A New Technique by Design an Efficient System for Intrusion Detection

AbdolHamid MomenZadeh [a,*]

[a,*] Corresponding Author: Department of Computer Engineering, Masjed-Soleiman Branch, Islamic Azad University (I.A.U), Masjed-Soleiman, Iran.
E-mail: momenzadeh.hamid@gmail.com Phone: +989166810031

***Abstract*—The basic standard of detect of intrusion is based on the assumption that intrusive activities are noticeably different from normal ones and thus are detectable. In past surveys, the capability of fuzzy systems to solve different kinds of problems confirmed. New attacks are emerging every day, detect of intrusion systems play a basic role in identifying possible attacks to the system, and give proper responses. Evolutionary Fuzzy System with the learning capability of Evolutionary Algorithms hybridizes the approximate reasoning method of fuzzy systems. Propose of this paper is to demonstrate the ability of Evolutionary Fuzzy to deal with detect of intrusion classification problem as a new real-world application area. The Evolutionary Fuzzy System would be capable of extracting accurate fuzzy classification in computer network rules to detect normal and intrusive behaviors from network traffic data and applies them. The experimental results were performed with detect of intrusion benchmark dataset which has information on computer networks, and intrusive behaviors during normal. Results of our model have been compared with several famous detect of intrusion systems.**

***Keywords*—component; Intrusion detection; simulated annealing; search; fuzzy if-then rules.**

I. INTRODUCTION

Attack to network are increasing day by day and whole worlds are affected by that. Security is a critical issue for life of today. An intrusion is defined as any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource [1]. A detect of intrusion system monitors and restricts user range to the computer system by applying certain rules. Those rules are based on expert knowledge extracted from skilled administrators who construct attack scenarios and apply them to find system exploits. The system identifies all intrusions by users and takes or recommends urgent action to stop an attack on the database.

Problems of detect of intrusion has been studied extensively in security of computer [2, 3], and it has received many attention in data mining and machine learning [4, 5].

Misuse detection in contrast approaches aim to encode knowledge about patterns in the data flow that is known to correspond to intrusive procedures by specific signatures. There are several ways for solving detect of intrusion problems. Lee made an detect of intrusion model using association rule and frequent episode techniques on system audit data [6]. Some recent researches have utilized Artificial Immune Systems to detection intrusive behaviors in a computer network [7]. Some applied techniques on detect of intrusion problem are Genetic Fuzzy Rule-Based Systems [8].

Nowadays, different ways have been suggested for automatically generating and adjusting fuzzy if– then rules without the aid of human experts [9].

The concept of Detect of intrusion system was first introduced to complement conventional computer security approaches by Anderson in 1980. Tabu Search (TS) is an evolutionary search method for combinatorial optimization problems proposed by Glover [10]. The goal of this algorithm is to obtain a list of forbidden solutions in the neighborhood of a solution to avoid cycling between solutions while allowing a solution, which may degrade the solution although it may help in escaping from the local optima. In this article, we have used the Tabu search based on misuse detection approach based fuzzy classification system to develop an IDS. The use of TS in IDS is an attempt to effectively explore and exploit the large search space usually associated with detect of intrusion classification problem.

Recently, the adversarial capabilities against detect of intrusion networks (IDNs) are presented in [11]. The authors indexed misclassified instances included in updating could considerably decrease the ability of anomaly-based detection approaches [12].

Tahta et al. employ GP for differentiate malicious peers from benign ones in peer-to-peer networks. They run P2P simulation for each individual to know how derived solutions are effective in preventing malicious peers from participating in the network [13].

A recent GA application shows on principal components instead of working on features directly for both increase the ability of SVM and to use less number of features. Components of principal are computed using Principal Component Analysis, a conventional technique by feature subset selection [14].

The studies on this immature research area have accelerated a survey on adversarial attacks against IDSs was recently proposed in recent years, and [15]. A system which attack evolves signatures, by using GP, is presented. The results show the derived rules are best at detecting both known and unknown attacks. Proposed recently was another signature-based detect of intrusion [16].

Stand-alone, distributed and cooperative architecture in the neighborhood are two different architectures are considered [17]. It is clear the optimal monitoring node selection problem is NP-hard [18]. Another contribution of deployment IDS on MANET using GA is proposed in [19]. Focus on evasion attacks as covered in the foundations section the applications of evolutionary computation mainly [20].

