the overall neural network error achieved the value of 0.23104. It is a stable value not changing after approximately 200 training cycles.



*Fig. 17 Learning NN 9-6-2 in linear and logarithmic scales*

Similar differences occurred in the other experiment (network with 10 neurons in the covered layer - *Fig. 19 Learning NN 9-10-2 in linear and logarithmic scale*). It showed that, in general, in the case of neural network proposals, higher quality results with sigmoidal activation function in the output neuron layer were achieved.

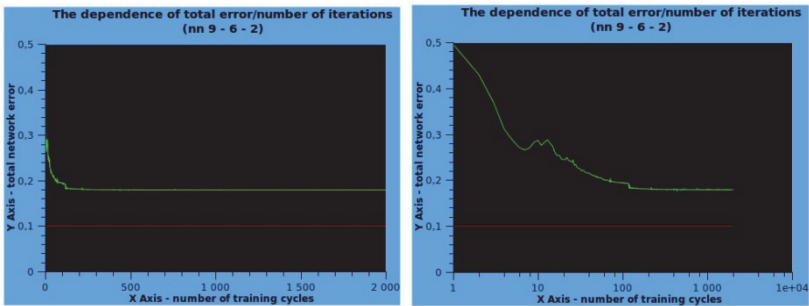
The quickprop method based on Newton method was selected as an algorithm of the purpose function minimization (75). The method is one of the fastest algorithms for neural networks learning.

Subsequently, we represent the final results achieved for various number of covered layer neurons. For better reader friendliness of results the network will operate in the classifier regime (CLASS = 1 or 2), network output will be in the form (0 1) – group one or (1 0) – group two. In result outputs the sigmoidal function for covered and output layers is used as an activation function.

## Neural network 9 – 6 – 2

A file of 60 patterns modified for the required format input was used as a training set. The results achieved are insufficient. Even the change of network parameters (such as output function, changed initialization scaling values, various coefficients of learning parameters, etc.) do not help the network achieve the required accuracy for the result (maximum deviation 0.1). Network training was stopped after 10,000 cycles.

For better illustration, the outputs achieved are shown in the following graphs.



*Fig. 18 Learning NN 9-6-2 in linear and logarithmic scales*

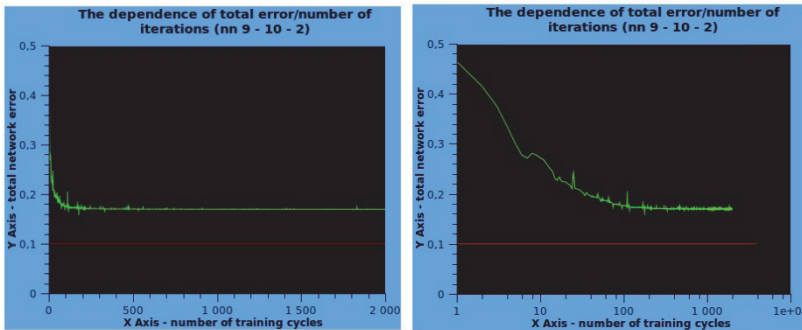
For better illustration the graphs comprise ly values up to 2,000 cycles. The graph shows that approximately after 1,000 iterations the levels of synoptic scaling are stable. The value of the training set absolute deviation is stabilized at 0.17905. The number of patterns meeting the required tolerance is 66.67%.

By contrast, the following network testing by the testing patterns group showed quite good values of object separation. The total number of

patterns in the testing group was 100; the number of samples meeting the given tolerance (0.10) in the testing group is 91%.

### Neural network 9 – 10 – 2

Similarly as the previous, the file of 60 patterns modified for the required format input was used as a training set. The achieved results after 10,000 iterations correspond to the values of the absolute malfunction/deviation values 0.1693, while the number of training set patterns rose to 68.33%. Nevertheless, the values are insufficient. Similarly, by the network parameters change (such as output function, changed initialization scaling values, various coefficients of learning parameters, etc.) do not help the network achieve the required accuracy in the result (maximum tolerable deviation 0.1). The network training was stopped after 10,000 cycles. And similarly, as in the previous case, for better illustration we introduce the outputs achieved in the following graphs.

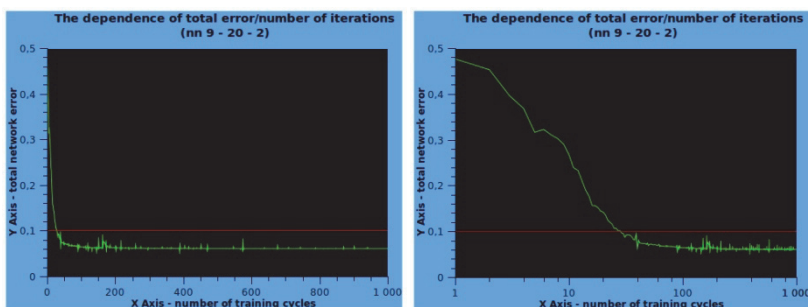


*Fig. 19 Learning NN 9-10-2 in linear and logarithmic scales*

## Neural network 9 – 20 – 2

In this case, the training set is the same as in previous cases, i.e. the file of 60 patterns modified for the required format input. In contrast to other networks with lower number of covered layer neurons, this network achieves very good values. The achieved results after 10,000 iterations correspond to the values of the absolute deviation 0.06041, while the number of the training set patterns achieving the required tolerance rose to 96.67%.

