Journal of Engineering Science and Technology
Vol. 5, No. 4 (2010) 457 - 471

ANOMALY NETWORK INTRUSION DETECTION SYSTEM BASED ON DISTRIBUTED TIME-DELAY NEURAL NETWORK (DTDNN)

LAHEEB MOHAMMAD IBRAHIM

National Advanced IPV6 Centre, 6th floor, school of computer sciences building, Universiti sains Malaysia, 11800 Penang, Malaysia
Email: dr.laheeb2007@yahoo.com

Abstract

In this research, a hierarchical off-line anomaly network intrusion detection system based on Distributed Time-Delay Artificial Neural Network is introduced. This research aims to solve a hierarchical multi class problem in which the type of attack (DoS, U2R, R2L and Probe attack) detected by dynamic neural network. The results indicate that dynamic neural nets (Distributed Time-Delay Artificial Neural Network) can achieve a high detection rate, where the overall accuracy classification rate average is equal to 97.24%.

Keywords: Anomaly, Intrusion detection system, Artificial neural network, Distributed time-delay artificial neural network.

1. Introduction

A single intrusion of a computer network can result in the loss or unauthorized utilization or modification of large amounts of data and causes users to question the reliability of all of the information on the network. There are numerous methods of responding to a network intrusion, but they all require the accurate and timely identification of the attack [1, 2].

Security policies or firewalls have difficulty in preventing such attacks because of the hidden weaknesses and bugs contained in software applications. Moreover, hackers constantly invent new attacks and disseminate them over the internet. Disgruntled employees, bribery and coercion also make networks vulnerable to attacks from the inside. Mere dependence on the stringent rules set by security personnel is not sufficient. Intrusion detection systems (IDS), which can detect, identify and respond to unauthorized or abnormal activities, have the potential to mitigate or prevent such attacks [3].

Abbreviations

ANN	Artificial neural network
DTDNN	Distributed time-delay neural network
GA	Genetic algorithm
IDS	Intrusion detection system

Intrusion detection systems (IDS) have emerged to detect actions which endanger the integrity, confidentiality or availability of a resource as an effort to provide a solution to existing security issues. This technology is relatively new, however, since its beginnings, an enormous number of proposals have been put forward to sort this situation out in the most efficient and cost effective of manners [4].

There are two general categories of attacks which intrusion detection technologies attempt to identify - anomaly detection and misuse detection, refer to Fig. 1. Anomaly detection identifies activities that vary from established patterns for users, or groups of users. Anomaly detection typically involves the creation of knowledge bases that contain the profiles of the monitored activities. The second general approach to intrusion detection is misuse detection. This approach involves the comparison of a user's activities with the known behaviors of attackers attempting to penetrate a system. While anomaly detection typically utilizes threshold monitoring to indicate when a certain established metric has been reached, misuse detection approach frequently utilize a rule-based approach. When applied to misuse detection, the rules become scenarios for network attacks. The intrusion detection mechanism identifies a potential attack if a user's activities are found to be consistent with the established rules. The use of comprehensive rules is critical in the application of expert systems for intrusion detection [1].

A number of approaches based on computing have been proposed for detecting network intrusions. The guiding principle of soft computing is exploiting the tolerance of imprecision, uncertainty, partial robustness and low solution cost. Soft computing includes many theories such as Fuzzy Logic (FL), Artificial Neural Networks (ANNs), Probabilistic Reasoning (PR), and Genetic Algorithms (GAs). When used for intrusion detection, soft computing is a general term for describing a set of optimization and processing techniques that are tolerant of imprecision and uncertainty. Soft computing is often used in conjunction with rule-based expert systems where the knowledge is usually in the form of if-then rules. Despite different soft computing based approaches having been proposed in recent years, the possibilities of using the techniques for intrusion detection are still underutilized [5-7].

Some early research on IDSs explored neural networks for intrusion detection. These can be used only after training on normal or attack behaviours, or combination of the two. Most supervised neural net architectures require retraining to improve analysis on varying input data, unsupervised nets, which offer greater adaptability, can improve their analysis capability dynamically [8].

The majority of currently existing IDS face a number of challenges such as low detection rates and high false alarm rates, which falsely classifies a normal

connection as an attack and therefore obstructs legitimate user access to the network resources. These problems are due to the sophistication of the attacks and their intended similarities to normal behavior. More intelligence is brought into IDS by means of Machine Learning (ML). Theoretically, it is possible for a ML algorithm to achieve the best performance, i.e. it can minimize the false alarm rate and maximize the detection accuracy. However, this normally requires infinite training sample sizes (theoretically). In practice, this condition is impossible due to limited computational power and real-time response requirement of IDS. IDS must be active at any time and they cannot allow much delay because this would cause a bottleneck to the whole network [9].