Computational intelligence methods have attracted a considerable interest due to their characteristics suitable for detect of intrusion such as fault tolerance, high computational speed and error resilience adaptation in recent years, in the face of noisy information [21].

Chen et al. discovers the applicability of GA with multi-objective optimization techniques for place sensors IDS by satisfying objectives conflict. Attack various detection rates are traded-off with false alarm costs and rates [22]. The proposed approach has been tested [23] detect of intrusion benchmark data set which has information on computer networks and it is widely used for IDS evaluation.

The rest of the paper is as follows. Sections II and III describe the presented Tabu search based fuzzy classification system for detect of intrusion. Empirical results and the comparison of proposed approach are reported in Section IV Section V is conclusion.

II. Fuzzy rule

In the presented fuzzy classifier system, we use fuzzy if-then rules of the following form:

Rule R_j : If x_1 is A_{j1} and … and x_n is A_{jn} , then Class C_j by $CF = CF_j$. (1)

where R_j is the label of the *jth* fuzzy if–then rule, $A_{j1},...,A_{jn}$ are antecedent fuzzy sets on the unit interval $[0,1]$, C_j is the consequent class, and CF_j is the degree of certainty of the fuzzy if– then rule R_j , We use a typical set of linguistic values in Fig. 1 as antecedent fuzzy sets in computer simulations.

Our system fuzzy searches classifier by a relatively small number of fuzzy if– then rules with ability by high classification.

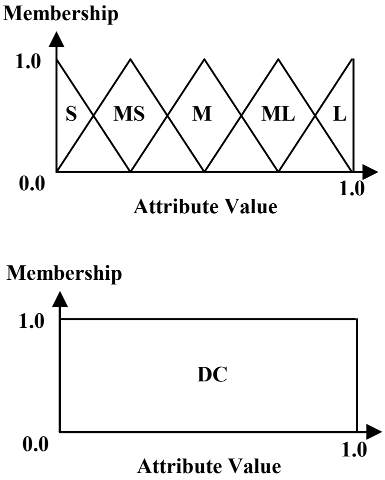

Figure 1. The used fuzzy antecedent sets in this paper. 1. Small, 2. medium small, 3. medium, 4. medium large, 5. large, and 6. don’t care.

III. A new Search based Fuzzy Classification System

The new system composes of *c* classifiers, where *c* is the number of classes. Each classifier includes a subset of rules with the same tags and develops regarding to the total classification rate. Combination of the obtained rule fuzzy sets are used in the structure of the system by final classification. The presented approach concentrates on learning of each class to improve the all accuracy of the goal classifier. Therefore, at any iteration the TSFS is repeated for one of the classes in the classification problem.

Outline of the new Search based Fuzzy System for detect of intrusion problem is as follows:

Step 1: Create an initial set of fuzzy rules and specify the new list (*NL*) size (Initialization).

Step 2: Evaluation current set of rules fuzzy using function evaluation (Evaluation).

Step 3: Produce a new set of fuzzy if– then rules by modifying on of its rules from set current of rules (Switch).

Step 4: Accept the new rule set if it is the modified rule is not in *TL* or better than current solution (Acceptance).

Step 5: The algorithm Terminate if the stopping condition are satisfied, otherwise return to Step 2 (Termination).

Each step of this system is described below:

A. Initialization

The method of coding fuzzy if-then rules is as follows: Each fuzzy rule if- then is coded string. The following symbols are used for denoting the five linguistic values (Fig. 1). 1:small, 2:medium small, 3:medium, 4:medium large, 5:large, and 6:DC. For instant, the upper fuzzy if-then rule is coded as “36”:

R_j: If x_1 is medium and x_2 is DC then Class C_j with CF=CF_j.

N_{init} denote the digit of fuzzy if-then rules in the initial set. For design a private set, one method is to product N_{init} fuzzy if-then rules of each rule using five linguistic values and DC by randomly specifying the antecedent fuzzy sets. Here, we increase the probability of DC, when specifying the antecedent fuzzy sets to create general rules to cover larger decision area of the state space. After production N_{init} fuzzy if- then rules as private set of rules S_{init}, the consequent class and the certainty degree of each are specified from training patterns by method described in section II. In simulation of computer N_{init} varies from 30 to 50.

$$fitness(R_j) = NCP(R_j) \tag{2}$$

where $NCP(R_j)$ is by the fuzzy if-then rule R_j number of correctly classified training patterns.

