Development of an evolutionary framework for autonomous rule creation for intrusion detection

Sunitha Guruprasad
Department of Computer Science & Engg
St Joseph Engineering College
Mangaluru, India
sunitagp@rediffmail.com

Rio D'Souza
Department of Computer Science & Engg
St Joseph Engineering College
Mangaluru, India
rio@ieee.org

***Abstract*—Network intrusion detection system (IDS) plays a major role in any security based architecture. Various IDS have been developed to detect the intrusions that occur in the real world. The most commonly used network security tool used is Snort IDS. Snort is a rule-based system that generates alerts for the matching network patterns. Most of the rules stored in the Snort database fail to generate alerts for real network traffic. It is necessary to create rules that detect the attacks efficiently. In this paper we have made an attempt to autonomously generate rules using the evolutionary approach. The rules produced were tested for Darpa 1999, ISCX 2012 and ICMP network packets and were able to detect attacks with a high detection rate.**

***Index Terms*— Intrusion detection, evolutionary, Genetic, algorithm, ISCX, Darpa, Snort, Bro, multiobjective.**

I. Introduction

Due to the increased use of internet in the recent decades, the data security of any organization has been a major issue. In the real world, we can find many new types of data security attacks emerging and it is very difficult to tackle these attacks using the traditional approach. Many security tools like firewall, antivirus, etc., are used for security purposes, but these tools only provide limited security and are not able to secure the systems fully. Intrusion detection systems (IDS) offer many techniques for detecting normal and abnormal behavior in an effective way. IDS are software that is used to detect and monitor the network traffic and generate alerts when it encounters an abnormal pattern.

Many IDS like Snort, Bro, etc., have been developed to detect intrusions and protect the systems from unauthorized access. Snort intrusion detection system, developed by Martin Roesch [1], is an open source, lightweight IDS. It is a signature based system which matches the network packets with the rules that are already stored in its database and generates alerts if a match occurs. Developing and maintaining the ruleset is a very cumbersome task in rule-based systems. Most of the rules have to be generated manually by analyzing the network pattern. This needs expert knowledge. The other method is to autonomously create the rules by using some computation based techniques. Most of the researchers have tried various methods in order to generate efficient rules.

In this paper we have made an attempt to autonomously generate the rules using an evolutionary approach. Our work is similar to the approach used by Vollmer, et al. [2], but in a different context. Genetic algorithm has been used mainly because it provides multiple offspring. They are intrinsically parallel and hence can explore the solution in multiple directions. If one path turns out to be a deadend, they can continue in a different direction giving them a greater chance of finding an optimal solution. We have used the concept of clustering by grouping together rules belonging to the same attack. The packets used as inputs are taken from the captured real-time data and ISCX datasets and have also been tested using Darpa 1999 dataset. The resulting rule set is able to identify the attacks at a satisfactory level.

The rest of the paper is organized as follows. Section II briefly presents an overview of the related work. Section III provides description of the attack type used, evolutionary algorithm and the Snort-IDS-rules. Section IV presents the datasets used and discusses the results obtained. Finally, section V presents the conclusion and the future work that can be carried out.

II. Related work

The main advantage of using a Snort IDS to detect intrusions is that it provides high detection accuracy. But the drawback is that it generates lots of irrelevant alerts. Also, most of the rules that are stored in Snort database do not detect attacks for real-time data.

Different approaches have been used by the researchers to improve the rules part in Snort. Gómez, Julio, et al. [3] attempted to extend the functionalities of Snort by designing an optimization strategy to allow the automatic generation of rules to detect the suspicious traffic. A Pareto-based multiobjective evolutionary algorithm (MOEA) was used within the detection engine of Snort to detect new attacks. The work was extended by Gómez, Julio, et al. [4] where, a Pareto-based MOEA was designed to optimize the rule generation within the detection engine of Snort. A wide range of solutions were obtained using two optimization modes: Pareto optimization and single aggregate objective function. The result showed that the solution quality increased when the population size was increased and also when the backtracking rate value increased. The main aim of both the works was to minimize the false positives and false negatives to optimize the rule generation.