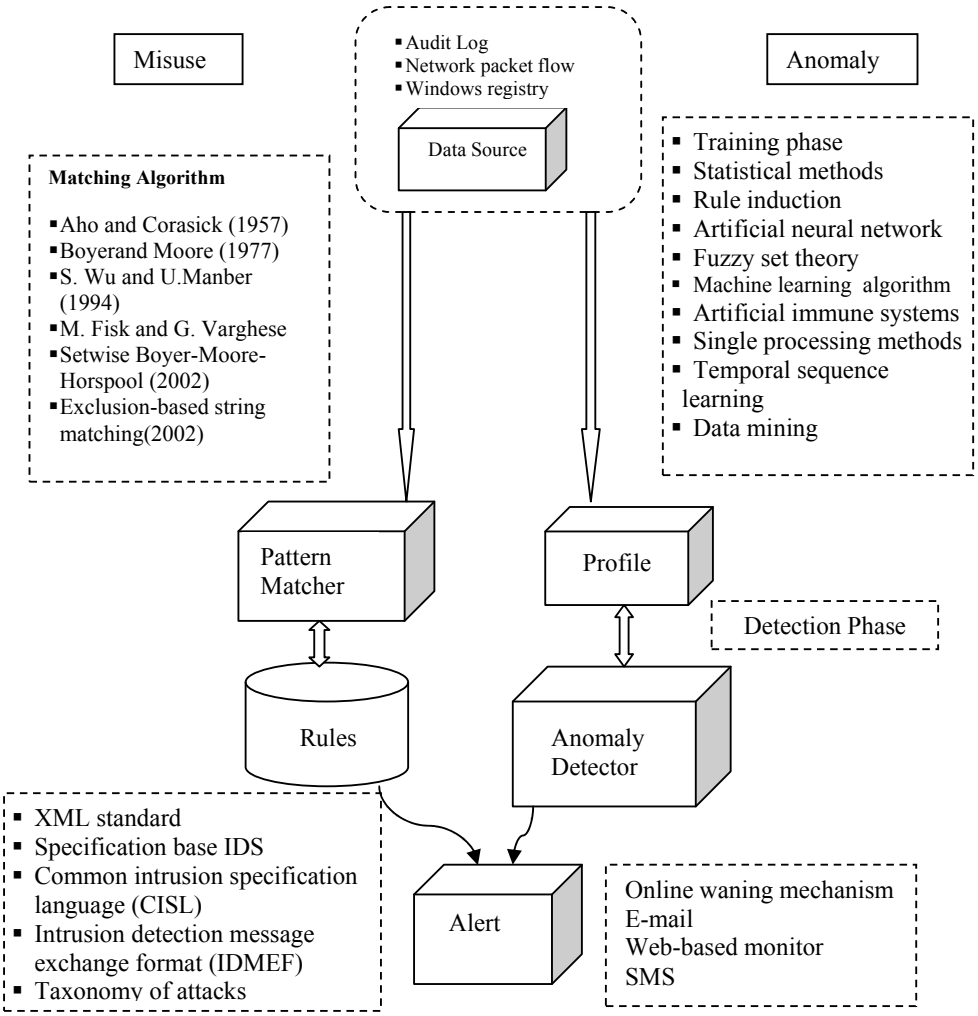

Fig. 1. The Flow Chart of Misuse Detection and Anomaly Detection Application [10].

To overcome low detection rate and high false alarm problems in currently existing IDS, we propose a hierarchical off line Anomaly intrusion detection system using Distributed Time-Delay Artificial Neural Network to enhance the performance of intrusion detection for rare and complicated attacks. In this paper, we introduce anomaly intrusion detection system, this can detect network-based attacks using dynamic neural nets, and has facilities for training, testing, and tuning of dynamic nets for intrusion detection purpose.

The remainder of the paper is organized as follows; Section 2 presents related works of intrusion detection systems with ANN. Section 3 introduces our proposal system. Section 4 shows the experiments and results and in Section 5 are the conclusions and future works.

2. Related Work with Artificial Neural Network

The goal for using ANNs for intrusion detection is to be able to generalize from incomplete data and to be able to classify data as being normal or intrusive. An ANN consists of a collection of processing elements that are highly interconnected. Given a set of inputs and a set of desired outputs, the transformation from input to output is determined by the weights associated with the interconnections among processing elements. By modifying these interconnections, the network is able to adapt to desired outputs. The ability of high tolerance for learning-by-example makes neural networks flexible and powerful in IDS [11].

Neural networks can easily represent non-linear relationships between input data and output data. Even if the data is incomplete, neural networks are able to correctly classify the different data classes captured from the network or other sources. An increasing number of researches have been conducted on intrusion detection based on neural networks. Neural-net-based IDSs can be classified into the following four categories [8], the first category MLFF neural-net-based IDSs includes the systems built on Multi-Layer Feed-Forward (MLFF) neural nets, such as the Multi-Layer Perceptron (MLP) and Back Propagation (BP). MLFF neural nets have been used in most early research in neural-net-based IDSs.

Works including [1, 3, 12, 13] used MLFF neural nets for anomaly detection based on user behaviours. Other researchers like [4, 6, 13, 14] have been used MLP to detect Anomaly IDSs and study the effective of using MLP in detecting anomaly IDSs.

InSeonin [15] in 2002 tried to integrate a smart detection engine into a firewall and detecting unusual structures in data packets uses a classical feed-forward multi-layer perceptron network: a back propagation neural network and time delay neural network to program-based anomaly detection. Also Byoung-Doo [16] in 2006 built IDS deals well various mutated attacks, as well as well-known attacks by using Time Delay Neural Network classifier that discriminates between normal and abnormal packet flows.

