GENETIC ALGORITHM BASED GENERATION OF CLASSIFICATION RULES FOR NETWORK INTRUSION DETECTION

Dr.N.Arumugam

(Lecturer (SG), Dept of ECE, Nachimuthu Polytechnic College, Pollachi, Tamilnadu, South India)

Abstract— *In recent days, due to the rapid expansion of Internet, computer systems are facing vast number of security threats. In spite of numerous detection and defense methodologies proposed for information assurance, it is still very difficult to protect computer systems. As a result, unwanted intrusions take place when the actual software systems are running. Recently soft computing based intrusion Detection systems (IDs) have been subjected to extensive researches because they can detect both misuse and anomaly detection. In this paper the method of learning the Intrusion Detection, rules based on genetic algorithms was presented. The genetic algorithm is employed to derive a set of classification rules from network audit data, and the support-confidence framework is utilized as fitness function to judge the quality of each rule. The generated rules are then used to detect or classify network intrusions in a real-time environment. The proposed representation of rules and the effective fitness function is easier to implement while providing the flexibility to either generally detect network intrusions or precisely classify the types of attacks. Experiments results shows, the characters of an attack such as SMURF and SNMP get attack were summarized through the Modified and corrected KDD 99 data set and the effectiveness and robustness of the approach are proved.*

Keywords— ***Intrusion Detection; Genetic Algorithm; KDD Cup Data Set***

1. INTRODUCTION

In the recent years the contribution of Internet to the society are expanding at an amazing rate. At the same time computer systems are exposed to increasing security threats that originate externally or internally. Computer networks are usually protected by a number of access restrictions policies like anti-virus software, firewall, secure network protocols, etc. Since it has been proven that a potential attacker can always find a way to gain access into a network, there is a need for additional support that would detect this type of security violation. These systems are known as intrusion detection systems (IDS) and are placed inside the protected network, looking for potential threats in network traffic and/or audit data recorded by hosts [1]. There are two general categories of intrusion detection systems; they are misuse detection and anomaly detection. Misuse detection systems detect intruders with known patterns, and anomaly detection systems identify deviations from normal network behaviors and alert for potential unknown attacks. Some IDS integrate both misuse and anomaly detection and form hybrid detection systems [2]

A number of soft computing based approaches have been proposed for detecting network intrusions. Soft computing refers to a group of techniques that exploit the tolerance for imprecision, uncertainty, partial truth, and approximation to achieve robustness and low solution cost. The principle constituents of soft computing are Fuzzy Logic (FL), Artificial Neural Networks (ANNs), Probabilistic Reasoning (PR), and Genetic Algorithms (GA) [3].For intrusion detection, soft computing techniques are often used in conjunction with rule- based expert systems acquiring expert knowledge [4, 5], where the knowledge is represented as a set of if-then rules. Despite different soft computing based approaches having been proposed, the possibilities of using the techniques for intrusion detection are still under-utilization.

2. KDD-99 DATASET

One of the most important datasets for testing IDS is the KDD 99 intrusion detection datasets. KDD-99 provides designers of IDS with a benchmark on which to evaluate different methodologies. This dataset is created by MIT Lincoln Lab's DARPA in the framework of the 1998 Intrusion Detection Evaluation Program [6]. In this paper, we used the subset that was preprocessed by the Columbia University and distributed as part of the UCI KDD Archive [7, 8]. The dataset can be classified into five main categories which are Normal, Denial of Service (DoS), Remote to Local (R2L), User to Root (U2R) and Probing.

- Denial of Service (DoS): Attacker tries to prevent legitimate users from using a service.
- Remote to Local (R2L): Attacker does not have an account on the victim machine, hence tries to gain access.
- User to Root (U2R): Attacker has local access to the victim machine and tries to gain super user privileges.
- Probe: Attacker tries to gain information about the target host.

The inherent drawbacks in the KDD cup 99 dataset has been revealed by various statistical analyses [10].

3. GENETIC ALGORITHM

Genetic algorithms employ metaphor from biology and genetics to iteratively evolve a population of initial individuals to a population of high quality individuals, where each individual represents a solution of the problem to be solved and is compose of a fixed number of genes. The number of possible values of each gene is called the cardinality of the gene [9].

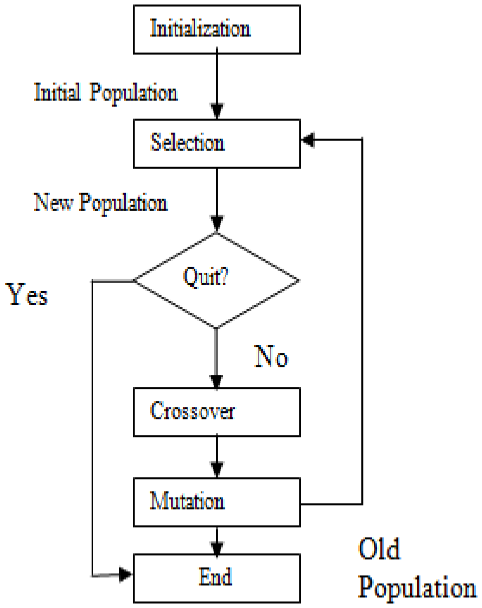


Figure.1 Procedure of GA

Fig 2 illustrates the operation of a genetic algorithm.

