International Journal of Innovative Research in Advanced Engineering (IJIRAE) ISSN: 2349-2163
Volume 1 Issue 10 (November 2014) www.ijirae.com

Using Learning Vector Quantization (LVQ) in Intrusion Detection Systems

Ezat mahmoud soleiman
Dept of ICT Eng, Maleke Ashtar Univ. of Tech

Abdelhamid fetanat
Dept of ICT Eng, Maleke Ashtar Univ. of Tech

Abstract— Attacks on computer infrastructure are becoming an increasingly serious problem nowadays, and with the rapid expansion of computer networks during the past decade, computer security has become a crucial issue for protecting systems against threats, such as intrusions. Intrusion detection is an interesting approach that could be used to improve the security of network system. Different soft-computing based methods have been proposed in recent years for the development of intrusion detection systems. This paper presents a Learning Vector Quantization artificial neural network to detect intrusion. A Supervised Learning Vector Quantization (LVQ) was trained for the intrusion detection system; it consists of two layers with two different transfer functions, competitive and linear. Competitive (hidden) and output layers contain a specific number of neurons which are the sub attack types and the main attack types respectively. The experiments and evaluations of the proposed method have been performed using the NSL-KDD 99 intrusion detection dataset.

Keywords—intrusion detection system,IDS, learning vector quantization, LVQ.

I. INTRODUCTION

Nowadays computers and the Internet are used almost in every part of our lives. Since the personal computer was invented it has been growing faster and faster and it is now impossible to imagine companies, universities or even a little shop that does not keep all the data of their customers, purchases and inventory in an electronic database or computer.

With the possibility of connecting several computers and networks was born the necessity of protecting all this data and machines from attackers (hackers) that would like to get some confidential information to use for their own benefit or just destroy or modify valuable information.

There are several security measures available to protect the computer resources of a company or a home user, but even if all expert recommendations are followed, our systems will never be safe against possible successful attacks. It is very difficult to get an invulnerable system, probably impossible and one may need to spend a lot of money designing and developing it. In companies, a very isolated system could drastically reduce productivity and for a not very experienced home user it may become a "hating technology" disease. For all these reasons the user or the security department should know what their values are, if they need to be protected and how much it costs, doing Risk Analysis [1].

According to the Edward Amoroso [2] the intrusion is "sequence of related actions by malicious adversary that results in the occurrence of unauthorized security threats to a target computing or networking domain".

An intrusion is considered as a sequence because it propagates under a current period. Actions causing the intrusion must be related, since unrelated ones are not of interest. As an intruder always has an attention to make an intrusion, he must be considered as a malicious adversary. Assuming there is one defined security policy, unauthorized security threat, is its violation.

A good security policy and a good risk analysis with well-educated users will make the system more secure to intrusions. An intrusion in the system will try to compromise one of the three main aspects in computer security.

- Confidentiality: the intruder has access to confidential information.
- Integrity: information can be modified or altered by the attacker.
- Availability: the system gets blocked so it cannot be used normally.

These three aspects are illustrated in figure1.

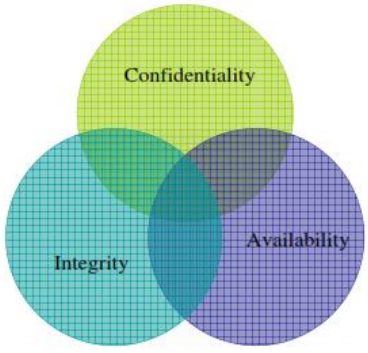

FIG. 1. THE THREE MAIN COMPROMISED ASPECTS

II. WHY USE AN IDS?

Intrusion detection allows protecting organization systems against threats that appear with increasing network connectivity and the interdependency of information systems [3].

IDSs have gained acceptance as a main part of the security infrastructure within an organization. There are several reasons for acquiring and using an IDS:

1- Avoid problems by dissuading hostile individuals
2- Detect attacks and other security violations not prevented by other protection measures.
3- Detect attack preambles
4- Record the organization risk
5- Provide useful information about the intrusions currently taking place

III. TYPES OF COMPUTER ATTACKS

The standard attack classification that is widely used is by Kendall and Gollmann [3][4]. The attacks are grouped into five major categories.