Other researchers have compared the effectiveness of MLFF neural nets to other neural nets, Siddiqui [17] in 2004 compared the effective of BP with Fuzzy ARTMAP, Grediaga [18] in 2006 compared the effective of MLFF with Self organization map (SOM), Zhang [19] in 2004 make comparison between BPL and RBF network in IDSs, and Vaitsekho [20] in 2009 compared effectives

between MLFF and recurrent neural network. MLFF neural nets have been shown to have lower detection performance than SOM.

The second category is recurrent and adaptive neural-net-based IDSs, this category includes systems built on recurrent and adaptive neural nets such as ELMAN and CMAC. By getting feedback from its output or its protected system, the neural net preserves the correlation of current system inputs with previous system inputs and states. Debar et al. [21] in 1999 used a simplified ELMAN recurrent net (GENT) and multi-layer recurrent net with back-propagation to predict the next acceptable command. Cannady in 2000 has applied the CMAC (Cerebellar Model Articulation Controller) net – a form of adaptive neural nets – to learn new attacks autonomously by modified reinforcement learning [22**].**

The third category; unsupervised neural-net-based IDSs uses unsupervised learning neural nets to classify and visualize system input data to separate normal behaviours from abnormal or intrusive ones. Most of the systems in this category use Self-Organizing Maps (SOMs), while a few use other types of unsupervised neural nets. Fox was the first to apply an SOM to learn the characteristics of normal system activity and identify statistical variations from the normal trends [23]. Rhodes et al. [24], Höglund et al. [25], Lichodzijewski et al. [26] and Ramadas [27] trained SOM on a collection of normal data from UNIX audit data and used it for detecting anomalous user activity**.**

Hybrid neural-net-based IDSs is last category of neural-net-based IDSs encompasses systems that combine supervised and unsupervised neural nets. Jirapummin [28] proposed employing hybrid neural network for both visualizing intrusions using Kohenen's SOM and classifying intrusions using a Resilient Propagation neural network (RPROP). Horeis [29] used a combination of SOM and Radial Basis Function (RBF) nets. The system offers generally better results than IDSs based on RBF nets alone. Integration and combination of neural-net-based IDSs (as an intelligent component in detecting variations of known and especially unknown attacks), with other preventive techniques such as firewalls and access control is a new research area. A sample of this research has been introduced by InSeon and Ulrich [6]. The main purpose of their research was integrating a smart detection engine (based on neural nets) into a firewall. The presented system not only detects anomalous network traffic as in classical IDSs, but also detects unusual structures in data packets that suggest the presence of virus data [8].

The idea of designing a flexible IDS system was conceived for applying more complicated types of supervised neural nets which probably have higher capability in intrusion detection and to solve a hierarchical multi class problem in which the type of attack (DoS, U2R, R2L and Probe attack) detected by dynamic neural network.

This system was constructed to provide the facilities for tuning, testing, and applying dynamic Distributed Time-Delay neural nets in intrusion detection. The system was used to detect the malicious attacks in the network.

3. Proposed Intrusion Detection System

The proposed intrusion detection system, as shown in Fig. 2, consists of the following modules

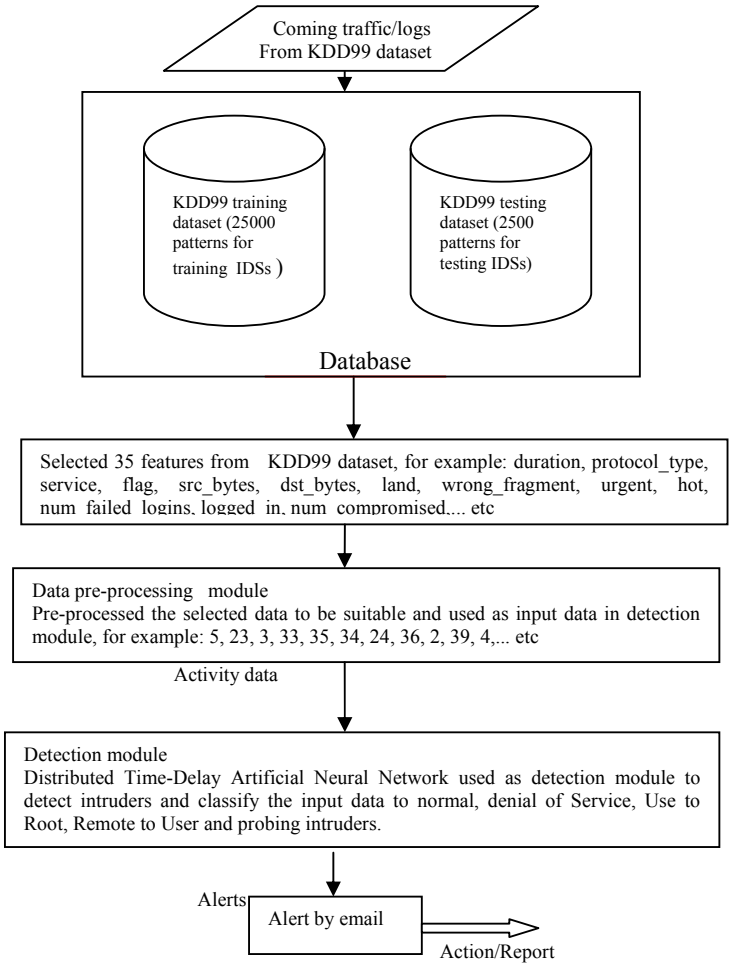

Fig. 2. Structure Intrusion Detection System.

