International Journal of Soft Computing 3 (4): 264-270, 2008
ISSN: 1816-9503

Datamining and Intrusion Detection Using Back-Propagation Algorithm for Intrusion Detection

[1]E. Anbalagan, [2]C. Puttamadappa, [3]E. Mohan, [4]B. Jayaraman
and [5]Srinivasarao Madane
[1]Vinayaka Missions University, Salem, Tamil Nadu, India
[2]New Horizon College of Engineering, Bangalore-560087, India
[3]Department of Electronics and Communication Engineering,
[4]Department of Computer Science and Engineering, Pallavan College of Engineering,
Thimmasamudram, Kanchipuram-631 502, Tamilnadu, India
[5]Adhiparasakthi College of Engineering, Kalavai, Tamilnadu, India

Abstract: Transmission of data over the internet keeps on increasing. The need to protect connected systems also increasing. Intrusion Detection Systems (IDSs) are the latest technology used for this purpose. Datamining plays an important role in matching intrusions with the data stored in the system. Although the field of IDSs is still developing, the systems that do exist are still not complete, in the sense that they are not able to detect all types of intrusions. Some attacks which are detected by various tools available today cannot be detected by other products, depending on the types and methods that they are built on. In this research, an artificial neural network using back-propagation algorithm has been used to implement the IDS. Inspite of much related work had been done, this study elucidates the implementation aspects of BPA for a real_ time IDS.Thousand packet information both normal and intrusion have been considered for implementation. The result of ID is very close to 99%. The topology of the ANN is (41×10×1). The network converged with 550 iterations. Very huge amount of packets are to be evaluated to know the complete performance of the developed system.

Key words: Datamining, intrusion detection, neural network, signature, back-propagation algorithm

INTRODUCTION

The complexity, as well as the importance, of distributed computer systems and information resources is rapidly growing. Due to this, computers and computer networks are often exposed to computer crime. Many modern systems lack properly implemented security services; they contain a variety of vulnerabilities and, therefore, can be compromised easily. As network attacks have increased in number over the past few years, the efficiency of security systems such as firewalls have declined.

It is very important that the security mechanisms of a system are designed to prevent unauthorized access to system resources and data. Building a complete secure system is impossible and the least that can be done is to detect the intrusion attempts so that action can be taken to repair the damage later. Organizations are increasingly implementing various systems that monitor IT security breaches. Intrusion Detection Systems (IDSs) have gained a considerable amount of interest within this area (Kanzienko and Dorosz, 2003). The main task of an IDS is to detect an intrusion and, if necessary or possible, to undertake some measures eliminating the intrusions.

Because most computer systems are vulnerable to attack, Intrusion Detection (ID) is a rapidly developing field. Intrusion Detection Systems (IDSs) detect intrusions using specific methodologies that are specific to each of them. A method describes how an IDS analyzes data to detect possible intrusions, based on the analysis approaches. The analysis approaches are anomaly detection and misuse detection. There are many methods that are used. Examples of them include statistical approaches (Sundaram, 2001), protocol anomaly detection (Verwoerd and Hunt, 2001), neural networks (Philippe, 2004), file checking, expert systems, rule-based measures (Gordeev, 2004) and Genetic Algorithms (GAs). Inspite of the different intelligent techniques datamining plays

Corresponding Author: E. Anbalagan, Vinayaka Missions University, Salem, Tamil Nadu, India

prominent role in quickly comparing the extracted features form the packets with the already stored database intrusion information.

Intruders tend to find new ways to compromise systems each day. As more intrusions occur, the weaknesses of existing technologies like firewalls are exposed. Since it is impossible to build a complete secure system, IDSs are used to detect the intrusions that occur. This is why IDSs are gaining acceptance in every organization. To understand what an IDS is, first one should know what intrusion and intruders are.

Sundaram defines intrusion as the unauthorized attempt to access information, manipulate information, or render a system unreliable or unusable. To detect intrusions and to prevent them, one has to be aware of how an intruder can cause intrusions. The primary ways an intruder can get into the system is through primary intrusion, system intrusion and remote intrusion (Graham, 2000). ID is the process of monitoring the events occurring in a computer system or network and analyzing them for intrusions (Bace and Mell, 2004). The prevention of intrusions should be done through effective IDSs. An IDS is a software or hardware product that automates this monitoring and analysis process.