A. *Denial of service (DOS)*

The example attacks include: Apache2arppoison, Back, Crashiis, dosnuke, Land, Mail-bomb, SYN Flood (Neptune), Ping of Death (POD), Process Table, selfping, Smurf, sshprocesstable, Syslogd, tcpreset, Teardrop, Udpstorm.

B. *User to root (U2R)*

U2R refers to a class of exploit in which the attacker breaks into the system as the normal user then eventually completely control the machine as the root user. The example attacks include anypw, casesen, Eject, Ffbconfig Fdformat, Loadmodule, ntfsdos, Perl, Ps, sechole, Xterm, yaga.

C. *Remote to local (R2L)*

R2L refers to the exploits that start from remote network-based access which intend to break into the machine and obtain the user account. Example attacks include: Dictionary Ftpwrite Guest, Httptunnel, Imap, Named, ncftp, netbus, netcat, Phf, ppmacro, Sendmail, sshtrojan, Xlock, Xsnoop

D. *Probes*

The example attacks include: insidesniffier, Ipsweep, ls domain,Mscan, NTinfoscan Nmap, queso, resetscan, Saint, Satan.

E. *Data,*

The example attacks include: Secret.

F. *Trojan horses / worms – attacks:*

That are aggressively replicating on other hosts

IV. LEARNING VECTOR QUANTIZATION

Learning vector quantization (LVQ) is a method for training competitive layers in a supervised manner (with target outputs). A competitive layer automatically learns to classify input vectors. However, the classes that the competitive layer finds are dependent only on the distance between input vectors. If two input vectors are very similar, the competitive layer probably will put them in the same class. There is no mechanism in a strictly competitive layer design to say whether or not any two input vectors are in the same class or different classes [5]. LVQ networks, on the other hand, learn to classify input vectors into target classes chosen by the user.

An LVQ network has a first competitive layer and a second linear layer. The competitive layer learns to classify input vectors in much the same way as the competitive layers of Self Organizing Feature Maps. The linear layer transforms the competitive layer's classes into target classifications defined by the user. The classes learned by the competitive layer are referred to as subclasses and the classes of the linear layer as target classes [5].

Both the competitive and linear layers have one neuron per (sub or target) class. Thus, the competitive layer can learn up to S1 subclasses. These, in turn, are combined by the linear layer to form S2 target classes. (S1 is always larger than S2) [5].

In the training process of LVQ different computational paradigms were used. First we have mentioned before that LVQ only consists of one competitive (hidden) layer and one output layer containing sub-class and main-class neurons respectively. Therefore in the competitive and output layers 23 (normal and 22 sub attack types) and 5 neurons were used respectively.

During the training of LVQ it was clearly that LVQ highly affected about how many patterns corresponding for each main class, therefore in designing the training dataset it was very critical to select approximately equal numbers of patterns to represent each class otherwise the LVQ will classify one main class and neglects the others.

Learning rate and number of epochs was selected iteratively where the best performance is the main criteria.

After the training process, the LVQ is ready to classify the testing dataset. LVQ will classify the dataset into five classes (Normal, Dos, U2R, R2L and Prob). Then the results of LVQ will be combined later with the results of the k-Nearest Neighbor classifier to provide maximum classification rates.

V. EXPERIMENT RESULTS

In this paper learning vector quantization (LVQ) and k-Nearest Neighbor was used to detect intrusion. NSL-KDD99 dataset [6] was used in this paper. Training dataset was used to tune the weights and testing dataset was used for the network evaluation. Testing set contains some attacks that it is not represented in the training set. The testing dataset details are shown in the table below:

TABLE 1. TESTING DATASET DETAILS

Dataset	Testing Dataset
Normal	1000
Dos	1200
U2R	37
R2L	500
Prob.	1200
All	3937

LVQ network as the first classifier was trained using the parameters shown in the table below:

