Volume 3, Issue 11, November 2013 ISSN: 2277 128X

International Journal of Advanced Research in Computer Science and Software Engineering

Research Paper

Available online at: www.ijarcsse.com

Implementation of intrusion Detection System using Genetic Algorithm

Ms Archana S Shidore [1] , Prof Vrunda Bhusari[2]

Department of Computer Science and Engineering

Bhivarabai Sawant Institute of Technology & Research (BSIOTR), India

Abstract:- In recent years, Intrusion Detection System (IDS) has become one of the hottest research areas in Computer Security. It is an important detection technology and is used as a countermeasure to preserve data integrity and system availability during an intrusion. When an intruder attempts to break into an information system or performs an action not legally allowed, we refer to this activity as an intrusion. IDSs can also be divided into two groups depending on where they look for intrusive behavior: Network-based IDS (NIDS) and Host-based IDS. The former refers to systems that identify intrusions by monitoring traffic through network devices (e.g. Network Interface Card, NIC). A host-based IDS monitors file and process activities related to a software environment associated with a specific host. Some host-based IDSs also listen to network traffic to identify attacks against a host. intrusion techniques may include exploiting software bugs and system misconfigurations, password cracking, sniffing unsecured traffic, or exploiting the design flaw of specific protocols An Intrusion Detection System is a system for detecting intrusions and reporting them accurately to the proper authority.

Keyword:- GA,IDS,HIDS,NIDS etc

I. Introduction:-

Genetic Algorithm

Genetic algorithm is a family of computational models based on principles of evolution and natural selection. These algorithms convert the problem in a specific domain into a model by using a chromosome-like data structure and evolve the chromosomes using selection, recombination, and mutation operators. Genetic Algorithm (GA) has been used in different ways in IDSs. GA uses different machine learning techniques, such as finite state machine, decision tree, and GA, to generate artificial intelligence rules for IDS. One network connection and its related behavior can be translated to represent a rule to judge whether or not a real-time connection is considered an intrusion. These rules can be modeled as chromosomes inside the population. The population evolves until the evaluation criteria are met. The generated rule set can be used as knowledge inside the IDS for judging whether the network connection and related behaviors are potential intrusions implemented an IDS using autonomous agents (security sensors) and applied AI techniques to evolve genetic algorithms. In the approaches described above, the IDS can be viewed as a rule-based system (RBS) and GA can be viewed as a tool to help generate knowledge for the RBS. These approaches have some disadvantages. In order to detect intrusive behaviors for a local network, network connections should be used to define normal and anomalous behaviors. Sometimes an attack can be as simple as scanning for available ports in a server or a password-guessing scheme. But typically they are complex and are generated by automated tools that are freely available from the Internet. An example can be a Trojan horse or a backdoor that can run for a period of time, or can be initiated from different locations. In order to detect such intrusions, both temporal and spatial information of network traffic should be included in the rule set. The current GA applications do not address these issues extensively. This paper shows how network connection information can be modeled as chromosomes and how the parameters in genetic algorithm can be defined in this respect. Some examples are used to show the implementation.

II. Intrusion Detection Overview

The below sections give a short overview of networking attacks, classifications and various

Networking Attacks

The main four major categories of networking attacks. Every attack on a network can comfortably be placed into one of these groupings

Denial of Service (DoS): A DoS attack is a type of attack in which the hacker makes a computing or memory resources too busy or too full to serve legitimate networking requests and hence denying users access to a machine e.g. apache, smurf, neptune, ping of death, back, mail bomb, UDP storm etc. are all DoS attacks.

Remote to User Attacks (R2L): A remote to user attack is an attack in which a user sends packets to a machine over the internet, which s/he does not have access to in order to expose the machines vulnerabilities and exploit privileges which a local user would have on the computer e.g. xlock, guest, xnsnoop, phf, send mail dictionary etc.

User to Root Attacks (U2R): These attacks are exploitations in which the hacker starts off on the system with a normal user account and attempts to abuse vulnerabilities in the system in order to gain super user privileges e.g. perl, xterm.

Probing: Probing is an attack in which the hacker scans a machine or a networking device in order to determine weaknesses or vulnerabilities that may later be exploited so as to compromise the system. This technique is commonly used in data mining e.g. saint, portsweep, mscan, nmap etc.

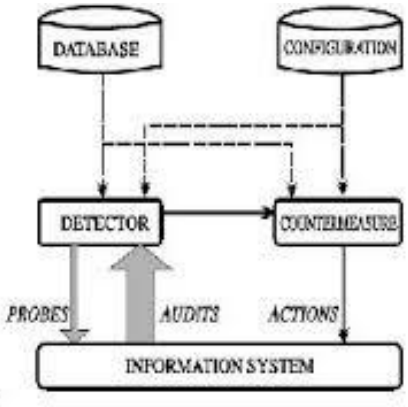
Classification of Intrusion Detection

Intrusions Detection can be classified into two main categories.