Even by the higher number of iterations the networks training did not show better results the 1,000 iterations mentioned. The achieved values of malfunctions/deviations in this case meet the requirements for detection parameters. The outputs are shown in the following graphs.



**Fig. 20** Learning NN 9-20-2 in linear and logarithmic scales

The testing group of patterns is identical with the previous ones and the number of patterns is 100. The number of samples in the testing group which met the given tolerance (0.10) is 98.56%. From these results, it is obvious that the best classification ability of the training sample is achieved by the neural network with the number of 20 neurons in the covered layer.

The experiment to apply the linear activation function for the output layer showed approximately similar deviations in the network decision quality as in previous cases. Regarding that the required accuracy was achieved, we can say that the results introduced as well as the neural network proposal, are suitable for the implementation into the system proposed. The arrangement of the neural network with 20 covered layer neurons is considered to be final.

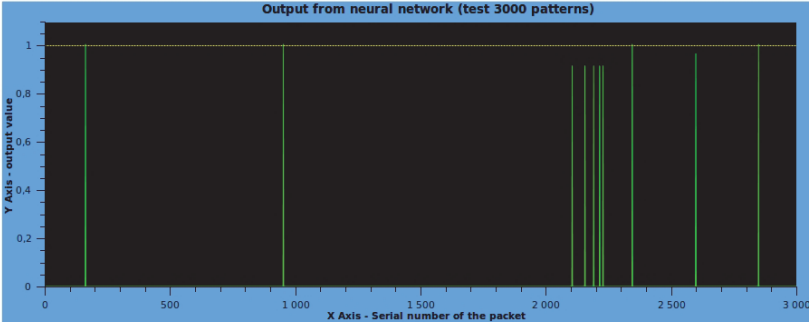
## **5.5 Experimental system verification**

After the end of neural network training, the values of synoptic scaling connections were stored and common data introduced to the network input.

### ***5.5.1 Testing of proposed neural network***

Testing file of patterns comprised 10 incorrect packets, particularly in the first case with the experiment of communication by non-standard TCP ports.

For the experiment, I used the sample of size of 3,000 data packets. Formatting of input samples was modified according to required parameters. The success of incorrect packets recognition achieved 100%.



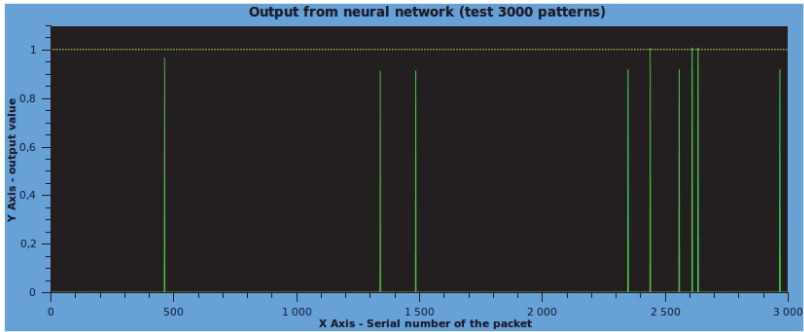
**Fig. 21** The output from neural network (non- standard TCP ports)

Representation of the incorrect packet is executed by the last layer neurons output. The following graphical illustration basically represents the file of the second neuron values in the third layer (N3-2). In relevant ordinal packet numbers, the first neuron gains opposite values. The specific output values for the samples are shown in the table.

NEURAL VALUES OF OUTCOME LAYER (TEST1) Table 2

Neuron	Packet number									
	163	953	2105	2156	2191	2215	2228	2345	2599	2849
N 3-1	0.000	0.000	0.095	0.095	0.095	0.095	0.095	0.000	0.030	0.095
N 3-2	1.000	1.000	0.905	0.905	0.905	0.905	0.905	1.000	0.96	0.905

In the other case, the testing file comprised the same number of 10 incorrect packets, here the combination of the parameters of tcp window non-standard size (smaller than 1028) and incorrectly introduced combination of SYN and FIN flags (FLAGS=7).



**Fig. 22** Output from neural network (non-standard combination Winsize and Flags)

For the experiment I used the same size sample – 3,000 data packets. Formatting of input samples was modified according to required parameters. The success of incorrect packets recognition in this case achieved 90%. Specifically, the packet with ordinal number of 895 was incorrectly recognized by the neural network. The results, however, are legible and the deviation mentioned is within the tolerance (0.1). It is possible to eliminate the shortcoming by improved neural network training, or by better selection of patterns in the training set.

NEURAL VALUES OF OUTCOME LAYER (TEST 2) Table 3

Neuron	Packet number									
	464	895	1342	1486	2350	2441	2559	2611	2635	2968
N 3-1	0.030	1.000	0.095	0.095	0.095	0.000	0.095	0.095	0.000	0.100
N 3-2	0.96	0.000	0.905	0.905	0.905	1.000	0.905	0.905	1.000	0.91



The results show that the given neural network is able to recognize the data packet types in the control and communication networks it was trained for with sufficient accuracy. The errors in values represent a certain deviation from the values required; however, the information is sufficiently legible.