Wuu, et al. [5] enhanced the functionalities of Snort IDS to detect the sequential intrusion behavior and to automatically

978-1-4673-8286-1/16 $31.00 © 2016 IEEE
DOI 10.1109/IACC.2016.105

generate patterns of misuse from attack data. Data mining techniques were used to extract single intrusion patterns and sequential intrusion patterns from the attack packets. These patterns were then converted to Snort detection rules for on-line intrusion detection.

Khamphakdee, et al. [6] have tried to improve the Snort rules for network probe attack detection. Based on the nature of the attack, they have classified the probe attacks into different groups for MIT-Darpa 1999 datasets. They have also compared the detected attacks with the Detection Scoring truth.

Wang, et al. [7] presented a tool called NetSpy that generates the NIDS signatures automatically for specific instances of spyware. It compares the network traffic from a clean system and the infected system for spyware activities. But, the main limitation is that it only works for browser plugins.

A multi-modal genetic algorithm has been devised by Vollmer, et al. [2] for autonomous rule creation. They have used an evolutionary based approach and tested the Snort rules on ICMP network packets. Three packet creation tools were used to hand craft ten test packets. These packets were used to generate alerts for ten different snort rules. The rules generated produced very low false positive rate.

Among the approaches used by the researchers, we found that the evolutionary based approach produces better results compared to all other approaches. In our work, we have used a slightly different approach and have developed a framework for autonomously creating the rules.

III. Detection Model

In this section we give a framework of our approach which consists of the attack type used, representation of genetic algorithm, and implementation of fitness function and rule creation.

A. Attack type

The system has been tested for finger abuses and Denial-of-Service (DoS) attacks like Internet Control Message Protocol (ICMP) attacks.

The finger is a program in unix that is used to find information like, login name and other details about users. It is the most common service used to crack passwords and to compromise the user's account.

DoS is an attempt to make a network resource or a machine unavailable to intended users by overwhelming it with useless traffic. Some DoS attacks are also used to exploit the limitations in protocols like TCP/IP. ICMP Ping flood attack is an attempt to flood the victim's machine with large number of network packets. This is usually done by pinging the victim's machine from a remote machine and flooding it.

B. Evolutionary approach

We have used an evolutionary approach in our experiments which goes through the process of selection of fit individuals, crossover of parents and mutation operations. In every generation, the selection operator is applied which selects the best individual based on the fitness function. Crossover operator constructs new individual by combining the characteristics of two best individuals. Mutation operator modifies the original individuals by making small variations in the individuals. Fig. 1 shows the evolutionary algorithm for the generation of IDS.

One of the main tasks in any genetic algorithm is to create an initial population and to decide about the number of individuals in the population. A single rule cannot classify all the attack types accurately. A set of rules are required to efficiently detect the attack. Hence, the rules belonging to the same attack type are clustered together and stored separately.