The types of IDSs can be described in terms of 3 fundamental functional components. They are the information source, analysis and response. The information source of the system mainly depends on where the IDSs are being placed, hence it is also known as the monitoring locations of the IDS. The information sources are mainly of three types: network-based IDSs, host-based IDSs and application-based IDSs (Bace, 2002). Since, the focus of this research is on network-based IDS, the other two types will not be considered here. Network-based IDSs detect attacks by capturing and analyzing network packets. They search for attack signatures within the packets. Signatures might be based on actual packet contents and are checked by comparing bits to known patterns of attack. If the bits are matched to known patterns of attack, then an intrusion is triggered. Once the information sources have monitored network traffic, the next step is to analyze the events to detect the intrusion. The two main techniques or approaches used to analyze events to detect attacks are misuse detection and anomaly detection. Response is the set of actions that the system takes once it detects intrusions. Some of the responses involve reporting results and findings to a pre-specified location, while others are more actively automated responses.

Commercial IDSs support both active and passive responses and sometimes a combination of the two. IDSs can be viewed as the second layer of protection against unauthorized access to networked information systems because despite the best access control systems, intruders are still able to enter computer networks. IDSs expand the security provided by the access control systems by providing system administrators with a warning of the intrusion.

They also provide the system administrators with necessary information about the intrusions. This assists the system administrators in controlling the intrusions that has occurred, in order to avoid them in the future or to minimize the damage that may occur due to an intrusion. Although, IDSs can be designed to verify the proper operation of access control systems by looking for the attacks that get past the access control systems, IDSs are more useful when they can detect intrusions that use methods that are different from those used by the access control systems. For this purpose, they must use more general and more powerful methods than simple database look-ups of known attack scenarios.

SIGNATURE BASICS

The different sets of signatures are stored in a database for template matching process during the actual intrusion detection implementation stage. The well known intrusion patterns are properly stored in a particular format in server incase of a network or in the host incase of intrusion detection system used only in host.

A network IDS signature is a pattern that we want to look for in traffic. Some of the methods that can be used to identify each one:

- Connection attempt from a reserved IP address. This is easily identified by checking the source address field in an IP header.
- Packet with an illegal TCP flag combination. This can be found by comparing the flags set in a TCP header against known good or bad flag combinations.
- Email containing a particular virus. The IDS can compare the subject of each email to the subject associated with the virus-laden email, or it can look for an attachment with a particular name.
- DNS buffer overflow attempt contained in the payload of a query. By parsing the DNS fields and checking the length of each of them, the IDS can identify an attempt to perform a buffer overflow using a DNS field. A different method would be to look for exploit shellcode sequences in the payload.
- Denial of service attack on a POP3 server caused by issuing the same command thousands of times. One signature for this attack would be to keep track of how many times the command is issued and to alert when that number exceeds a certain threshold.

- File access attack on an FTP server by issuing file and directory commands to it without first logging in. A state-tracking signature could be developed which would monitor FTP traffic for a successful login and would alert if certain commands were issued before the user had authenticated properly.

Purpose of signatures: Different signatures have different goals. The obvious answer is that, want to be alerted when an intrusion attempt occurs. But let's take a moment to think about other reasons why we might want to write or modify a signature. Perhaps seeing some odd traffic on network and want to be alerted the next time it occurs. It has been noticed that it has unusual header characteristics and want to write a signature that will match this known pattern. Perhaps are interested in configuring IDS to identify abnormal or suspicious traffic in general, not just attacks or probes. Some signatures may tell which specific attack is occurring or what vulnerability the attacker is trying to exploit, while other signatures may just indicate that unusual behavior is occurring, without specifying a particular attack. It will often take significantly more time and resources to identify the tool that's causing malicious activity, but it will give more information as to why being attacked and what the intent of the attack is.