Regarding the achieved results, we can say that the implemented system of data packets identification allows detecting random data transferred in the communication network. As seen in graphs *Fig. 21* and *Fig. 22*, the type of detected parameters of transfer packets depends on the selection of the training set. Since for the generation of the set we use software in which transfer parameters of transformation so that they are usable as neural network input data, the development and learning of the proposed neural network is fast and efficient.

### ***5.5.2 Possibilities of practical implementation***

The entire neural network proposal considers the conclusions mentioned in Chapter 5, particularly in 5.1. Practical implementation lies in the development of the complementary program object for the host application. In the case of the mentioned application, it is a pre-processor including the functions for the operation with the proposed neural network. Since the entire neural network is available in C++ Language and the standard programming tool for the pre-processor of the mentioned host application (90) is the same language, the practical implementation is in the utilization of available programming tools. In this case it is the same environment as for the programs compilation of the developed neural network, particularly compiler of C Language (GNU Compiler Collection)

(103) available directly in depositories of Linux Debian operation system, in gcc, version 4.x.

Functions representing the neural network operations are available in the appendix in source codes “bp.c”, “ibp.h” (for operation with 16 bit scaling values) and “rbp.h” for operation with real scaling values. The libraries are based on the source texts mentioned in References (101).

To develop a pre-processor with the mentioned libraries (functions) we utilize the templates available in (63), (64).

The implementation of the proposed system (*Fig. 12 Block scheme of the proposed system*) assumes besides IDS modules with installed pre-processor for incorrect packets detection via neural networks also the utilization of an external state packet filter, i.e. classical state firewall. Practical implementation of the device is represented by a server with installed *Iptables* component (63) and by interface for dynamic control of expert database of LIBNET rules (63), (107).

## **6. CONCLUSION**

This contribution provides a solution to the security of control systems data communication via neural networks technologies in connection with classical methods used in expert systems. As mentioned in the monograph, the issue of control systems is a complex process, which, regarding the openness and interoperability of transport channels and communication systems, is subject to the surroundings' influence. The proposed methodology of the implementation of data communication control system defines the identification of data elements in the transport network, solves the transformation of their parameters for neural network input as well as defines the type and architecture of a suitable neural network. This is supported by experiments with various types of architecture and neural networks activation functions and subsequently by tests in a real environment. The result is a functional proposal with possible practical applications. The determination of parameters for unambiguous data packets identification, proposal and implementation of these data transformation for neural network input are the most important monograph benefits. The proposal, verified by experiments and subsequent neural network implementation, represents the suitable way of this neural network type utilization in dealing with the issue.

The monograph comprises the analysis results of the subject matter (Chapter 3) and analysis of the current state of the issues in the researched field (Chapter 2); the main aims were defined (Chapter 5) as well as the research itself. Regarding the complexity of the proposed solution, I divide the monograph into individual sub-issues.

The first stage of the solution lay in the proposal of the system structure comprising functionalities necessary for the solution. In our case, we propose a modular system, in which the host IDS Snort application is utilized. The application proposed is modular, open (from the point of modules development) and in the proposal implementation, we are supported by the environment provided. The solution, in which we integrate the neural network technologies in the proposed system, is the development of the module in the part of a pre-processor. The overall integration of the proposed system in the control system communication network in practice is illustrated in the figure *Fig. 11 Implementation in real environment*.

The second stage of the solution is the specification of parameters for transfer anomalies detection. Communication of systems in data networks lies in the data exchange among individual elements, specified transport protocols. The analysis of data elements parameters and determination of unambiguous identifiers is introduced in Chapter 5.2.1. For this utilization, it is considered necessary to excavate the list of parameters so that they describe the running transfer sufficiently. The mentioned process of parameters selection serving as the input to the proposed neural network is then elaborated in detail in the Chapter 5.4.2.

The proposal of neural network module type and architecture suitable for determined parameters detection in the communication flow is the essential part of the monograph (Chapter 5.4). Regarding the wide scale of neural networks architecture types, the selection of a certain specific type is quite complicated. I came to the decision on the basis of the sources studied and partially on the basis of my own experience. The neural network topology selection (Chapter 5.3) with feed forward neural network with

error back-propagation algorithm finally showed as a solution suitable for the application. The topology selection is directly related to the development environment and application equipment essential for the detection system function. The description of application equipment parameters for the module development is the basis of the chapter.

The issue of the training and testing data group and subsequent solution to the selected data elements conversion is described in detail in Chapter 5.4.2. It is possible to utilize more ways to simulate and generate incorrect packets. From all the possibilities mentioned, I selected the combined solution, which is the combination of real communication in a real network and random generation of non-standard data packets. Such simulated communication was then stored in the database on a local device. To convert the recorded data for module input of the neural network, as well as for easy and fast development of the training and testing set, it was necessary to develop software operating as a conversion bridge.