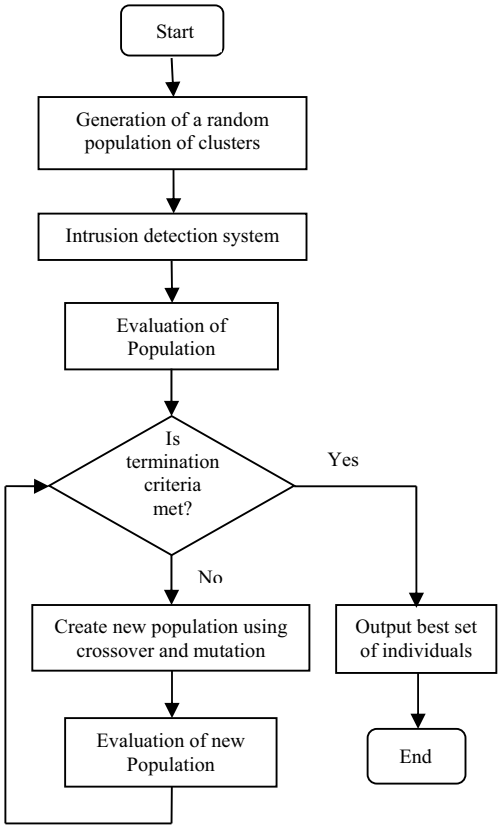

Fig. 1. Evolutionary Algorithm for development of IDS

Each rule in a cluster is taken as an individual or a chromosome and a set of individuals form a population. Each individual is represented as a set of fields or genes ($g_1,g_2,\ldots,g_n$) of variable length and type. These individuals are tested separately using Snort IDS and then evaluated using a fitness function. The detection rate of each individual is taken as a fitness value.

Detection rate is calculated as the ratio of the number of intrusions detected by the system to the total number of intrusions present.

Then the average fitness of all the individuals is calculated. Let f_i be the fitness of the individuals. Given a fitness f_i and the number of individual's n, the average fitness f_A is calculated using the formula,

$$f_A = (f_1+f_2+...+f_n)/n \quad (1)$$

In order to select the best rule, the rules are classified as good and bad by comparing them with the average fitness f_A. Given a threshold t, f_i is selected as the best fitness rule iff it satisfies t. This becomes the first population. For example, if the average fitness is 51.3%, then t value will be 25.65 (50% of average fitness). The average fitness is recalculated for the best set of rules.

In the next level, the individuals for the next population are selected. This is done using crossover and mutation operators. For the crossover operation, the parameters of the rules are divided into two parts from 40% to 60% each. The first two parents are selected from the first two individuals and two offspring are generated. This process is repeated for the remaining individuals. Finally, the entire population is replaced by the new generation. We have used only the option field to perform the crossover. In order to select the rule header for the new offspring, the header of the first parent is taken as the header for the first offspring and the header of the second parent as the header for the next offspring.

Mutation is performed by making small variations in the individuals.

The new sets of individuals are again tested using Snort-IDS and evaluated using a fitness function. This process is repeated until a stopping criterion is met. For our experiments we have taken a fixed number of 50 iterations as a stopping criterion.

C. Snort-IDS-rules

In our experiments, we have used the rules that are already stored in the rules set of Snort-IDS. Minor changes were made to these rules so that they fit according to our requirements. Few rules are also generated manually by analyzing the network patterns and the datasets.

TABLE I. PARAMETERS OF FINGER ABUSE RULES

Parameters	Values
content	"0"; "."; "version"; "search"; "|00|"; "@@"; "|0A| "; "root"; "|7C|"; "/"; etc.,
pcre	"/^\x2f/smi"
metadata	service finger
flow	to_server,established
reference	arachnids,130; cve,1999-0198; nessus,10072; bugtraq,974; etc.,
classtype	attempted-recon; attempted-user; attempted-dos
fragoffset	0
fragbits	D
flags	A, AP
Attack type	FINGER 0 query, bomb attempt, redirection attempt, root query, etc.,
Priority, sid, rev	Used as required by the rules

In order to generate the rules, the real-time network packets were captured and stored in files. After carefully examining the network patterns, the rules were written manually for the most common pattern and tested.

Since these rules were not able to detect all the attacks, a parameter-selection method is used to select the best parameter for the rule. We have used a method similar to apriori algorithm [8] to perform this operation. In this method, an analysis of the fields in the rules is made and the frequency count f_c of all the fields is noted down. The fields are included in the rule randomly based on f_c. Table I shows a few parameters that were used in the rules. Each parameter in the table is taken as a gene and a set of genes forms a chromosome. Table II shows the options of a few rules that were used to detect the finger abuses and ICMP attack.