Header values: Simple signature characteristic header values is presented. Some header values are clearly abnormal, so they make great candidates for signatures. A classic example of this is a TCP packet with the SYN and FIN flags set. This is a Violation of Request For Comments (RFC 793) (which defines the TCP standard) and has been used in many tools in an attempt to circumvent firewalls, routers and intrusion detection systems. Many exploits include header values that purposely violate RFCs, because many operating systems and applications have been written on the assumption that the RFCs would not be violated and don't perform proper error handling of such traffic. Many tools either contain coding mistakes or are incomplete, so that crafted packets produced by them contain header values that violate RFCs. Both poorly written tools and various intrusion techniques provide distinguishing characteristics that can be used for signature purposes.

There's a catch. Not all Operating system (Os) and applications completely adhere to the RFCs. In fact, many have at least one facet of their behavior that violates an RFC. Over time, protocols may implement new features that are not included in an RFC. New standards emerge over time, which may "legalize" values that were previously illegal; RFC 3168, for Explicit Congestion Notification (ECN), is a good example of this. So an IDS signature based strictly on an RFC may produce many false positives. Still, the RFCs make a great basis for signature development, because so much malicious activity violates RFCs. Because of RFC updates and other factors, it's important to review and update existing signatures periodically.

Sample signature:

- Various source IP addresses.
- TCP source port 21, destination port 21.
- Type of service 0.
- IP identification number 39426.
- SYN and FIN flags set.
- Various sequence numbers set.
- Various acknowledgment numbers set.
- TCP window size 1028.

Packet values that are completely normal don't make good signature characteristics by themselves, although they are often included to limit the amount of traffic that we study. For example, include the normal IP protocol value of 6 for a protocol, so that only check TCP packets. But other characteristics that are completely normal, such as the type of service set to 0, are much less likely to be helpful in signature development.

A signature based on few suspicious characteristics may be too specific. Although, it would provide much more precise information about the source of the activity, it would also be far less efficient than a signature that only checks one header value. Signature development is always a tradeoff between efficiency and accuracy. In many cases, simpler signatures are more prone to false positives than more complex signatures, because simpler signatures are much more general. But more complex signatures may be more prone to false negatives than simpler signatures, because one of the characteristics of a tool or methodology may change over time.

ARTIFICIAL NEURAL NETWORK (ANN)

An Artificial Neural Network (ANN) is a mathematical way of simulating the capability of human brain. The category of supervised method requires inputs and target outputs. The Back-Propagation Algorithm (BPA) is a supervised method (Hirose *et al*., 1991; Bershard *et al*., 1993a; Rumelhart *et al*., 1986; Lippmann 1987) that uses steepest-descent method to reach global minima. The multi layer perceptron is shown in Fig. 1. The flowchart for the BPA is given in Fig. 2. The number of layers and