3.1. Data pre-processor module

The first module of proposed IDS is data pre- processor that means collects and formats the data to be analyzed by the detection algorithm. In proposed IDSs, KDD99 is used as database to train and test the system performance; the KDD99 data is original from 1998 DARPA Intrusion Detection Evaluation. Under the sponsorship of Defense Advanced Research project s Agency (DARPA) and Air Force Research Laboratory (AFRL), MIT Lincoln Labs has collected and distributed the datasets for the evaluation of computer network intrusion detection system. [7, 10, 13, 30, 31].

The first step of preprocessing is to select features from KDD99 dataset, the features of the dataset have been seen below and divided into three sets as in [6, 14], these sets are: Features describing the commands used in the connection (instead of

the commands themselves), features describing the connection specifications and features describing the connection to the same host in last 2 seconds

35 features are selected (e.g. duration, protocol-type, service … etc, see appendix (A)) from KDD99 data packets were selected as in [6] because they are typically present in network data packets and they provide a complete description of the information transmitted by the packet. The second step of preprocessing is to convert the 35 features into standardized numeric representation, (Table 1). A 36'Th element was assigned to each record based on a determination of whether this event represented part of an attack on a network; this element was used during training as target output of the neural network for each record.

Table 1. 35 Features Selected from KDD99 Data Packet and Target Element.

Feature	**Duration**	**Protocol_ type**	**Service**	**Flag**	**src_bytes**
First step of preprocessing	0	tcp	http	SF	181
Second step of preprocessing	0	3	19	10	181
feature	dst_bytes	land	wrong_fragment	urgent	hot
First step of preprocessing	5450	0	0	0	0
Second step of preprocessing	5450	0	0	0	0
feature	num_failed_ logins	logged_in	num_ compromised	root_shell	su_attempted
First step of preprocessing	0	1	0	0	0
Second step of preprocessing	0	1	0	0	0
feature	num_root	num_file_ creations	num_shells	num_access _files	num_outbound _cmds
First step of preprocessing	0	0	0	0	0
Second step of preprocessing	0	0	0	0	0
feature	is_host_login	is_guest_login	count	srv_count	serror_rate
First step of preprocessing	0	0	8	8	0.00
Second step of preprocessing	0	0	8	8	0
feature	srv_serror _rate	rerror_rate	srv_rerror _rate	same_srv_rate	diff_srv_rate
First step of preprocessing	0.00	0.00	0.00	1.00	0.00
Second step of preprocessing	0	0	0	1	1
feature	srv_diff_ host_rate	dst_host_coun t	dst_host_srv _count	dst_host_same _ srv_rate	dst_host_diff_ srv_rate
First step of preprocessing	0.00	0.00	9	0.00	0.00
Second step of preprocessing	0	0	9	0	0
Target data First step of preprocessing		Target data Normal	Target data Second step of preprocessing		Target data 0

3.2. Detection module

The most important component of proposed IDSs is a detection module whose function is to analyse and detect intrusion using artificial neural network. Neural

net used as detection module because of the utilization of a neural network in the detection of intrusion would be the flexibility that the network would provide. A neural network would be capable of analyzing the data from the network, even if the data is incomplete or distorted. Similarly, the network would possess the ability to conduct an analysis with data in a non-linear fashion. Both of these characteristics are important in a networked environment where the information which is received is subject to the random failings of the system. Further, because some attacks may be conducted against the network in a coordinated assault by multiple attackers, the ability to process data from a number of sources in a non-linear fashion is especially important. The inherent speed of neural networks is another benefit of this approach. Because the protection of computing resources requires the timely identification of attacks, the processing speed of the neural network could enable intrusion responses to be conducted before irreparable damage occurs to the system [1]. In this paper Distributed Time-Delay Neural Network (DTDNN) is used as detection module in IDSs.

3.2.1. Why distributed time-delay neural network (DTDNN)

DTDNN provides a simple and efficient way of classifying data sets. To process data for classification we believe that DTDNN are best suited due to their high speed and fast conversion rates as compared with other learning techniques. Also DTDNN preserves topological mappings between representations, a feature which is desired when classifying normal v.s. intruder behavior for network data. That is, the relationships between senders, receivers and the protocols them, which are the primary features that we use, are preserved by the mapping. A DTDNN is it dynamic networks are generally more powerful than static networks because dynamic networks have memory, they can be trained to learn sequential or time-varying patterns. This has applications in such disparate areas as prediction in financial markets, channel equalization in communication systems, phase detection in power systems, sorting, fault detection, speech recognition, and even the prediction of protein structure in genetics. But static (feed forward) networks have no feedback elements and contain no delays; the output is calculated directly from the input through feed forward connections. The training of static networks was discussed in Backpropagation. In dynamic networks, the output depends not only on the current input to the network, but also on the current or previous inputs, outputs, or states of the network [32].