TABLE II. SAMPLE RULES

No	Packet Details
1	(msg:"FINGER bomb attempt"; flow:to_server,established; content:"@@"; metadata:service finger; reference:arachnids,381; reference:cve,1999-0106; classtype:attempted-dos; sid:1000047; rev:9;)
2	(msg:"FINGER redirection attempt"; flow:to_server,established; content:"@"; metadata:service finger; reference:arachnids,251; reference:cve,1999-0105; reference:nessus,10073; classtype:attempted-recon; sid:1000048; rev:10;)
3	(msg:"FINGER null request"; flow:to_server,established; content:"|00|"; metadata:service finger; reference:arachnids,377; reference:cve,1999-0612; classtype:attempted-recon; sid:1000044; rev:7;)
4	(msg:"GPL ICMP L3retriever Ping"; icode:0; itype:8; content:"ABCDEFGHIJKLMNOPQRSTUVWABCDEFGHI"; depth:32; reference:arachnids,311; classtype:attempted-recon; sid:2100466; rev:5;)
5	(msg:"GPL ICMP Destination Unreachable undefined code"; icode:>15; itype:3; classtype:misc-activity; sid:2100407; rev:9;)

IV. RESULTS AND DISCUSSIONS

In the first part of this section, we present the datasets used in our work. In the next part, the performance of the system for different attacks is described.

A. Datasets used

To evaluate the performance of the approach, a series of experiments were conducted using the pcap files that are available through various sources. For training and testing purpose, the ISCX 2012, Darpa 1999 and the real-time datasets captured from the network were considered. Snort IDS was used to match the rules with the packets stored in the pcap files and to generate alerts.

The MIT-DARPA 1999 [9] dataset provided by MIT Lincoln Lab consists of five weeks of normal and abnormal data which are stored as pcap files. We have used the inside.tcpdump data of week 4 in our experiment which is a set of unlabelled data.

The ISCX-2012 intrusion detection evaluation dataset presented by Shiravi, et al. [10] consists of labeled network traces including packet payload. The dataset is available publicly through their website.

The real-time dataset was captured for ICMP type attacks which were stored as pcap file in the victim machine.

Since the datasets contained many redundant packets, it had to be preprocessed before use. The datasets were preprocessed using wireshark and only the data that are relevant for the experiments were taken and tested.

B. Test Results

Table III shows the performance of our system for training and testing datasets after applying the evolutionary based techniques. The table provides the Average and the Best fitness values for each population P, tested for finger and ICMP alerts.

The average fitness of each population is calculated by using the formula (1). Best fitness is taken as the highest detection rate of the individuals. The result shows that the detection rate varies for different population based on the parameters used in the rules.

TABLE III. PERFORMANCE OF THE SYSTEM

P	Finger		ICMP	
	Avg Fitness	***Best Fitness***	***Avg Fitness***	***Best Fitness***
1	0.513	0.801	0.424	0.803
2	0.521	0.824	0.561	0.854
3	0.558	0.886	0.532	0.827
4	0.539	0.853	0.621	0.932
5	0.582	0.897	0.642	0.953
6	0.641	0.945	0.595	0.870
7	0.617	0.935	0.611	0.944
8	0.603	0.882	0.678	0.960
9	0.590	0.873	0.510	0.813
10	0.623	0.908	0.588	0.866

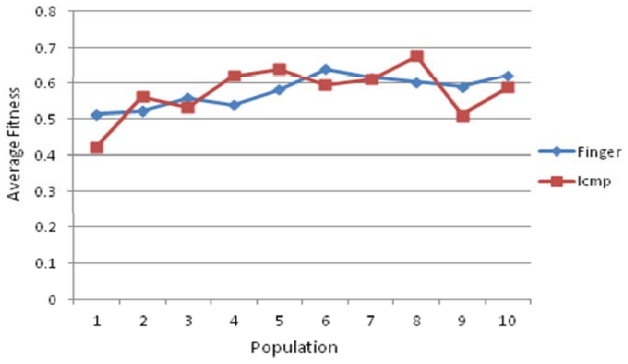

Fig. 2. Curves showing the variation of average fitness for finger and icmp alerts

Fig. 2 shows the variations of values for different generations for finger and ICMP alerts. As can be seen in the graph, population 6 has the highest average fitness value for finger alerts and population 8 for the ICMP alerts. The variation of fitness values between generations was mainly because of the difference in the parameters taken in different rules. The rule which contained few parameters showed high detection rate whereas the rule which contained more parameters showed low detection rate. This variation greatly affected the average fitness of the population in each generation. The population with high average fitness contained many fit individuals.