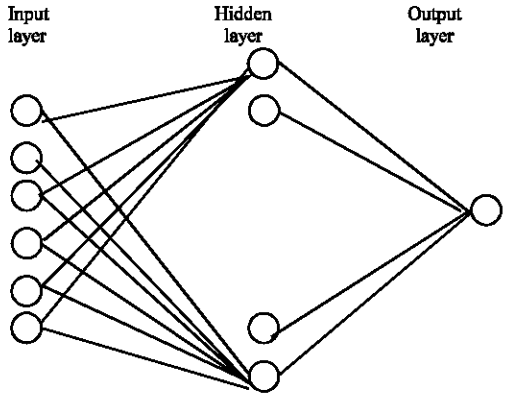

Fig. 1: Back propagation network

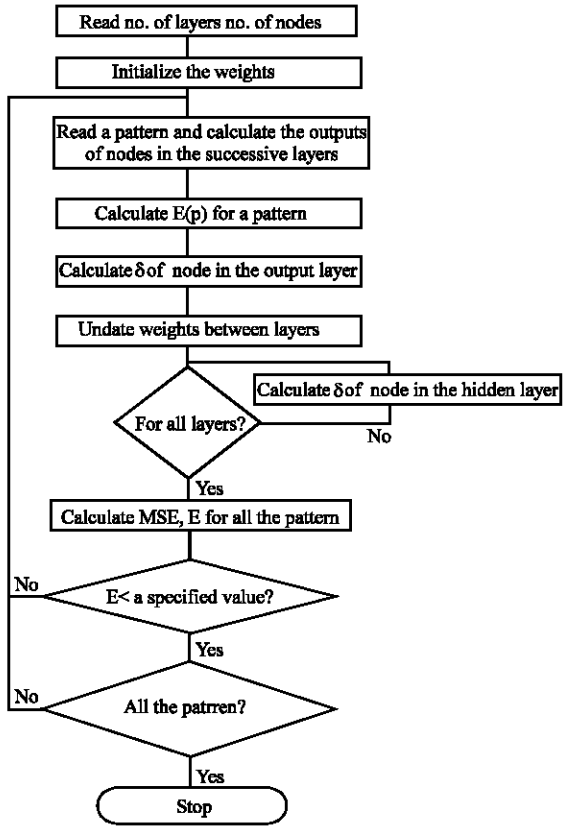

Fig. 2: Flow chart for BPA

number of nodes in each layer is decided. The connections between nodes are initialized with random weights. A pattern from the training set is presented in the input layer of the network and the error is calculated in the output layer. The error is propagated backwards towards the input layer and the weights are updated. This procedure is repeated for all the training patterns. At the end of iteration, test patterns are presented to ANN and the classification performance of ANN is evaluated. Further training of ANN is continued till the desired classification performance is reached

STEPS INVOLVED

Forward propagation: The output of each node in the successive layers is calculated:

$$o(\text{output of a node}) = \frac{1}{1+e^{-\sum w_{ij}x_i}} \quad (1)$$

Where,

w_{ij}: The weight matrix connecting nodes of the previous layer i with nodes of next layer j.
x_i: The variables of a pattern.
o: The output of a node in the successive layer.

The error E (p) of a pattern number p is calculated

$$E(p) = \frac{1}{2}\sum (d(p) - o(p))^2 \quad (2)$$

Where,
d: target value of a pattern

Reverse propagation: The error δ for the nodes in the output layer is calculated

$$\delta(\text{output layer}) = o(1-o)(d-o) \quad (3)$$

The new weights between output layer and hidden layer are updated

$$w(n+1) = w(n) + (\text{output layer})\ o(\text{hidden layer}) \quad (4)$$

Where,
η: is the learning factor (0<η<1)

The error δ for the nodes in the hidden layer is calculated

$$(\text{hidden layer}) = o(1-o)\ (\text{output layer})\ w(\text{updated weights between hidden and output layer}) \quad (5)$$

The weights between hidden and input layer are updated.

$$w(n+1) = w(n) + (\text{hidden layer})\ o(\text{input layer}) \quad (6)$$

The above steps complete one weight updation. The remaining training patterns are presented and Eq. 1- 6 are followed which form one iteration. The training of the network is stopped once the desired mean squared error (MSE) is reached as given below:

$$E(MSE) = \sum E(p) \quad (7)$$

The final updated weights are saved for detecting the intrusion.

NETWORK INTRUSION DETECTION SYSTEM (NIDS) USING ANN

Packet capture: Packet capture is the beginning process in NIDS. It can be implemented by setting the working mode of the network card as the promiscuous mode. The network card under common mode can only receive the packet whose destination address is the network card itself. However, only those packets are not sufficient to serve for the data source of the NIDS. So, it is necessary to set the network card's working mode as the promiscuous mode. Under this mode, the network card can receive not only the packets sent to itself but also the packets routed to some other hosts. Thus, the NIDS can monitor the network stream of all hosts of some local area network and detect whether intrusion happens or not.

Feature extraction: Feature selection and extraction is one of the pivotal problems in implementing the intrusion detection system. Network stream itself is not suitable directly as the input of the back-propagation (BP) classifier, so it is necessary to extract some features from the network stream. The features extracted from the network stream form a feature vector which serves for the description of the packet. Whether the feature vector can describe the network stream correctly and efficiently or not has a large effect on the efficiency and correctness of the NIDS. Selecting several features such as the protocol code, the packet head length, the checksum, the port number and some TCP Flags, etc have been done. Based on these features, a vector is obtained as follow to describe an intrusion.

- Attack (type)=(P-id, H-len, C-sum, S-port, D-port, lcmp-type, lcmp-Code, Flag, P-len, P-data)

Above is the general description form of an abstract attack.

Perhaps some concrete examples can explain the vector well.