The operation starts from an initial population of randomly generated individuals. Then the population is evolved for a number of generations and the qualities of the individuals are gradually improved. During each generation, three basic genetic operators are sequentially applied to each individual with certain probabilities like selection, cross over, and mutation. First, the numbers of best-fit individuals are selected based on a user-defined fitness function. The remaining individuals are selected and paired with each other. Each individual pair produces one offspring by partially exchanging their genes around one or more randomly selected crossing points. At the end, a certain number of individuals are selected and the mutation operations are applied. When a GA is used for problem-solving, three factors will have impact on the effectiveness of the algorithm,

they are: 1) the selection of fitness function
2) the representation of individuals
3) the values of the GA parameters.

4. CLASSIFICATION RULES FOR INTRUSION DETECTION

An Intrusion Detection System is a computer program that attempts to perform intrusion detection by either misuse/anomaly detection or a combination of techniques. Once an intrusion has been detected, the detection system issues alerts information to the administrators. The next step is undertaken either by the administrators or the IDS itself. Intrusion detection may sometimes produce false alarms.

The genetic algorithm is employed to derive a set of classification rules from network audit data, and the support-confidence framework is utilized as fitness function to judge the quality of each rule. The generated rules are then used to detect or classify network intrusions in a real-time environment. Applying genetic algorithm to intrusion detection seems to be a promising area. It can be used to evolve simple rules for network traffic. These rules are used to differentiate normal network connections from anomalous connections. These anomalous connections refer to events with probability of intrusions. The rules are usually in the following form:

if {condition} then { proceed }

the condition generally refers to a match between current network connection and the rules in IDS, such as source and destination IP addresses and port numbers, duration of the connection, protocol used, etc., indicating the probability of an intrusion. The proceed field usually refers to an action defined by the security policies within an organization, such as reporting an alert to the system administrator, stopping the connection, logging a message into system audit files, or all of the above. The condition using this format refers to the attributes in the rule set that forms a network connection in the dataset. Based on the condition the result may be ‘true’ or ‘false’. The attack name will be specified only if the condition is true.

S. No	Protocol	Service	Flag	Src bytes	Dest bytes	Srv_count	Serror_rate
1	udp	private	SF	105	147	1	1
2	udp	private	SF	105	147	2	2
3	udp	private	SF	105	147	1	1
4	udp	private	SF	105	147	2	2

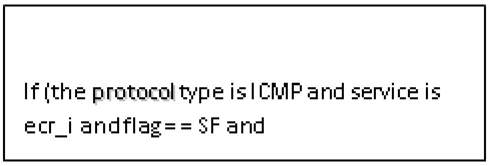
5. CLASSIFICATION RULES FOR SMURF AND SNMP GET ATTACK

The data set KDD 99 can be classified in to five main categories which are Normal, DoS (Denial of Service attack), R2L (Remote to Local), U2R (User to Root) and Probing. Out of five classes of attacks this paper forms the rule for SMURF and SNMP get attack. The rules are generated using the C4.5 algorithm on the part of corrected KDD training data set. Smurf attack is one of the DoS attacks, in which the attacker makes some computing or memory resource too busy or too full to handle legitimate requests or denies legitimate users access to a machine. In the smurf attack, attacker use “ICMP” echo request packets directed to IP broadcast addresses from remote locations to create a DOS attack.

TABLE I SMURF ATTACK SAMPLE DATA SET WITH REDUCED FEATURES

S. No	**Protocol**	**Service**	**Flag**	**Src bytes**	**Dest bytes**	**Count**	**Srv_count**
1	icmp	ecr_i	SF	1032	0	508	508
2	icmp	ecr_i	SF	1032	0	509	509
3	icmp	ecr_i	SF	1032	0	510	510
4	icmp	ecr_i	SF	1032	0	511	511

These are three parties in these attacks; the attacker sends ICMP echo request packets of the broadcast address of many subnets with the source address spoofed to be that of the intended victim. Any machines that are listening on these subnets, will respond by sending ICMP “echo reply” packets to the victim.



If (the protocol type is ICMP and service is ecr_i andflag = = SF and

Like smurf attack the classification rules are developed for the SNMP get attack. SNMP get attack is very difficult to detect attack. Normally, monitors the SNMP community using password guessed by SNMP guess.

TABLE II SNMP GET ATTACK SAMPLE DATA SET WITH REDUCED FEATURES

If (the protocol type is UDP and service is private and flag = = SF and srv_count value<=2 and >=1) then SNMP get attack

The block diagram of the GA based intrusion detection method is shown in the figure 2. The generated rule set for intrusions detection will be framed to the GA format. The first part of the GA will act as a search algorithm. It will match the rules with any anomalous connections that occur the network to detect an intrusion.