3.2.2. Distributed time-delay artificial neural network structure

Each layer in the Distributed Time-Delay Artificial Neural Network is made up of the following parts:

- Set of weight matrices that come into that layer (which can connect from other layers or from external inputs), associated weight function rule used to combine the weight matrix with its input (normally standard matrix multiplication), and associated tapped delay line.
- Bias vector
- Net input function rule that is used to combine the outputs of the various weight functions with the bias to produce the net input (normally a summing junction)
- Transfer function

The network has inputs that are connected to special weights, called input weights, and denoted by IWi,j, where j denotes the number of the input vector that enters the weight, and i denotes the number of the layer to which the weight is connected. The weights connecting one layer to another are called layer weights and are denoted by LWi,j, where j denotes the number of the layer coming into the weight and i denotes the number of the layer at the output of the weight [32].

3.2.3. Architecture of distributed time delay artificial neural network

A two layer Distributed Time-Delay Artificial Neural Network structure is used to detect attackers (DoS, U2R, R2L and Probe). The 35 features from KDD99 datasets are used for input data, The DTDNN transform 35-dimensional input data vector into 5 dimensional output vectors (0 if entrance pattern is not attack, and 4 values for attackers (1 for DoS, 2 for U2R, 3 for R2L, 4 for Probe). The DTDNN processes those given data to recognize type of attaches or normal transactions. Figure 3 illustrates the architecture and parameters used in simulation process, we determined the best values of important parameters for DTDNN by doing primary experiments were carried out and the values of Fig. 3 were achieved.

Input nodes	35	**Hidden nodes**	5	**Output nodes**	5
Learning Rate	0.9	**Epoch**	405 iteration from 5000	**Performance mean squared error**	0.001
Momentum constant	0.7	**Time**	0.00.4	**Transfer function for hidden layer**	Hyperbolic tangent sigmoid transfer function (tansig)
Transfer function for output layer	Linear transfer function (purelin)	**Network training function**	Levenberg-Marquardt backpropagation (trainlm)	**Weight/bias function**	Gradient descent with momentum (learngdm)

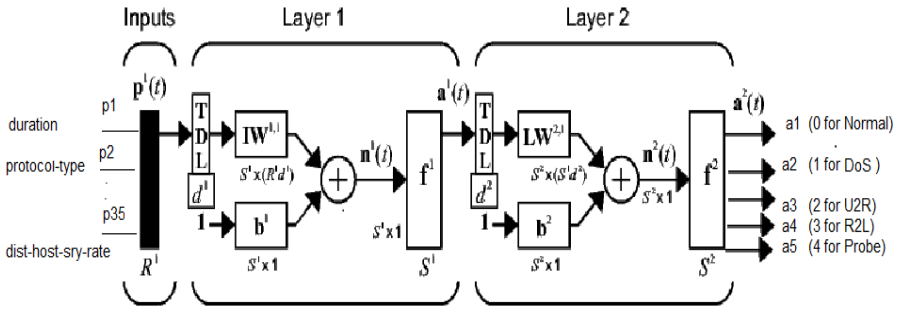

Fig. 3. Distributed Time Delay Neural Network Structure.

(This figure represents DTDNN structure from MATLAB 7.6.0 (R2008a) neural network Toolbox software [32]).

3.3. Alert filter

Depending on the outcome from the detection module, is taking the necessary precautions and to take quick decision to stop the intruder to penetrate to the computer network, in proposed IDSs we used email to send a warning to stop the intruder.

4. Experiment and Results

In this section, we summarize our experimental results to detect Anomaly intrusion detections using Distributed Time-Delay Artificial Neural Network over KDD99 dataset. The full training set of the KDD99 dataset has 4,898,431 connections covering normal network traffic and four categories of attacks [6, 13, 27, 33]:

- **Denial of Service (DoS):** A DoS attacks is a type of attack in which the hacker makes memory resources too busy to serve legitimate networking requests and hence denying users access to a machine.
- **User to Root Attacks (U2R):** Unauthorized access to local root privileges.
- **Remote to User attack (R2L): A**n attacker sends packets to a machine over a network, then exploits machine's vulnerability to illegally gain local access as a user.
- **Probing**: Attacker tries to gain information about the target host.

For our experiments, the training dataset consist of 25000 patterns (5000 patterns for each class of DoS, R2L, U2R, Probe, Normal), and testing dataset consist of 2500 patterns (500 patterns for each class). We are only interested in knowing to which category (Normal, DoS, R2l, U2R, Probe) a given connection belonged. The accuracy of each experiment is based on percentage of successful classification (PSC) on train and test dataset, where

PSC= ***(number of correctly classified instance / number of instance in the test dataset)***

The DTDNN network was trained until the desired mean square error of 0.001 was met, during the training process the goal was met at 405 epochs for Distributed Time-Delay. Table 2 show the performance of the neural network training algorithm, the bottom row shows that overall accuracy classification is 99.884% for Distributed Time-Delay.

Table 2. Training Performance for Distributed Time-Delay ANN.