- Attack (Cgl)=(Tcp, 32, O, 2345, 80, null, null, A, 421, get cgi-bin)
- Attack(FTP)--(Tcp, 24, 16, 21, 21, null, null, PA, 256, ROOm
- Attack(Redirect)=(Icmp, 20, null, null, 8, 3, null, 192, la)
- Attack(UDP)=(Udp, 16, 10, 138, 126, null, null, nuli, 448, 3c)

Combined with the general description form of an intrusion. When the features of a packet are set as above, it represents a CGI attack, a FTP attack, a REDIRECT attack and a UDP attack respectively. The feature vector will serve for the input of the BP Classifier., then the BP Classifier will judge whether the feature vector represents an intrusion or not.

EXPERIMENTAL SETUP

The simulation results were obtained from the standard KDD data set. It is a well defined as normal and with different types of attack for TCP, UDP, ICMP, etc. A set of sample data set is shown in Table 1. Each row is a pattern. The fields in each pattern describe the properties of respective packet. Instead of KDD data set, free sniffer softwares like network sniffer, packet sniffer and more

Table 1: Sample KDD dataset

S. No	Packet details
1	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00, 0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal
2	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00, 0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal.
3	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00, 1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal.
4	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,0.00,1.00, 0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack.
5	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00, 0.00,0.00,255,254,1.00,0.01,0.01,0.00,0.00,0.00,0.00,0.00,snmpgetattack.
6	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00, 0.00,255,255,1.00,0.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack.
7	0,udp,domain_u,SF,29,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,1,0.00,0.00,0.00,0.00,0.50,1.00,0.00,10,3,0.30,0.30,0.30,0.00,0.00,0.00,0.00,0.00,0.00,normal.
8	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,1,1,0.00,0.00,0.00,0.00,1.00,0.00,0.00, 255,253,0.99,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,normal.
9	0,udp,private,SF,105,146,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2,2,0.00,0.00,0.00,0.00,1.00,0.00,0.00,255,254,1.00,0.01,0.00,0.00,0.00,0.00,0.00,0.00,0.00,snmpgetattack.
10	0,tcp,http,SF,223,185,0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0,4,4,0.00,0.00,0.00,0.00,1.00,0.00,0.00,71, 255,1.00,0.00,0.01,0.01,0.00,0.00,0.00,0.00,normal.

Table 2: Sample dataset used for training

S.no	Patterns used for training input to ANN	Target outputs
1	0 0.2 0.01 0.1 105 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0.00 0.00 0.00 0.00 1.00 0.00 0.00 255 254 1.00 0.01 0.00 0.00 0.00 0.00 0.00 0.00	0.1
2	0 0.2 0.01 0.1 105 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0.00 0.00 0.00 0.00 1.00 0.00 0.00 255 254 1.00 0.01 0.00 0.00 0.00 0.00 0.00 0.00	0.1
3	0 0.2 0.01 0.1 105 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0.00 0.00 0.00 0.00 1.00 0.00 0.00 255 254 1.00 0.01 0.00 0.00 0.00 0.00 0.00 0.00	0.1
4	0 0.2 0.01 0.1 105 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0.00 0.00 0.00 0.00 1.00 0.00 0.00 255 254 1.00 0.01 0.00 0.00 0.00 0.00 0.00 0.00	0.2
5	0 0.2 0.01 0.1 105 146 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 2 0.00 0.00 0.00 0.00 1.00 0.00 0.00 255 254 1.00 0.01 0.01 0.00 0.00 0.00 0.00 0.00	0.2

softwares can be used to extract the values of a packet, which can be further labeled as normal or an attack to be used for training. The contents of the packet should be suitably modified into a meaningful numerical values. A sample dataset used for training is shown in Table 2.

RESULTS AND DISCUSSION

The topology of ANN used is 41 × 10 × 1; no. of nodes in the input layer is 41, no. of nodes in the hidden layer is 10 and no. of nodes in the output layer is 1. The labeling is set as 0.1(Normal) or 0.2 (attack). It is mandatory to use huge amount of patterns to be presented for training ANN. However, it would take enormous amount of time for the ANN to learn the patterns. Hence, only 1000 patterns have been considered for training purpose. The dataset has been separated as training and testing (intrusion detection). Training indicates the formation of final weights which indicate a thorough learning of intrusion and normal packets along with corresponding labeling.