A very important parameter here is the choice of Tabu list size. This is a problem-dependent parameter, since the choice of a large size would be inefficient in terms of memory capacity and the time required for scanning the list. On the other hand, choosing the list size to be small would result in a cycling problem; that is, revisiting the same state again. In computer simulation *TL* size is set equal to 100.

B. Evaluation

The set of fuzzy if-then rules must have high accuracy. Thus we use the following function to evaluate each rule set:

$$NNCP(S) = m - \sum_{j=1}^{N} NCP(R_j) \tag{3}$$

where N is the number of rules in the rule set, m is in the training set the number of all patterns, and $NNCP(S)$ is by S the number of non-correctly classified training patterns.

C. Switch

A rule is selected from a class of the current rule set $S_{current}$ randomly, and then one of its linguistic antecedent parts is changed to other linguistic variables. The probability to change by distance one is greater than other distances. For example, if the selected antecedent part is "*Medium*" then the probability of conversion to the "*Medim Small*" and "*Medium Large*" is greater than the probability of conversion to "*Small*" or "*Large*". After performing the *Switch* operation, consequent class of the changed fuzzy if-then rule is determined. If consequent class of the new rule is the same as its parent class then this rule will be replaced by its parent to create new set of rules S_{new}. Otherwise, this process is repeated.

D. Acceptance

If the new rule set S_{new} is better than the best rule set found so far S_{best}, it is accepted and saved as the best found by setting S_{best}=S_{new}. If the new rule set is better than the rule set $S_{current}$, it is saved and accepted by the current rule set by setting $S_{current}$=S_{new} . If the new rule set is not better than the current rule set, and the new generated rule R_{new} is not in a direction within the Tabu list *TL*, it is accepted as the current rule set and the search continues from there. If the R_{new} is in Tabu list, the current rule set remains unchanged and a new rule set is generated. After accepting a new rule set, *TL* is updated and contains the new generated rule R_{new} to forbid returning to this rule again.

E. Termination

There are several conditions that we can terminate the proposed algorithm. One alternative is to finish the algorithm after a fixed number of iterations. Another alternative is to terminate the algorithm when no decrement in the *NNCP* value obtains after some number of iterations. In TSFS, stopping condition is a predefined number of iterations. In computer simulation, the number of iterations is set equal to 10000.

IV. Experimental results

The Fuzzy IDS was simulated and tested via two different approaches, one was to cluster and recognize all types of packets, and the other was to cluster and recognize four pairs having pattern "normal packet vs. some attack type". This was done to determine the success in classification of each type of attack using proposed model. Experiments were in the framework of the 1998 Intrusion Detection Evaluation Program and carried out on a subset of the database created by DARPA. We distributed as part of the UCI KDD Archive used the subset that was pre-processed by the Columbia University and [28]. The available database is made up of malicious traffic a large and number of network connections related to normal. Every connection is represented with a 41 -dimensional feature vector. Connections are also labeled as belonging to one out of five classes. One of these classes is the rest indicates four different intrusion classes and the normal class.

These intrusion classes are classification of 23 different types of spells in a computer network.

We consigned the train and test data sets, that each numerical value in the data set is normalized between 0.0 and 1.0 according to the following equation:

42 numeric features are constructed and consigned to the interval [0, 1]. This section consists of two subsections. First we present some experiments of applying proposed approach to the intrusion detection classification problem. at next subsection we compare the performance of our algorithm to some of other intrusion detection algorithms.

The average accuracy rate obtained from proposed model varies from 93% to 98% with the number of rules ranging from 50 to 100. The hybrid model which extracts 67 fuzzy if-then rules with the average rule length of 5.54 from training data and obtains 97.81% accuracy rate is applied for test validation. Table IV is the confusion matrix of hybrid model. The top -left entry of Table III of the actual Normal test set were detected to be Normal shows that 59976 instances.