Class name	Distributed Time-Delay		
	Number of test patterns for each Class	Number of correctly classified patterns	percentage of successful classification (PSC)
Normal	5000	5000	100%
DoS	5000	4990	99.8%
R2l	5000	4993	99.86%
U2R	5000	4999	99.98%
Probe	5000	4989	99.78%
Overall Accuracy Classification Rate Average = (PSC(normal)+ PSC(DoS) + PSC(R2L)+ PSC(U2R)+ PSC(Probe)) / 5)			99.884%

Table 3 shows that by using DTDNN, the last column indicate that the PSC is 98.4% of the actual 'Normal' data points were detected correctly. In the same way PSC for 'DoS' 97.6% , 'Probe ' 98.2% , 'R2l' 95.8% and 'U2R' 96.2% of actual 'attack' test were correctly detected. The bottom row shows that the Overall classification rate average was 97.24%. Its detection rates (PSC) on deferent attack categories are displayed in Fig. 4. We could discover a general trend of increasing performance as more intrusions are added into training set.

Several recently published resulted and our results on the same dataset are listed in Table 4. We can find that our IDSs are greatly competitive with other and Fig. 5 indicates that our system has possibilities for detection and classification computer attacks. Distributed Time-Delay Artificial Neural Network is implemented using MATLAB 7.6.0 (R2008a) neural network Toolbox software.

Table 3. Performance of the Distributed Time-Delay ANN.

Class name	Number of test patterns for each Class	Number of correctly classified patterns	percentage of successful classification (PSC)
Normal	500	492	98.4%
DoS	500	488	97.6%
R2l	500	479	95.8%
U2R	500	481	96.2%
Probe	500	491	98.2%
Overall Classification Rate Average = (PSC(normal)+ PSC(DoS) + PSC(R2L)+ PSC(U2R)+ PSC(Probe)) / 5)			97.24%

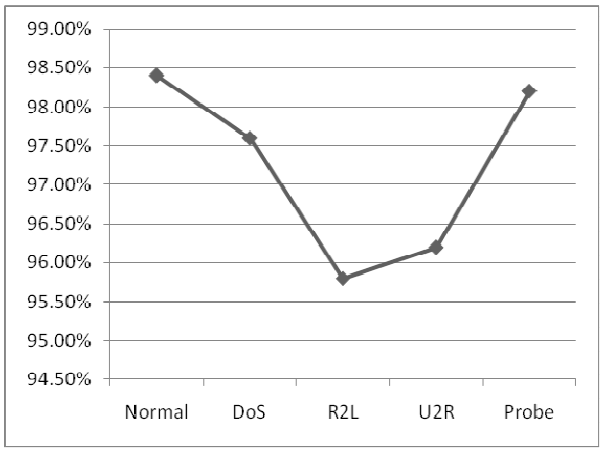

Fig. 4. Detection Rate on Dataset for Anomaly Detection System.

Table 4 . Compression for Intrusion Detection Systems Using ANN.

Research	ANN type	Database	percentage of successful classification (PSC) for test dataset
[1] Cannady, 1998	MLFF	RealSecure™ network monitor	91%
[27] Ramadas M., 2003	(SOM	DARPA	95.42%
[14] Moradi M., 2004	2 hidden layers MLP	DARPA	91%
[17] Siddiqui M., 2004	backpropagation and fuzzy ARTMAP	DARPA	81.37% for BP and 80.52% for fuzzy ARTMAP (overall PSC = 80.945)
[34] Mukkamalaa S., 2005	Backpropagation	DARPA	97.04%
[18] Grediaga, A , 2006	(MLP and a SOM	DARPA	For MLP is 94.2997% and for SOM is 99.01%
[6] Sammany M., 2007	2 hidden layers MLP	DARPA	93.43% (overall PSC = 96.65)
[20]Vaitsekhovich L.,2009	RNN and MLP.	KDD-99	Detection for DoS (94.20%) , U2R (86.54%) ,R2L (85.59%) , Probe (97.78%), Normal (85.22%) (overall PSC = 89.886)
Proposed IDSs	DTDNN	KDD-99	Detection for DoS (97.6 %), U2R (96.2%) , R2L (95.8%) Probe (98.2%), Normal (98.4%) (overall PSC = 97.24)

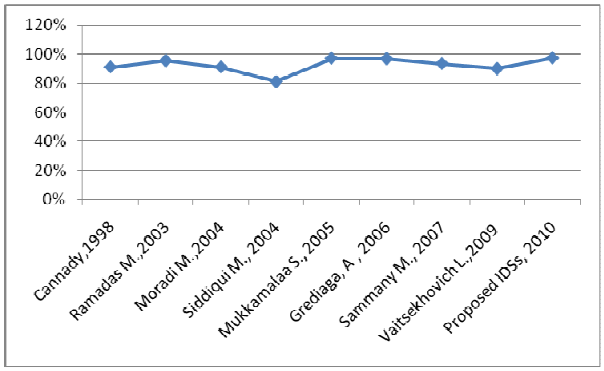

Fig. 5. Detection Rate on Dataset for IDSs.

5. Conclusions

In this paper, we presented a practical solution to using dynamic supervised artificial neural network in hierarchical anomaly intrusion detection system. The system is able to employ dynamic supervised neural nets for classifying and separating normal traffic from the attack traffic (DoS, R2L, U2R, Probe).