After developing and recording of needed data, the last step of neural network implementation follows. Experiments and simulation were carried out in the environment of available simulation programs. The proposal developed more neural networks with a different number of neurons in the covered layer and similarly different activation functions. The results of the experiments are illustrated in graphs in Chapter 5.4.3.

The subsequent verification of the proposed solution showed (Chapter 5.5), that the proposed and trained neural network is able to sufficiently classify individual data packets within the determined tolerance. This proved the possibility of artificial intelligence technologies application in

solving matters related to the verification of data transfers in control and communication networks.

The final test of the proposed network was in testing two samples, where each of them comprised 3,000 data elements. The results show (Chapter 5.5), that the parameter type in the data flow, or possibly in the anomaly to be detected by the system, depend only on the development of a suitable training set and subsequent neural network training.

## **6.1 Possibilities of further development**

Regarding the overall extent and complexity of data transfers/transports in control and industrial data networks, this monograph and its results represent only the introduction to investigation in the field. The following points can be subjects of further research:

- Propose significantly larger neural network that will be able to classify incoming packets into more classes. Then it would be possible to distinguish the transferred data and subsequently react to detected anomalies with higher accuracy. Obviously, this step assumes the development of a significantly larger training set of samples and extension of conversion tables mentioned in the monograph as well as other elements.
- Develop software able to generate the training set automatically on the basis of requirements and ensure the whole learning process (training with required accuracy, scaling arrangement, activation of network with new parameters).
- Improve the selection of packets sets for neural network learning from the recorded operation by the implementation of database

system via data stores technologies (39) and data mining in data communication records.

- Extend the mentioned model by other neural networks, i.e. by more pre-processors able to detect further anomalies in data flows, e.g. the quantification of the transferred/transported data after Fourier's transformation (covert channel detection), or possibly other neural networks module applicable for the preparation of training data set for the main detection neural network.
- Examine the application of other neural network types in the model introduced, e.g. the application of mentioned Kohonen network, or possibly other network type without controlled learning.

## REFERENCES

1. TAN, K. The Application of Neural Networks to UNIX Computer Security. In: *Proceedings of the IEEE International Conference on Neural Networks, Vol.1*. Melbourne, 1995, pp. 476-481.
2. CONTI, G., KULSOOM, A. Passive visual fingerprinting of network attack tools. In: *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security table of contents*. Washington DC, USA: ACM, Washington DC, 2004, pp. 45-54.
3. CRAIG, V., NIRBHAY, G. *Honeypots*. (on-line). [cit. 2009-08-20]. [http://scissec.scis.ecu.edu.au/publications/2003\\_VALLI\\_GUPTA\\_IW\\_AR2003\\_honeyd.pdf](http://scissec.scis.ecu.edu.au/publications/2003_VALLI_GUPTA_IW_AR2003_honeyd.pdf)
4. SPITZER, L. *Definitions and Value of Honeypot*. (on-line). [cit. 2009-09-15]. <http://www.trackinghackers.com/papers/honeypots.html>
5. SINČÁK, P., ANDREJKOVÁ, G. *Neurónové siete Inžiniersky prístup (1. diel)*. (Neural Networks – Engineering Approach, Part 1). Košice: Technická Univerzita Košice, 2005.
6. HIRSTEV, R., M. *The ANN Book. Edition 1*. (on-line). [cit. 2009-08-14] <ftp://ftp.funet.fi/pub/sci/neural/books>
7. KVASNICKA, V. a kol. *Úvod do teórie neurónových sietí*. (Introduction to Neural Networks Theory). Bratislava: IRIS, Bratislava, 1997. 140 p. ISBN 80-8878-30-1
8. BROWN, M. *Layered Perceptron Networks and the Error Back Propagation Algorithm*. (on-line). [cit. 2009-07-7] [http://www.dynamics.unam.edu/seminarios/sistemasdinamicos/miguel/Neural%20Networks%20books/perceptron/Brown\\_tutorial\\_perceptr.pdf](http://www.dynamics.unam.edu/seminarios/sistemasdinamicos/miguel/Neural%20Networks%20books/perceptron/Brown_tutorial_perceptr.pdf)
9. SZIBIRK, B. N. Towards a Model of Multi-Agent Connectionist Hybrid System. In: *Artificial Neural Networks and Expert Systems, New Zealand Conference, 2nd New Zealand Two-Stream International Conference on Artificial Neural Networks and Expert Systems*. IEEE Computer Society: New Zealand, 1995, pp. 273-284.
10. MINSKY, M. Logical vs. Analogical or Symbolic vs. Connection& or Neat vs. Scruffy. In: *Artificial Intelligence at MIT., Expanding Frontiers”vol.1*. Cambridge, MA: MIT Press, 1990. pp. 177-192.
11. GOODALL, J. *Information Visualization for Intrusion Detection. The Intrusion Detection Tool Kit* (on-line). [cit. 2009-10-15] <http://userpages.umbc.edu/~jgood/idtk.php>