Figure 3 shows the convergence curve for the topology of 10 nodes in the hidden layer. The network was trained to reach MSE of 0.001. It converged in 550 iterations. If the number of patterns is further

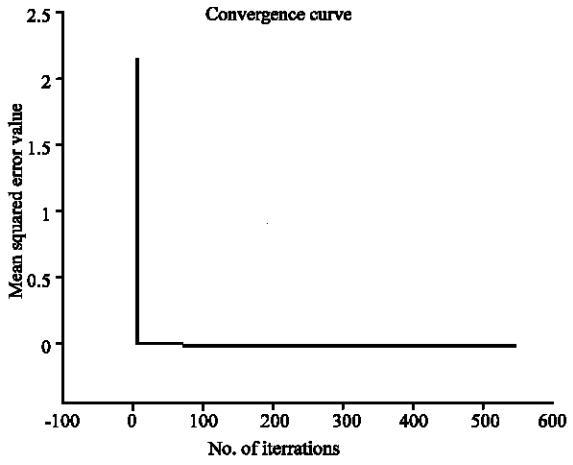

Fig. 3: Mean squared error curve

Table 3: Distribution of patterns chosen for training

Packet type	Total numbers used for training
Normal	363
Intrusion	637

Table 4: Classification performance

Packet type	Total number tested	No. classified	No. misclassified
Normal	363	360	3
Intrusion	637	600	37

Table 5: False acceptance/rejection rate

Packet type	False Acceptance Rate (FAR)	False Rejection Rate (FRR)
Normal	5.8% (37/637)	0.8% (3/360)
Intrusion	10.1%(37/363)	0.4%(3/637)

increased from 1000, then the convergence iterations will increase. In addition, the convergence depends on the orthogonolity of data presented for training. If the subsequent patterns are not orthogonal, the convergence would take a long time or it may not converge. Table 3 gives distribution of patterns chosen for training and testing. The classification performance is given in Table 4. In Table 5, False Acceptance Rate (FAR) and False Rejection Rate (FRR) are presented.

CONCLUSION

In this research, KDD dataset has been considered to experiment the performance of BPA in classifying the LAN intrusion packets. A topology of 41 × 10 × 1 had been chosen. Instead of 10 nodes in the hidden layer, different number of nodes could have been chosen. That would give different performance of classification rate. The future research will involve in implementing an echostate neural network for classification of intrusion packet.

REFERENCES

Bace, R. and P. Mell, 2004. NIST Special Publication on Intrusion Detection Systems, http:// www. nist.gov, Retrieved-23-02-2004.

Bace, R., 2002. Intrusion Detection, Macmillan Technical Publishing.

Bershad, N.J., J.J. Shynk and P.L. Feintuch, 1993. Statistical analysis of the single-layer-back-propagation algorithm:part-I-Mean weight behavior. IEEE Transactions on Acoustics. Speech and Signal Processing, pp: 573-582.

Gordeev, M., 2004. Intrusion Detection Techniques and Approaches, http://www.ict.tuwein.ac.a. Retrieved. 12-03-2004.

Graham, R., 2000. FAQ: Network Intrusion Detection Systems, http://www.robertgraham.com, 21-04-2000.

Hirose, Y., K. Yamshita and S. Hijiya, 1991. Back-propagation algorithm which varies the number of hidden units. Neural Networks, pp: 61-66.

Kazienko, P. and P. Dorosz, 2003. Intrusion Detection Systems (IDS) part I-(network intrusions, attack symptoms, IDS tasks and IDS.

Lippmann, R.P., 1987. Introduction to computing with neural nets. IEEE. Tran. ASSP. Mag., 35, 4 (2): 4-22.

Philippe, J., 2004. Application of Neural Networks to Intrusion Detection, http://www.sans.org, Retrieved-23-02-2004.

Rumelhart, D.E., G.E. Hinton and R.J. Williams, 1986. Learning internal representations by error propagation. In: Rumelhart, D.E., J.L. McLELL and and the PDP research group (Eds.). Parallel distributed processing: Explorations in the Microstructure of Cognition, pp: 318-362.

Sundaram, A., 2001. An Introduction to Intrusion Detection, http://www.acm.org, Retrieved-23-1.

Verwoerd, T. and R. Hunt, 2001. Intrusion detection techniques and approaches. http://www.Elsevier.com.