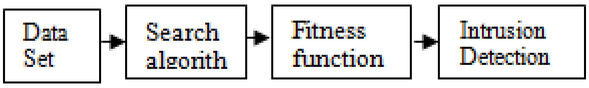


Fig 2: GA based Intrusion Detection Method

Each rule will carry values for the intrusions that they have detected and a value for the intrusion. Each rule will carry values for a false alarm that the rule produces. The second part of the GA is the fitness function. The fitness function F determines whether a rule is good or bad. F is calculated for each rule using the support confidence framework.

Support = │A and B│ / N
Confidence = │A and B│ /│ A│
Fitness = X1 * support + X2 * confidence

Where,

N is the total number of records

│A│ stands for the number of network connections matching the condition A

│A and B│ is the number of records that matches the rule.

X1 and X2 are the thresholds to balance the two terms.

6. PERFORMANCE EVALUATIONS OF PROPOSED RULES ON THE KDD DATA SET

Since the KDD99 data set has 41 variables, the selection of most important features are identified to form classification rules and observe their performance with respect to the detection rate, false alarm and missed alarm rates. If the rules are well formed, then the detection rate is expected to be high while concurrently achieving low false alarm and missed alarm rates. For each chromosome in the population, the number of network connections and the number of connection that matches the condition is initialized to zero. For each record in the training set, if the record matches the chromosome update the network connection by 1 and if the record only matches the condition part, then update that value A by 1. Then calculate the fitness of each rule and select the best fit rules into new population. Then apply the crossover and mutation operators to each rule in the new population. Finally, decide whether to terminate the training process or to enter the next generation to continue.

TABLE III

Record Type	Training Set	Testing Set
Normal	96.2%	93.8%
Smurf attack	90.6%	67.3%
SNMP get attack	92.7%	76.6%

7. RESULTS (DETECTION RATES)

The experimental result, Table III shows that the proposed method yielded good detection rates when using the generated rules to classify the training data itself.

8. CONCLUSIONS

This paper utilized a technique for creating rules for SNMP get attack and Smurf (DOS) attacks. Probability of detection and false alarm rates are computed on the KDD data set. The overall performance is reasonably good with an average of 60% detection rate and achieved with only 0.15% false rate. Most of the network security systems require capability to handle over one million of concurrent sessions. However, some limitations of the method are also observed. First the generated rules were biased to the training data set. This paper is mainly an attempt towards overcoming various short comings in the context of network intrusion detection. The data set used serves as the basis for the intrusion detection process to detect intrusions. It is represented in the form of a table or a record set. In future work, a generalized form in which the data set should be represented from various representations of data sets, still needs to be researched to be compatible with any kind of IDS. Second while the support – confidence framework is simple to implement and provides improved accuracy, it requires the whole training data to be loaded into memory before any computation. For large data sets, it is neither efficient nor feasible. Future work will attempt to use artificial intelligent technologies to automate the generation of rules.

REFERENCES

[1] A. Adetoye, A. Choi, M. Md. Arshad, and O. Soretire , “Network Intrusion Detection & Response System”, Group Report, September 2003, http://www.cs.ucl.ac.uk/teaching/dcnds/group-reports /2003/2003-hailes-b.pdf (accessed in January 2005).

[2] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, “Network intrusion detection”, IEEE Network, 8(3): 26-41, May/June 1994.

[3] M. Moradi and M. Zulkernine, "A Neural Network Based System for Intrusion Detection and Classification of Attacks", Proceedings of the 2004 IEEE International Conference on Advances in Intelligent Systems - Theory and Applications,

Luxembourg, November 2004.

[4] J. Gomez and D. Dasgupta, “Evolving Fuzzy Classifiers for Intrusion Detection”, Proceedings of the IEEE, 2002.

[5] G. Helmer, J. Wong, V. Honavar and L. Miller, “Automated discovery of concise predictive rules for intrusion detection”, The Journal of Systems and Software, issue 60, pp. 165-175, 2002.

[6] Intrusion Detection Evaluation Program (http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/index.html).

[7] Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, “The 1999 DARPA off-line intrusion detection evaluation” The International Journal of Computer and Telecommunications Networking, Volume 34, Issue 4 (October 2000) Page(s): 579 – 595.

[8] UCIKDD Archive, http://kdd.ics.uci.edu/ databases /kddcup99/ kddcup99.html) September 2009.

[9] Pohlheim H, “Genetic and Evolutionary Algorithms: Principles Methods and Algorithms”, http://www.gearbx.com/docu/index.html, January 2005.

[10] Sapna S. Kaushik, Dr. Prof. P.R.Deshmukh,” Detection of Attacks in an Intrusion Detection System”, International Journal of Computer Science and Information Technologies, Vol. 2 (3), 2011, 982-986.