12. DOKAS, P., ERTOZ, L., KUMAR, V., LAZAREVIC, A., SRIVASTAVA, J., PANG-NING T. *Data Mining for Network Intrusion Detection* (on-line). [cit. 2009-09-05] [http://www.cs.umn.edu/research/MINDS/papers/nsf\\_ngdm\\_2002.pdf](http://www.cs.umn.edu/research/MINDS/papers/nsf_ngdm_2002.pdf)
13. BLOEDORN, E., CHRISTIANSEN, A., HILL, W., SKORUPKA, C., TALBOT, L., TIVEL, J. *Data Mining for Network Intrusion Detection: How to Get Started*. (on-line). [cit. 2009-09-07] [http://www.mitre.org/work/tech\\_papers/tech\\_papers\\_01/bloedorn\\_datamining/bloedorn\\_datamining.pdf](http://www.mitre.org/work/tech_papers/tech_papers_01/bloedorn_datamining/bloedorn_datamining.pdf)
14. JOYCE, J. B. *Methods and apparatus for heuristic firewall*. (on-line). [cit. 2009-09-07] <http://www.wipo.int/pctdb/en>
15. ALFY, E. A Heuristic Approach for Firewall Policy Optimization. In: *IEEE International Conference on Advanced Communication Technology (ICACT'07)*. Jordan: IEEE Computer Society, 2007, pp. 819-824.
16. DENNING, E. An Intrusion Detection Model. In: *Proceedings of the Seventh IEEE Symposium on Security and Privacy May 1986*. Oakland Cal.: ACM Press, Ltd., 1996, pp. 119-131.
17. MINSKY, M., PAPERT, S. *Perceptrons. An Introduction to Computational Geometry*. Cambridge: The MIT Press, MA, 1969. 268 p. ISBN 0-26-2130-43-2
18. HANSEN, K. Redundancy Ethernet in industrial automation. In: *Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference on Volume: 2*. Catania Italy: CRC Press, 2005, pp. 947 – 952. ISBN 0-7803-9401-1
19. ZIMMERMANN, H. OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection. In: *IEEE Transactions on Communications*, **28**(4). Norwood, MA, USA: Artech House, Inc., 1980, pp. 425 – 432.
20. FENG, W., GOEL, A., BEZZAZ, A., WALPOLE, J. TC Pivo A High-Performance Packet Replay Engine. In: *ACM SIGCOMM 2003 Workshop on Models, Methods, and Tools for Reproducible Network Research (MoMeTools)*. NY USA: ACM Press, Ltd., 2003, pp. 57 – 64. ISBN 1-58113-748-8
21. Industrial communication networks - Profiles - Part 1. Fieldbus profiles Publication IEC61784-1 Ed. 2.0 (on-line) <http://webstore.iec.ch/webstore/webstore.nsf/artnum/038786>
22. RUMELHART, E. D., HINTON, G.E., WILIAMS, R. J. Learning internal representation by error propagation. In: *Parallel Distributed*

- Processing. Explorations in the Microstructure of Cognition. Vol 1: Foundation.* Cambridge: The MIT Press, 1987, pp. 318 – 362.
23. BIGELOV, S. J. *Mistrovství v počítačových sítích.* (Mastery of Computer Networks). Praha: Computer Press, 2004. 992 p. ISBN 80-251-0178-9
  24. KABELOVÁ, A., DOSTÁLEK, L. *Velký průvodce protokoly TCP/IP a systémem DNS.* (Large Guide for TCP/IP Protocols and DNS Systems). Praha: Computer Press, 2008. 488p. ISBN 9-7880-25122-36-5
  25. SNAPP, S. R., DIAS, J., GIHAN, V., TERRANCE, L., HEBERLEIN, L., CHE-LIN, H. *DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and An Early Prototype.* In: *The 14th National Computer Security Conference.* California. Washington DC: IEEE Computer Society, 1991, pp. 167-176.
  26. Interactive Catalog Siemens: Industrial Communication (on-line). [cit. 2011-12-01] <https://mall.automation.siemens.com/CZ/guest/index.asp>
  27. IETF RFC 4034: Resource Records for the DNS Security Extensions. (online, cit. 1 Dec 2009) <http://www.rfc-archive.org/getrfc.php?rfc=4034>:
  28. Industrial communication networks - Fieldbus specifications - Part 5-11: Application layer service definition. (on-line, 14 Dec 2007) <http://webstore.iec.ch/webstore/webstore.nsf/artnum/038786>
  29. SEBRING, M., WHITEHURST, R. A. Expert Systems in Intrusion Detection: A Case Study. In: *The 11th National Computer Security Conference Baltimore.* Berlin: Springer – Verlag, 1988, pp. 74-81.
  30. DECOTIGNIE, J. D. A perspective on Ethernet-TCP/IP as a fieldbus. In: *IFAC International Conference on Fieldbus Systems and Their Applications.* France: Pergamon, 2001, pp. 138-143.
  31. SCHIFFER, V., VANGOMPEL, D., ROMITO, R. *The Specification for DeviceNet™ Volume One: Common Industrial Protocol (CIP) Specification.* (on-line). [cit. 2008-10-15] <http://www.odva.org/default.aspx?tabid=79>
  32. Modbus protocol Specification: Overview of MODBUS over TCP/IP (on-line). [cit. 2008-12-01] <http://www.modbus.org/docs/>
  33. Reference IEC 61588: Precision clock synchronization protocol for networked measurement and control systems. (on-line). [cit. 2009-10-21] <http://webstore.iec.ch/webstore/webstore.nsf/artnum/042668>
  34. RÖSSEL, T. *Distributing security for Industrial Ethernet networks.* (on-line). [cit. 2007-10-05] <http://www.industrial-embedded.com>