The proposed system was used to tuning, training, and testing Distributed Time Delay neural network in intrusion detection. Evaluation of the DTDNN efficiency in

anomaly intrusion detection was performed detection performance, the result show that DTDNN in 97.24% were able to recognized attack traffic (PSC for DoS (97.6 %), U2R (96.2%), R2L (95.8%) Probe (98.2%) from normal one (Normal (98.4%)).

Experiments on the KDD99 network intrusion dataset show that DTDNN are best suited due to their high speed and fast conversion rates as compared with other learning techniques and a DTDNN are more powerful than static networks because dynamic networks have memory, they can be trained to learn sequential or time-varying patterns, and also show that our approach by using DTDNN obtains superior performance in comparison with other state-of-the-art detection methods.

In the future, we will hope to detected attackers in each class of (DoS, R2L, U2R, Probing) combine Artificial neural network methods and fuzzy logic to improve the accuracy of IDS.

References

1. Cannady J. (1998). Artificial neural networks for misuse detection. *Proceedings of the 1998 National Information Systems Security Conference (NISSC'98)*, 443-456, Arlington, VA.
2. Lippmann, R.; Haines, J.; and Zissman, M. (2003). *An overview of issues in testing intrusion detection systems*. National institute of standards and technology (NTIS).
3. Chen, W.H.; Hsu, S.H.; and Shen, H.P. (2005). Application of SVM and ANN for intrusion detection. *Computers & Operations Research*, 32(10), 2617–2634.
4. Lorenzo-Fonseca, I.; Maciá-Pérez, F.; Mora-Gimeno, F.; Lau-Fernández1, R.; Gil-Martínez-Abarca, J.; and Marcos-Jorquera, D. (2009). Intrusion detection method using neural networks based on the reduction of characteristics. *LNCS,* 5517, 1296–1303.
5. Mukkamala, S. (2002). Intrusion detection using neural networks and support vector machine. *Proceedings of the 2002 IEEE International Honolulu*, HI.
6. Sammany, M.; Sharawi, M.; El-Beltagy, M.; and Saroit, I. (2007). Artificial neural networks architecture for intrusion detection systems and classification of attacks. *Accepted for publication in the 5th international conference INFO2007*, Cairo University.
7. Selvakani, S.; and Rajesh, R.S. (2009). Escalate intrusion detection using GA–NN. *International Journal of Open Problems in Computer Science and Mathematics*, 2(2), 272-284.
8. Morteza, A.; Jalili, R.; and Hamid R.S. (2006). RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks. *Computers & Security*, 25(6), 459 – 468.
9. Tran. T.P.; Cao, L.; Tran, D.; Nguyen, C.D. (2009). Novel intrusion detection using probabilistic neural network and adaptive boosting. *International Journal of Computer Science and Information Security (IJCSIS)*, 6(1), 83-91.
10. Chen, R.C.; Cheng, K.F.; and Hsieh, C.F. (2009). Using rough set and support vector machine for network intrusion detection. *International Journal of Network Security & Its Applications (IJNSA)*, 1(1), 1-13.
11. Lia, L.B.; Chang, R.I.; Kouh, J.S. (2009). Detecting network intrusions using signal processing with query-based sampling filter. *Hindawi Publishing*

Corporation, EURASIP Journal on Advances in Signal Processing, 2009, Article ID 735283, 1-8.
12. Alfantookh, A.A. (2006). DoS attacks intelligent detection using neural networks. *Comp. & Info. Sci.*, 18, 27-45.
13. Mukkamalaa, S.; Sung, A.H.; and Abraham, A. (2005). Intrusion detection using an ensemble of intelligent paradigms. *Journal of Network and Computer Applications*, 28(2), 167-182.
14. Moradi, M.; and Zulkernine, M. (2004). A neural network based system for intrusion detection and classification of attacks. *IEEE International Conference on Advances in Intelligent Systems - Theory and Applications*, Luxembourg-Kirchberg, Luxembourg.
15. InSeon Y.; and Ulrich U. (2002). An intelligent firewall to detect novel attacks? An integrated approach based on anomaly detection against virus attacks, Mária Bieliková (Ed.): *SOFSEM 2002 Student Research Fo*rum, 59–64, 2002. http://www2.fiit.stuba.sk/~bielik/sofsem2002srf/clanky/09Yoo.pdf.
16. Kang, B.D.; Lee, J.W; Kim, J.H.; Kwon, O.H.; Seong, C.Y.; Park, S.M.; and Kim, S.K. (2006). A mutated intrusion detection system using principal component analysis and time delay neural network. *LNCS*, 3973, 246 – 254.
17. Siddiqui, M.A. (2004). *High performance data mining techniques for intrusion detection*. Master's Thesis, University of Engineering & Technology, School of Computer Science, College of Engineering & Computer Science at the University of Central Florida.
18. Grediaga, A.; Ibarra, F.; García, F.; Ledesma, B.; and Brotons, F. (2006). Application of neural networks in network control and information security. *LNCS*, 3973, 208–213.
19. Zhang, C.; Jiang, J.; and Kamel, M. (2004). Comparison of BPL and RBF Network in intrusion detection system. *LNCS (LNAI)*, 2639, 460–470.
20. Vaitsekhovich, L. (2009). *Intrusion detection in TCP/IP networks using immune systems paradigm and neural network detectors*. Brest State Technical University, XI International PhD Workshop, OWD 2009. http://www.cs.ucc.ie/misl/publications/files/idssteinebach.pdf.
21. Debar, H.; Becker, M.; and Siboni, D. (1992). A neural network component for an intrusion detection system. *IEEE Computer Society Symposium on Research in Security and Privacy*, 240-250.
22. Cannady J. (2000). Applying CMAC-based online learning to intrusion detection. *International Joint Conference on Neural Networks, 2000. IJCNN 2000*, *Proceedings of the IEEE-INNS-ENNS*, 5, 405-410.
23. Fox, K.L.; Henning, R.R.; and Reed, J.H. (1990). A neural network approach towards intrusion detection. *In Proceedings of the 13th National Computer Security Conference.*
24. Rhodes, B.C.; Mahaffey J.A.; and Cannady, J.D. (2000). Multiple self-organizing maps for intrusion detection. *Proceedings of the 23rd National Information Systems Security Conference.*
25. Höglund, A.J.; Hätönen, K.; and Sorvari, A.S. (2000). A computer host-based user anomaly detection system using the self-organizing map. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks (IJCNN'00)*, 5, 411-416.
26. Lichodzijewski, P.; Zincir-Heywood, A.N.; and Heywood, M.I. (2002). Host-based intrusion detection using self-organizing maps. *Proceedings of the 2002 International Joint Conference on Neural Networks, 2002. IJCNN '02*, 1714-1719.