Classification performance of TSFS is measured and compared with that of different classification algorithms including pruning C4.5, Naïve Bayes (NB), *k*-Nearest Neighbor (*k*-NN), Support Vector Machine (SVM), and Multi-Objective Genetic Fuzzy Intrusion Detection System (MOGFIDS) [6]. In *k*-NN parameter by using the well-known fast sequential minimal optimization method with a polynomial kernel *k* is set to 5, and the SVM is trained. The results compare with the winner of KDD-Cup99 contest [29].

TABLE I
CLASSES IN THE 10% OF THE KDD-CUP 99 DATA SET

CLASS	SUB-CLASSES	SAMPLES
NORMAL		97278 (19.6911%)
PRB	ipsweep, nmap, portsweep, satan	4107 (0.8313%)
DOS	back, land, neptune, pod, smurf, teardrop	391458 (79.2391%)
U2R	buffer_overflow, loadmodule, multihop, perl, rootkit	52 (0.0105%)
R2L	ftp_write, guess_passwd, imap, phf, spy, warezclient, warezmaster	1126 (0.2279%)

TABLE II
DISTRIBUTION OF DIFFERENT CLASSES IN THE TRAIN AND TEST SETS

Class	Train	Test
NORMAL	10000	60593
PRB	4107	4166
DOS	5467	229853
U2R	52	228
R2L	1126	16189

$$X_{normalized} = \frac{X - X_{\min}}{X_{\max} - X_{\min}} \tag{4}$$

This section consists of two subsections. First we present some experiments of applying proposed approach to detect of intrusion classification problem. The average accuracy rate obtained from the proposed approach varies from 93.50% to 98.75% with the number of rules ranging from 20 to 50. The fuzzy rule base which extracts 34 fuzzy if-then rules from training data and obtains 98.41% accuracy rate is applied for test validation.

TABLE III
CONFUSION MATRIX FOR THE TABU SEARCH BASED IDS

	Detected Class					
Real Class	NOR	PRB	DOS	U2R	R2L	%
NOR	60027	356	110	24	205	99.06
PRB	434	3508	215	34	1	84.20
DOS	5721	117	223981	15	3	97.44
U2R	139	51	8	34	9	15.93
R2L	12350	24	2768	7	2053	13.58
%	79.30	90.26	98.74	32.87	91.80	**94.56**

Table III is the confusion matrix of proposed approach. The last column indicates that 13.58% of the actual R2L samples were detected correctly. The bottom-right entry of the table III shows that 94.56% of the all patterns in the test data set are correctly classified. The results are also compared with the winner of KDDCup99 contest [24].

Table III shows the results of Recall, Precision, and F-measure of different classifier for each class. For the class of PRB attacks proposed approach outperforms other classifiers in terms of both Precision and F-measure. Regarding the rare class of R2L attacks, proposed approach obtains competitive results in comparison with other algorithms.

According to the table III the proposed approach obtains the highest accuracy rate among the other classifiers. This is because of the proposed approach explores the state space of the high-dimensional classification problem better than other algorithms and with adequate initialization, modification, and cost function tries to reach the global optimum and escape from local optima.

V. CONCLUSION

In this paper, we have introduced the TS-based fuzzy classification system for detecting intrusive behaviors in a computer network. Simulations of computer on data sets demonstrate presented method achieves robust performance for classifying both intrusion attacks and network normal traffic. As discussed in the article detect of intrusion is a real-world application area that is not previously investigated by Tabu search based system fuzzy. Also, it is Very important to investigate other TS System Fuzzy architectures for this complex classification problem. Also, the use of multi-objective Tabu search based fuzzy systems to extract a comprehensible fuzzy classifier for detect of intrusion is another considerable investigation topic which is left for our future work.

REFERENCES

[1] R. Heady, G. Luger, A. Maccabe, and M. Servilla, “The architecture of network level intrusion detection system,” Technical Report, Department of Computer Science, University of New Mexico, 1990.

[2] N. Ye, S. Vilbert, and Q. Chen, “Computer Intrusion Detection Through EWMA for Autocorrelated and Uncorrelated Data,” IEEE Transactions on Reliability, vol. 52, no. 1, Mar. 2003, pp. 75-82.

[3] N. Ye, Q. Chen, and C.M. Borror, “EWMA Forecast of Normal System Activity for Computer Intrusion Detection,” IEEE Transactions on Reliability, vol. 53, no. 4, Dec. 2004, pp. 557-566.