35. DEBAR, H., BECKE, M., SIBONI, D. A Neural Network Component for an Intrusion Detection System. In: *Proceedings of the IEEE Computer Society Symposium on Research in Security and Privacy. Oakland, California.* Washington DC, USA: IEEE Computer Society, 1992, pp. 166 – 170.
36. LUNT, T. F. Detecting Intruders in Computer Systems. In: *Journal of Network and Computer Applications: Conference on Auditing and Computer Technology, SRI International.* London: Academic Press Ltd., 1993, pp. 114 – 132.
37. BRADEN, R. *Requirements for Internet Hosts, Application and Support (RFC 1123).* (on-line). [cit. 2007-10-05] <http://tools.ietf.org/html/rfc1123>
38. BRADEN, R. *Requirements for Internet Hosts, Application and Support (RFC 1122).* (on-line). [cit. 2007-10-05] <http://tools.ietf.org/html/rfc1122>
39. HALENÁR, R. Methods of data warehouses building for university purpose. In: *Infokomunikacionnye tehnologii v nauke, proizvodstve i obrazovanii: četvertaja meždunarodnaja naučno-techničeskaja konferencija* (Info-communication Technologies in Science, Production and Education: 4th International Scientific Conference). 2012 , pp. 154-157. ISSN 2219-293X
40. SATRAPA, P. *Internetový protokol IPv6* (Internet Protocol Ipv6). Praha: CZ.NIC, 2008. 359 p. ISBN 978-80-904248-0-7
41. MARSALL, M. McCulloch-Pitts Neurons. In: *The 2008 Annual Meeting of the consortium on cognitive science instruction (ccsi).* Washington DC: IEEE Computer Society, 2008, pp. 172 – 179.
42. LEE, W., STOLFO, S.J. A framework for constructing feature and models for intrusion detection systems. In: *ACM Transactions on Information and System Security (TISSEC), Volume (4).* NY USA: ACM Press, Ltd., 2002. pp. 227–261.
43. FRANK, J. Artificial Intelligence and Intrusion Detection: Current and Future Directions. In: *Proceedings of the 17th National Computer Security Conference.* Berlin: Springer – Verlag, 1994, pp. 562 – 565. ISSN 1063-9527
44. HELMAN, P., LIEPINS, G., RICHARDS, W. Foundations of Intrusion Detection. In: *Proceedings of the Fifth Computer Security Foundations Workshop.* New Hampshire, USA: IEEE Computer Society, 1992, pp. 114-120.

45. DEBAR, H., DORIZZI, B. An Application of a Recurrent Network to an Intrusion Detection System. In: *Proceedings of the International Joint Conference on Neural Networks. (II)*. London: ACM Press, Ltd., 1992, pp. 478-483.
46. CHANDOLA, V., BANERJEE, A., KUMAR, V. Anomaly Detection: A Survey. In: *ACM Computing Surveys, Vol. 41(3), Article 15*. NY USA: ACM Press, Ltd., 2009, pp. 167 – 182. ISSN: 0360-0300
47. AXELSSON, S. *Intrusion Detection Systems: A Taxonomy and Survey*. Dept. of Computer Engineering, Chalmers University of Technology, Sweden. (on-line). [cit. 2009-10-11] <http://www.ce.chalmers.se/staff/sax/taxonomy.ps>
48. FOX, L., HENNING, R., REED, J. H. A Neural Network Approach Towards Intrusion Detection. In: *Proceedings of the 13th National Computer Security Conference Baltimore*. Berlin: Springer – Verlag, 1990, pp. 125 – 134. ISSN 1063-9527
49. THOTTAN, M., JI, CH. Anomaly Detection in IP Networks. In: *Journal of Network and Systems Management : IEEE Transactions on signal processing, vol. 51, no. 8*. New York: Springer, 2007, pp. 267 – 283. ISSN 1573-7705
50. CANNADY, J. Neural Networks for Misuse Detection. In: *Proceedings of the 1998 National Information Systems Security Conference (NISSC'98)*. Berkeley: USENIX Association, 1999, pp. 443–456.
51. LASKOV, P., DUSSEL, P., SCHAFFER, CH., RIECK, K. Learning intrusion detection: Supervised or unsupervised? In: *Image Analysis and Processing - ICIAP, 13th International Conference, Cagliari*. Germany: Springer – verlag, 2005, pp. 50-57.
52. ZHONG, S., KHOSHGOFTAAR, T., SELIYA, N. Clustering-based Network Intrusion Detection. In: *International Journal of Reliability, Quality and Safety Engineering, Vol. 14*. NY USA: ACM Press, Ltd., 2007, pp. 169-187.
53. LICHODZIJEWski, P., ZINCIR-HEYWOOD, A., HEYWOOD, M. Dynamic intrusion detection using self organizing maps. In: *Proceedings of the 2002 IEEE World Congress on Computational Intelligence*. New Jersey: IEEE Computer Society, 2002, pp. 412 – 419.
54. HAMMERST, D. Working with neural networks. In: *IEEE Spectrum Magazine*, 1993, **30**, pp. 46-53. ISSN 0018-9235