27. Ramadas, M.; Ostermann, S.; and Tjaden, B. (2003). Detecting anomalous network traffic with self-organizing maps. *LNCS*, 2820, 36–54.
28. Jirapummin, C.;Wattanapongsakorn, N.; and Kanthamanon, P. (2002). hybrid neural networks for intrusion detection system. *Proceedings of the 2002 International Technical Conference On Circuits/Systems, Computers and Communications.*
29. Horeis, T. (2003). Intrusion detection with neural network – Combination of self-organizing maps and redial basis function networks for human expert integration. http://ieee-cis.org/_files/EAC_Research_2003_Report_Horeis.pdf.
30. 19.DARPA1998 http://www.ll.mit.edu/IST/ideval/docs/1998/introduction/index.htm.
31. KDDCup1999 : http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
32. MATLAB 7.6.0 (R2008a) neural network Toolbox software.
33. Panda, M.; and Patra, M.R. (2007). Network intrusion detection using naïve bayes. *International Journal of Computer Science and Network Security (IJCSNS)*, 7(12), 258-263.
34. Mukkamala, S.; and Sung, A.H. (2003). Feature selection for intrusion detection using neural networks and support vector machines. *Transportation Research Record*, 1822, 33-39.

Appendix A

35 Features of KDD99 Dataset

Basic features of individual TCP connection

Feature name	Description	Type
duration	length (number of seconds) of the connection	continuous
protocol_type	type of the protocol, e.g. (TCP, UDP, ICMP and unknown)	discrete
service	network service on the destination, e.g.(http, telnet, etc.)	discrete
src_bytes	number of data bytes from source to destination	continuous
dst_bytes	number of data bytes from destination to source	continuous
flag	normal or error status of the connection	discrete
land	1 if connection is from/to the same host/port; 0 otherwise	discrete
wrong_fragment	number of ``wrong'' fragments	continuous
urgent	number of urgent packets	continuous

Content features within a connection suggested by domain knowledge.

Feature name	Description	Type
hot	number of ``hot'' indicators	continuous
num_failed_logins	number of failed login attempts	continuous
logged_in	1 if successfully logged in; 0 otherwise	discrete
num_compromised	number of ``compromised'' conditions	continuous
root_shell	1 if root shell is obtained; 0 otherwise	discrete
su_attempted	1 if ``su root'' command attempted; 0 otherwise	discrete
num_root	number of ``root'' accesses	continuous
num_file_creations	number of file creation operations	continuous
num_shells	number of shell prompts	continuous
num_access_files	number of operations on access control files	continuous
num_outbound_cmds	number of outbound commands in an ftp session	continuous
is_hot_login	1 if the login belongs to the ``hot'' list; 0 otherwise	discrete
is_guest_login	1 if the login is a ``guest''login; 0 otherwise	discrete

Traffic features computed using a two-second time window.

Feature name	Description	Type
count	number of connections to the same host as the current connection in the past two seconds	continuous
Srv_cont	number of connections to the same services as the current connection in the past two seconds	
serror_rate	% of connections that have ``SYN'' (S0)errors	continuous
rerror_rate	% of connections that have ``REJ'' errors	continuous
same_srv_rate	% of connections to the same service	continuous
diff_srv_rate	% of connections to different services	continuous
srv_diff_host_rate	% of connections to different hosts	continuous
dst_host_count	number of connection having the same destination host	continuous
dst_host_srv_count	Number of connection having the same destination host using the same service	continuous
srv_serror_rate	% of connections that have ``SYN'' errors	continuous
srv_rerror_rate	% of connections that have ``REJ'' errors	continuous
dst_host_same_srv_rate	% of connection having the same destination host using the same service	continuous
dst_host_diff_srv_rate	% of different services on the current host	continuous