TABLE 2. THE DESIGNED LVQ TRAINING PARAMETERS

Parameters	Details
Learning	Supervised
Input Nodes	Input Dimensionality: 41
Hidden Nodes	Number of sub-classes: 23
Output Nodes	Number of Main classes: 5
Distance Function	Negative Euclidean Distance (**negdist**)
Transfer Function in the Hidden Layer	Competitive Transfer Function (**compet**)
Transfer Function in the output layer	Linear Transfer Function (**purelin**)
Learning Function	Learning Vector Quantization 1
Training Function	Random Weight/Bias Rule
Learning Rate	0.09
Number of epochs	6
Network Performance	Mean Square Error (**MSE**)

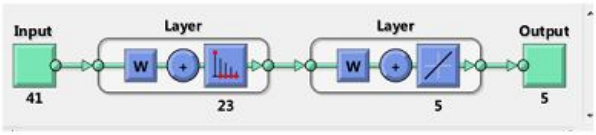

FIGURE 2. LVQ ARCHITECTURE WITH 23 HIDDEN NODES

Above figure represents the network model architecture that includes 41 input nodes which is the data dimensionality, 23 nodes in the competitive (hidden) layer that represents the subclass number and finally 5 nodes in the output (linear) layer which represents the number of main classes. The output should be either normal or main attack types (DoS, U2R, R2L & Prob.).

Following tables demonstrate the confusion matrix and the detection rate for each class using LVQ:

TABLE 3. LVQ CONFUSION MATRIX RESULTS

	LVQ Results					
Target	**1**	**2**	**3**	**4**	**5**	**Total**
1	387	39	478	89	7	1000
2	28	966	0	206	0	1200
3	2	1	30	4	0	37
4	248	8	152	53	39	500
5	0	1039	0	11	150	1200
Total	**665**	**2053**	**660**	**363**	**196**	**3937**

The detection rates for each class varied using LVQ, as shown in the table above the detection rates for normal, R2L and Prob was severely low but on the other hand the detection rate for the U2R attack was very good considering that this specific attack is very difficult to detect because the training dataset doesn't have enough records for this attack, therefore 81% is a very good result. Finally DoS attack has a good detection rate with 81%. The results of five classes will be combined with the results of the k-Nearest Neighbor classifier and the detection rates for all five classes will either rise up or stay the same.

As the second classifier k-Nearest Neighbor machine learning algorithm was used. First norm and second norm (Euclidean distance) were used in implementing kNN. First Norm demonstrates better result than the second norm (Euclidean distance). Tables below represent the confusion matrix and detection rates for the second norm (Euclidean distance).

TABLE 4. LVQ CLASS DETECTION RATES RESULTS

Dataset	Testing Dataset	Detected	Percentage
Normal	1000	387	39%
Dos	1200	966	81%
U2R	37	30	81%
R2L	500	152	30%
Prob.	1200	150	13%
All	3937	1586	40%

TABLE 5. K-NEAREST NEIGHBOR (SECOND NORM) CONFUSION MATRIX RESULTS

	kNN Results – Second Norm					
Target	1	2	3	4	5	Total
1	920	23	4	14	39	1000
2	7	1048	0	0	145	1200
3	14	0	16	6	1	37
4	247	3	8	191	51	500
5	0	89	6	3	1102	1200
Total	**1188**	**1163**	**34**	**214**	**1338**	**3937**

TABLE 6. K-NEAREST NEIGHBOR (SECOND NORM) CLASS DETECTION RATES RESULTS

Dataset	Testing Dataset	Detected	Percentage
Normal	1000	920	92%
Dos	1200	1048	87%
U2R	37	16	43%
R2L	500	191	38%
Prob.	1200	1102	92%
All	3937	3277	83%

Using the first norm as the distance measure gave better results. The following tables represent the confusion matrix and class detection rates respectively for the first norm. The kNearest Neighbor demonstrates better results in classifying normal, DoS, R2L and Prob than LVQ, but the classification rate for R2L is still poor.

TABLE 7. K-NEAREST NEIGHBOR (FIRST NORM) CONFUSION MATRIX RESULTS

	kNN Result – First Norm					
Target	1	2	3	4	5	Total
1	929	19	3	13	36	1000
2	7	1064	0	0	129	1200
3	12	0	18	6	1	37
4	238	10	18	190	44	500
5	0	44	4	0	1152	1200
Total	**1186**	**1137**	**43**	**209**	**1362**	**3937**

Combining the results for both classifiers the detection rates for the five classes were improved in the hybrid system, specifically the denial of service attack detection rate rise up to 99%, while the root to local attack still suffers from low detection rate with only 39%.