55. ĎUĐÁK, J., GAŠPAR, G., KEBÍSEK, M. Application of modified MODBUS protocol for the needs of modern measurement systems. In: *Advances in Mechatronics 2011 Proceedings of the 6th International Conference on Advances in Mechatronics 2011 (AIM'11)*. Brno: University of Defence, 2011, pp. 9-14. ISBN 978-80-7231-848-3
56. CHEN, J.R., MARS, P. Stepsize Variation Methods for Accelerating the Back-propagation Algorithm. In: *International Joint Conference on Neural Networks volume 1*. NY USA: ACM Press, Ltd., 1998. pp. 601-604.
57. JOOST, M., SCHIFFMANN, W. Speeding Up Backpropagation Algorithms by Using Cross-Entropy Combined with Pattern Normalization In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems (IJUFKS) Volume: 6*. New York: Springer, 1998, pp.117-126. ISSN 1572-9338
58. HAYKIN, S. *Neural Networks and Learning Machines, third edition*. Canada: Prentice Hall, 2009. 936 p. ISBN 01-31471-39-2
59. TVETER, D. *The Pattern Recognition Basis of Artificial Intelligence*. Washington: Wiley-IEEE Computer Society Press, 1998. 388 p. ISBN 978-0-8186-7796-0
60. ROBBINS, A. *GAWK Effective AWK Programming: A User's Guide for GNU Awk, for the 3.1.7 (or later) version of the GNU implementation of AWK The GNU*. Cambridge: O'Reilly Media, 2001. 456 p. ISBN 978-0-596-55594-8
61. AHO, V.A., KERNIGHAN, W.B., WEINBERGER, P. J. *The AWK Programming Language*. Boston: Addison-Wesley, 1988. 210 p. ISBN 0-201-07981-X
62. ROESCH, M., GREEM, CH. *SNORT User's manual 2.8.5, The Snort Project, 2009*. (on-line). [cit. 2009-11-21] <http://www.snort.org/docs>
63. OREBAUGH, A., BILES, S., BABBIN, J. *Snort cookbook*. Sebastopol: O'Reilly Media Inc., 2005. 288 p. ISBN 0-596-00791-4
64. BEALE, J., BAKER,R.B., ESLER, J., KOHLENBERG, T., NORTHUTT, S. *Snort: IDS and IPS toolkit*. Burlington: Syngress Publishing, 2007. 768 p. ISBN 978-1-59749-099-3
65. MASTERS, T. *Practical neural network recipes in C++*. California: Timothy Masters Academic Press, San Diego, 1998. 493 p. ISBN 978-0-12479-040-7
66. TANG, H., TAN, K.C. , ZHANG, Y. *Neural networks: computational models and applications*. Berlin: Springer-Verlag, Heidelberg, 2007. 299 p. ISBN-10 3-540-69225-8

67. OBAIDAT, M.S., BOUDRIGA, N. *Security of e-systems and computer networks*. New York: Cambridge University Press, 2007. 386 p. ISBN: 978-0-521-83764-4
68. LIPTÁK, B.G. *Instrument Engineers' Handbook: Process Software and Digital Networks v.3: Process Software and Digital Networks Vol 3*. Danvers: CRC Press, 2002. 912 p. ISBN 0-8493-1082-2
69. GREGG, M., WATKINS, S., MAYSG., RIES, CH. *Hack the Stack: Using Snort and Ethereal to Master the 8 Layers of an Insecure Network*. Rockland: Syngress Publishing, USA, 2006. 442 p. ISBN 978-1-59749-109-9
70. YAGHMOUR, K., MASTERS, J., BEN-YOSSEF, G., GERUM, P. *Building Embedded Linux Systems*. Sebastopol: O'Reilly Media, Sebastopol, 2003. 416 p. ISBN 978-0-596-10327-9
71. PAPA, M., SHENOI, S. *Critical Infrastructure Protection II*. Berlin: Springer-Verlag, 2008. 270 p. ISBN: 978-0-387-88522-3
72. BATTEN, L.M., SAFAVI-NAINI, R. Information security and privacy. In: *11th Australasian conference ACISP 2006 proceedings*. Berlin: Springer-Verlag, 2006. pp. 387 – 401.
73. ŠIMON, M., HALENÁR, R., TRNKA, A. Possibilities of using firewalls in process control networks In: *Management of Production Systems with Support of Information Technologies and Control Proceedings*. Nitra: Department of Machines and Production Systems Slovak University of Agriculture in Nitra, 2006, pp. 294-299. ISBN 80-8069-743-4
74. SCOTT, E. F. An empirical study of learning speed in back-propagation networks. In: *Technical Report CMU-CS-88-162, School of Computer Science*. Pittsburgh: Carnegie Mellon University, 1988. p.7.
75. YOUNG, S.E., AITEL, D. *The hacker's handbook: the strategy behind breaking into and defending Networks*. Florida: Auerbach Publications, Florida, 2004. 896 p. ISBN 0-8493-0888-7
76. PRECHELT, L. *Proben1: A Set of Neural Network Benchmark Problems and Benchmarking*. In: *Technical Report 21/94*. Fakultat für Informatik Universität Karlsruhe, 1994. p. 9.
77. YAO, Y., YU, G., FU-XIANG, G. A neural network approach for misuse and anomaly intrusion detection. In: *Journal of natural of natural sciences*, 2005, **10**(1), pp. 115 – 118. ISSN1007-1202