[4] S.B. Cho, “Incorporating soft computing techniques into a probabilistic intrusion detection system,” IEEE Transactions on Systems, Man and Cybernetics, Part C, Volume 32, Issue 2, May 2002, pp.154-160.

[5] S. Cho, S. Cha, “SAD: web session anomaly detection based on parameter estimation,” Computers & Security, Vol.23, No.4, Jun. 2004, pp.265-351.

[6] W. Lee, J.S. Salvatore, and K.W. Mok, “Mining audit data to build intrusion detection models,” In: Proceedings of ACM SIGKDD international conference on knowledge discovery and data mining, 1998, pp. 66-72.

[7] D. Dasgupta, and F. González, “An Immunity-Based Technique to Characterize Intrusions in Computer Networks,” IEEE Transactions on Evolutionary Computation, vol. 6, no. 3, Jun. 2002, pp. 1081-1088.

[8] C.H. Tsang, S. Kwong, H. Wang, “Anomaly Intrusion Detection Using Multi-Objective Genetic Fuzzy System and Agent-Based Evolutionary Computation Framework,” ICDM05, 2005, pp. 789-792.

[9] H. Ishibuchi, K. Nozaki, and H. Tanaka, “Distributed representation of fuzzy rules and its application to pattern classification,” Fuzzy Sets and Systems, 52(1), pp. 21-32, 1992.

[10] F. Glover, M. Laguna, “Tabu Search,” Kluwer Academic Publishers, Dordrecht, 1997.

[11] Pastrana, S. (2014). Attacks against intrusion detection networks: evasion, reverse engineering and optimal countermeasures. PhD Thesis, Computer Science and Engineering Department, Universidad Carlos III De Madrid,

[12] Sen, S. (2014). Using instance weighted naive Bayes for adapting concept drift in masquerade detection. International Journal of Information Security.

[13] Tahta, U.E., Can, A.B., Sen, S. (2014). Evolving a trust model for peer-to-peer networks using genetic programming, In Proc. of EvoStar, LNCS Series, Springer. (to appear)

[14] Ahmad, I., Hussain, M., Alhgamdi, A., Alelaiwi, A. (2014). Enhancing SVM performance in intrusion detection using optimal feature subset selection based on genetic principal components. Neural Computing and Applications, Vol. 24, Issue 7-8, pp. 1671-1682.

[15] Corona, I., Giacinto, G., Roli, F. (2013). Adversarial attacks against intrusion detection systems: Taxonomy, solutions and open issues. Information Sciences, Vol. 239, pp. 201-225.

[16] Gomez, J., Gil, C., Banos, R., Marquez, A.L., Montoya, F.G., Montoya, M.G. (2013). A Pareto-based multi-objective evolutionary algorithms for automatic rule generation in network intrusion detection systems, Soft Computing, Vol. 17, pp. 255-263.

[17] Hassanzadeh, A., Stoleru, R. (2013). On the optimality of cooperative intrusion detection for resource constrained wireless networks, Computers & Security, Vol. 34, pp. 16-35.

[18] Hassanzadeh, A., Stoleru, R., Shihada, B. (2011). Energy efficient monitoring for intrusion detection in battery-powered wireless mesh networks. In Proc. of ADHOC-NOW, LNCS 6811, Springer, pp. 44-57.

[19] Hassanzadeh, A., Stoleru, R. (2011b). Towards optimal monitoring in cooperative ids for resource constrained wireless networks. In Proc. of the 20th International Conference on Computer Communications and Networks.

[20] Kayacik, H. G., Zincir-Heywood, A. N., Heywood, M. I. (2011). Evolutionary computation as an artificial attacker: generating evasion attacks for detector vulnerability testing. Evolutionary Intelligence, Vol. 4, issue 4, pp. 243-266.

[21] Wu, S. X., Banzhaf, W. (2010). The use of computational intelligence in intrusion detection systems: a review, Applied Soft Computing, Vol. 10(1), pp. 1-35.

[22] Chen, H., Clark, J.A., Shaikh, S.A., Chivers, H., Nobles, P. (2010). Optimising IDS Sensor Placement, In Proc. of International Conference on Availability, Reliability, and Security.

[23] KDDCup data set: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

[24] C. Elkan, “Results of the KDD99 classifier learning,” ACM SIGKDD Explorations 1, 2000, pp. 63–64.