V. CONCLUSION AND FUTURE WORK

In this paper, a framework to autonomously generate the rules based on evolutionary computation based techniques has been proposed. The approach is capable of producing rules which are able to detect the attacks at a satisfactory level with a high detection rate. The test has been performed not only on the Darpa and ISCX datasets but also on the captured real network traffic.

The main drawback of the approach is that it works for a limited population size since the number of parameters taken was less. Also, it has been tested for few attack types and only for finding the detection rate. The work can be extended to detect more attacks and also for finding the solution for multiple objectives.

REFERENCES

[1] Roesch, Martin. "Snort: Lightweight Intrusion Detection for Networks." In LISA, vol. 99, no. 1, pp. 229-238. 1999.

[2] Vollmer, Todd, Jim Alves-Foss, and Milos Manic. "Autonomous rule creation for intrusion detection." Computational Intelligence in Cyber Security (CICS), 2011 IEEE Symposium on. IEEE, 2011.

[3] Gómez, J., Gil, C., Baños, R., Márquez, A. L., Montoya, F. G., & Montoya, M. G. "A multi-objective evolutionary algorithm for network intrusion detection systems." Advances in Computational Intelligence. Springer Berlin Heidelberg, 2011. 73-80.

[4] Gómez, J., Gil, C., Baños, R., Márquez, A. L., Montoya, F. G., & Montoya, M. G. "A Pareto-based multi-objective evolutionary algorithm for automatic rule generation in network intrusion detection systems." Soft Computing 17.2 (2013): 255-263.

[5] Wuu, Lih-Chyau, Chi-Hsiang Hung, and Sout-Fong Chen. "Building intrusion pattern miner for Snort network intrusion detection system." Journal of Systems and Software 80, no. 10 (2007): 1699-1715.

[6] Khamphakdee, Nattawat, Nunnapus Benjamas, and Saiyan Saiyod. "Improving Intrusion Detection System based on Snort rules for network probe attack detection." Information and Communication Technology (ICoICT), 2014 2nd International Conference on. IEEE, 2014.

[7] Wang, Hao, Somesh Jha, and Vinod Ganapathy. "NetSpy: Automatic generation of spyware signatures for NIDS." In Computer Security Applications Conference, 2006. ACSAC'06. 22nd Annual, pp. 99-108. IEEE, 2006.

[8] Agrawal, Rakesh, and Ramakrishnan Srikant. "Fast algorithms for mining association rules." Proc. 20th int. conf. very large data bases, VLDB. Vol. 1215. 1994.

[9] MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, http://www.ll.mit.edu/ideval/data/1999data.html

[10] Shiravi, Ali, Hadi Shiravi, Mahbod Tavallaee, and Ali A. Ghorbani. "Toward developing a systematic approach to generate benchmark datasets for intrusion detection." *Computers & Security* 31, no. 3 (2012): 357-374.

[11] Roesch, Martin. "Writing Snort Rules: How To write Snort rules and keep your sanity." (2001), http://www.snort.org.

[12] Ding, Yu-Xin, Hai-Sen Wang, and Qing-Wei Liu. "Intrusion scenarios detection based on data mining." In Machine Learning and Cybernetics, 2008 International Conference on, vol. 3, pp. 1293-1297. IEEE, 2008.

[13] Hwang, Kai, Min Cai, Ying Chen, and Min Qin. "Hybrid intrusion detection with weighted signature generation over anomalous internet episodes." Dependable and Secure Computing, IEEE Transactions on 4, no. 1 (2007): 41-55.