78. DOSEDĚL, T. *Počítačová bezpečnost a ochrana dat.* (Computer Safety and Data Protection). Brno: Computer Press, 2004. 184 p. ISBN 80-251-0106-1
79. ANDRÉS, S., KENYON, B. *Security Sage's guide to hardening the network infrastructure.* Rockland: Syngress Publishing, Rockland USA, 2004. 608 p. ISBN 1-931836-01-9
80. GOFTON, W.P. *Sériová komunikace* (Series Communication), Praha: Grada Publishing, 1995. 240 p. ISBN 80-7169-131-3
81. FLICKENGER, R. *Linux server na maximum.* (Linux Server at Maximum), Brno: CP Books, Brno, 2005. 232 p. ISBN 80-251-0586-5
82. PUŽMANOVÁ, R. *TCP/IP v kostce.* (IP in a Nutshell) České Budějovice: KOPP nakladatelství (KOPP Publishing), České Budějovice, 2004. 608 p. ISBN 978-80-7232-388-3
83. BLAKELEY, B., BOYD, D.J., SMITH, S. *Introduction to Networking Technologies.* Washington: IBM Redbooks, USA, 1994. 248 p. ISBN 0-738406-30-9
84. LIU, W., MATTHEWS, C., PARZIALE, L., ROSSELOT, N., DAVIS, CH., FORRESTER, J., BRITT, T.D. *TCP/IP Tutorial and Technical Overview,* Washington: IBM Redbooks, 2006. 1004 p. ISBN 0-73-8494-68-2
85. WEI, D., JAFARI, M., LU, Y. On Protecting Industrial Automation and Control Systems against Electronic Attacks. In: *Proceedings of the 3rd Annual IEEE Conference on Automation Science and Engineering* Scottsdale. USA, 2007, pp. 176 – 181.
86. <http://www.snort.org>
87. <http://www.bro-ids.org/>
88. <http://www.prelude-ids.com>
89. <http://www.ossec.net/>
90. <http://www.akmalabs.com>
91. <http://www.snort.org/snort/license/gpl>
92. <http://www.membrain-nn.de>
93. <http://www.ra.cs.uni-tuebingen.de/SNNS/>
94. <http://www.genesis-sim.org/GENESIS/>
95. <http://www.dontveter.com/basisofai/basisofai.html>
96. <http://gcc.gnu.org/>
97. <http://tcl.sourceforge.net/>
98. <http://lobster.ics.forth.gr/traces/>
99. <http://www.nsauditor.com/>
100. <http://www.gnu.org/software/gawk/gawk.html>

# CONTENTS

<b>INTRODUCTION.....</b>	<b>8</b>
<b>1. DATA TRANSFER IN NUMERIC SYSTEMS CONTROL .....</b>	<b>10</b>
1.1 Communication standards of control and production systems.....	10
1.1.1 Automation networks protocols encapsulation.....	11
1.2 Reference structure model of communication protocols.....	15
1.3 Protocols of RM OSI 3rd and 4th layers.....	18
<b>2. SECURITY OF CONTROL SYSTEMS IN DATA TRANSFER..</b>	<b>20</b>
2.1 Categories of communication systems attacks.....	21
2.1.1 Special techniques of data transfer.....	23
2.2 Ways of industrial systems security.....	25
2.2.1 Firewall.....	26
2.2.2 IDS.....	27
<b>3. ANALYSIS OF CURRENT NEURAL NETWORKS TECHNOLOGIES.....</b>	<b>28</b>
3.1 Neural structure.....	29
3.1.1 Topology of neural networks.....	33
<b>4. SUBJECT MATTER FORMULATION .....</b>	<b>35</b>
<b>5. POSSIBILITIES OF NN USE BY SYSTEMS COMMUNICATION VALIDATION.....</b>	<b>38</b>
5.1 System proposal.....	40
5.2 Proposal of host system.....	41
5.2.1 Determination of parameters for transfer anomalies detection.....	47
5.3 Selection of neural network algorithm.....	48
5.4 Implementation proposal and learning of network.....	51
5.4.1 Application equipment for neural networks implementation.....	52
5.4.2 Preparation of input data.....	55
5.4.3 Neural network implementation.....	64
5.5 Experimental system verification.....	70
5.5.1 Testing of proposed neural network.....	70
5.5.2 Possibilities of practical implementation.....	73
<b>6. CONCLUSION .....</b>	<b>75</b>
6.1 Possibilities of further development.....	78
<b>REFERENCES.....</b>	<b>80</b>