Host Based Intrusion Detection: HIDSs evaluate information found on a single or multiple host systems, including contents of operating systems, system and application

Network Based Intrusion Detection: NIDSs evaluate information captured from network communications, analyzing the stream of packets which travel across the network .

Intrusion Detection System



Intrusion detection is defined as the process of intelligently monitoring the events occurring in a computer system or network, analyzing them for signs of violations of security policy. The primary aim of Intrusion Detection System (IDS) is to protect the availability, confidentiality and integrity of critical networked information systems. Intrusion Detection Systems are an important component of defensive measures protecting computer systems and networks from abuse. When an IDS is properly deployed it can provide warnings indicating that a system is under attack. It is critical for intrusion detection in order for the IDS to achieve maximal performance.

An intrusion detection system can be described at a very macroscopic level as a detector that processes information coming from the system to be protected. This detector can also launch probes to trigger the audit process, such as requesting version numbers for applications. It uses three kinds of information: long-term information related to the technique used to detect intrusions (a knowledge base of attacks for example), configuration information about the current state of the system, and audit information describing the events that are happening to the system. The role of the detector is to eliminate unneeded information from the audit trail. It then presents either a synthetic view of the security-related actions taken during normal usage of the system, or a synthetic view of the current security state of the system. A decision is then taken to evaluate the probability that these actions or this state can be considered as symptoms of an intrusion or vulnerabilities. A countermeasure component can then take corrective action to either prevent the actions from being executed or change the state of the system back to a secure state.

Crossover and Mutation Operator

The generated subnet of an input node is the set of its all output connections. The generated subnet of a hidden node is the set of itself, its all input and output connections. The generated subnet of an output node is the set of itself and its all input connections. The connection information includes whether or not the connection exists and the connection weight value. The node information includes bias and activation function. The employed mutation operator, which can prevent the premature convergence in evolution, includes five operations: node deletion, connection deletion, connection addition, node addition and weight adjustment. The number of output nodes is invariable, so the deletion and addition of node are limited for input and hidden nodes. The node deletion means making a node inoperative and deleting its generated subnet. The node addition is that an inoperative node is turned operative and assigned connection weights with the other operative nodes. The mutation operator has the same influence as the crossover, thus the necessary step following mutation operation is the same as the crossover. There are also other parameters that need to be considered. These parameters should be adjusted according to the application environment of the system and the organizations security policy

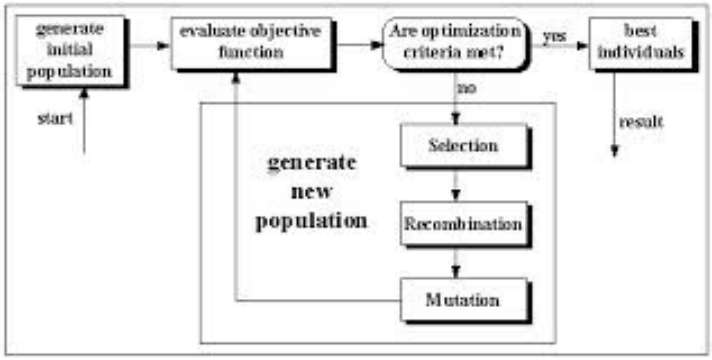


Fig 1 Structure of Genetic algorithms

GA Operators According to the figure above, the selection, mutation and crossover are the most effective parts in the algorithm as they are they participate in the generation of each population.

- Selection is the phase where population individuals with better fitness are selected, otherwise it gets damaged.
- Crossover is a process where each pair of individuals selects randomly participates in exchanging their parents with each other, until a total new population has been generated.
- Mutation flips some bits in an individual, and since all bits could be filled, there is low probability of predicting the change.

Algorithm: Intrusion Detection

Input: Inflowing network connection
Output: Decision if connection is intrusive or not
1. Loop Forever {fetch incoming packet
(network probe)}
2. For each rule in base
3. Match rule to network connection (analysis
console)
4. If rules match then
5. Mark connection as intrusion (policy control)
6. End if
7. End For Each
8. End Loop Forever

III. Experimental Set Up

1) OBJECTIVE

The scope of our experiment was focused to generate classifiers or rules for six attack types belonging to two different classes. The training set contains maximum connections of the smurf attack type, 280,790 to be precise. The number of other connections are: 107201 neptune connections, 21 land connections, 15 satan connections, 30 ipsweep connections and 15 portsweep connections. Hence, we wanted to create a rule that can classify all of these connections with a minimal false positive rate. Although we would have preferred to extend our implementation to all the attack types and connection features, the enormous training time complexity of the algorithm, very large data sets and lack of time restricted us.

2) TOOLS

For our implementation, we have used the GALIB C++library especially suited to develop GAs Owing to the large hypothesis search space and high time complexity, we wanted to use a tool or library that is high on performance and computing speed. After a comprehensive survey of many tools, we decided to use GALIB since it is a C++ library, has been widely used by other researchers and well documented. We used a LINUX based Dell computer with a Pentium 4 processor, 120GB of hard disk space and 1 GB of RAM to execute the computer program.