TABLE 8. K-NEAREST NEIGHBOR (FIRST NORM) CLASS DETECTION RATES RESULTS

Dataset	Testing Dataset	Detected	Percentage
Normal	1000	929	93%
Dos	1200	1064	89%
U2R	37	18	49%
R2L	500	190	38%
Prob.	1200	1152	96%
All	3937	3353	85%

TABLE 9. HYBRID (LVQ_KNN) CLASS DETECTION RATES RESULTS

Dataset	Testing Dataset	Detected	Percentage
Normal	1000	938	94%
Dos	1200	1187	99%
U2R	37	30	81%
R2L	500	194	39%
Prob.	1200	1157	96%
All	3937	3506	89%

VI. RELATED WORKS

The early effort of using GAs for intrusion detection can be dated back to 1995, when Crosbie et al. [7] applied the multiple agent technology and GP to detect network anomalies. Each agent monitors one parameter of the network audit data and GP is used to find the set of agents that collectively determine anomalous network behaviors.

Bridges et al. [8] develop a method that integrates fuzzy data mining techniques and genetic algorithms to detect both network misuses and anomalies. In most of the existing GA based IDSs, the quantitative features of network audit data are either ignored or simply treated, though such features are often involved in intrusion detection.

Chittur et al. [9] 41 unique attributes were compiled from nine weeks of raw TCP dump data from a network. Five million separate connection records were created.

Li et al. [10] Applied a GA to network IDS. It considered both temporal and spatial information of network connections in encoding the network connection information into rules in the IDS. The final goal of the GA was to generate rules that matched only the anomalous connections. In this implementation, the network traffic used for GA is pre-classified data set that differentiates normal network connections from anomalous ones.

Lu et al. [11] used GP for detecting novel attacks on networks. The use of GP to detect unknown attacks was based on the hypothesis that the new rules would have better performance than the initial rules that were based on known attacks.

Xiao et al. [12] presented an approach that used information theory and GA to detect abnormal network behaviors. Based on the mutual information between network features and the types of network intrusions, a small number of network features are closely identified with network attacks. Then a linear structure rule is derived using the selected features and a GA.

VII. REFERENCES

[1] Dieter ollmann. Computer Security, Second Edition. Wiley, New Jersey, 2002.
[2] Edward G.Amoroso, “Intrusion Detection – An Introduction to Internet Surveillance, Correlation, Trace Back, Traps and Response”, Intrusion.net Books, 1999.
[3] Ignacio Porres Ruiz, “an evaluation of current ids”, master of science, university of Linköping, February, 2008.
[4] Dieter Gollmann . Computer Security, Second Edition. Wiley, New Jersey, 2002.
[5] MathWorks Matlab Help, "Learning Vector Quantization Networks",2011.
[6] Information Security Center of eXcellence (ISCX), The NSL-KDD Data Set, 2009. Retrieved October 26, 2011, from http://www.iscx.ca/NSLKDD/
[7] M. Crosbie and E. Spafford, “Applying Genetic Programming to Intrusion Detection”, Proceedings of the AAAI Fall Symposium, 1995
[8] S. M. Bridges and R. B. Vaughn, “Fuzzy Data Mining and Genetic Algorithms Applied to Intrusion Detection”, Proceedings of 12th Annual Canadian Information Technology Security Symposium, pp. 109-122, 2000
[9] Chittur, A. “A Model Generation for an Intrusion Detection System Using Genetic Algorithms”. http:// ww1.cs.columbia.edu/ids/publications/ gaids-thesis01.pdf. Accessed January, 2005
[10] Li, W. “A Genetic Algorithm Approach to Network Intrusion Detection”. http://www.giac.org/practical/GSEC/Wei_Li_GSEC.pdf. Accessed January 2005
[11] W. Lu and I. Traore, “Detecting New Forms of Network Intrusion Using Genetic Programming”, Computational Intelligence, vol. 20, pp. 3, Blackwell Publishing, Malden, pp. 475-494, 2004