3) HYPOTHESIS SEARCH SPACE

In this experiment, we restricted our hypothesis search space to the eight most important fields that we could identify. Although, we can extend our search space to all 41 fields, that will however require the computation to go over many hours. Hence, for our experiment we confined our search space to only eight important fields. We selected these fields based on a heuristic analysis of the training data to identify potential fields that seemed unique to a particular attack type. These fields provide information on the type of protocol, type of service, flag (error or normal connection), duration of bytes sent, duration of the connection, % of connections to different hosts, number of operations on access control files and number of outbound commands in an ftp control session features of the connection. From our analysis of the training data, these fields appear to be amongst the decisive fields that can help identify an attack from a normal connection. We intend to extend our implementation to more fields for next experiments. The population size for each generation contains 100 individuals.

4) GENETIC OPERATORS and PARAMETERS

We say that an individual matches an attack type when all the eight fields that constitute our search space of the individual match those of the attack connection. The rate of crossover was set to 0.6 i.e. given 100 individuals in any population, 60 best fit individuals will be selected based on highest fitness score and be made to undergo crossover to create offspring. Since at present we are exploring only eight fields, the crossover occurs only over these fields. Out of the 60, the best 40 parents are then selected to complete the population size of 100. Thus, the best fit parents also participate in the subsequent generations. The mutation rate has been affixed to 1% wherein, only 1 individual out of a population size of 100 undergoes a change in one of the eight fields as explained in section 4.3.

5) RESULTS

From the above experiment, we were able to create a rule that could successfully classify all of the 280,790 smurf type of attack connections. Along with this, it also classified 410 normal connections as attack. The false positive rate is thus around 0.08%. In the entire testing data set, the smurf rule set correctly classified 2,807,880 connections, and with a false positive of 0.17%. The rule set that classifies Probe attacks was able to correctly classify 52 Probe attack connections in the training data set, out of a total of 60 probe type of attacks. The rule set for Probes on the entire test data showed

results as follows total number of probe attacks = 38,786, total classified correctly = 35,829. The percentage accuracy = 92.3%. These are encouraging results considering we have used only 8 fields. All these results are shown in Figure 2.

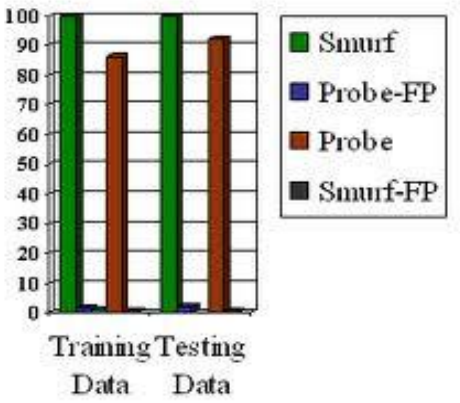


Figure 2: Graph showing percentage accuracy of the rule base for the two attack classes namely smurf (DOS) and probe attacks, along with the falsepositive rates denoted by FP.

6) FUTURE WORK

Other than searching over the entire hypothesis space of 41 features, there are certain areas, which if looked into can improve the performance of our system. At present, our fitness criterion to select fit individuals is based on the eight fields that match exactly with the attack connection. One can create a statistics driven metric to find the range of values for each of these eight fields that would best classify the attack labels. One could also use data mining techniques to identify the most significant fields that can identify the attack connections. This would add more scientific reason to the search space of the algorithm.

IV. Conclusion:-

In this work we have deployed genetic algorithm approach to intrusion detection. Software implementation of the proposed approach is presented. Genetic algorithm was used to obtain classification rules for intrusion detection while correlation technique was used to identify the most important features of network connections. As in real word types of intrusions are changing rapidly and becoming increasingly complex, an intrusion detection system should be adaptive in order to be able to cope with the evolution of the threat-space. As our system can upload and update new data y evolve new rules for detecting new intrusions, it is adaptive and representation of the rules and effective fitness functions that can be applied, is easy to implement and maintain

High attack detection rate and low false-positive rate demonstrate advantages of applying this technique to intrusion detection without using any complementary technique typically used with other soft-computing techniques. Our system is using only three features of the network connections maintaining high detection rates, so it can perform intrusion detection process fast and could be applied to high speed networks.

References

[1] S. Selvakani Kandeeban" A Mutual Construction for IDS Using GA" International Journal of Advanced Science and Technology Vol. 29, April, 2011.

[2] P.Lakshami Prasanna "Intrusion Detection System Using Genetic Algorithm"International journal of p2p Network Trends and Technology volume 1 issue 2 -2011

[3] Mohammad Sazzadul Hoque "Implementation of Genetic Algorithm" International Journal of Network Security & its Application" Vol 4 No2 March 2012.

[4] A.O. Eboka, O.E. Okonta "Genetic Algorithm Rule-Based Intrusion Detection System" Journal of Emerging Trends in Computing and Information Sciences.

