Information and Knowledge Management
ISSN 2224-5758 (Paper) ISSN 2224-896X (Online)
Vol 1, No.1, 2011

A Genetic Algorithm Based elucidation for improving Intrusion Detection through condensed feature set by KDD 99 data set

S. Selvakani Kandeeban (Corresponding author)

Professor and Head, Department of Computer Applications, Francis Xavier Engineering College, Tirunelveli, Tamilnadu, India

E-mail: sselvakani@hotmail.com

Dr. R.S. Rajesh

Associate Professor, Department of Computer Science and Engineering,

Manonmanium Sundaranar University, Tirunelveli, Tamilnadu, India.

E-mail: rs_rajesh@yahoo.co.in

The research is financed by Asian Development Bank. No. 2006-A171(Sponsoring information)

Abstract

An Intrusion detection system's main aim is to identify the normal and intrusive activities. The objective of this paper is to incorporate Genetic algorithm with reduced feature set into the system to detect and classify intrusions from normal. The experiments and evaluations of the proposed method were done using KDD cup 99 dataset. The Genetic algorithm is used to derive a set of rules from the reduced training data set, and the fitness function is employed to judge the quality of rules.

Keywords: Genetic Algorithm, Detection Rate, Intrusion Detection System, Reduced Feature Set, KDD 99 data set.

1. Introduction

In the last five years, the information revolution has finally come of age. More than ever before, we see that the Internet has changed our lives. The possibilities and opportunities are limitless; unfortunately, so too are the risks and likelihood of malicious intrusions. Intruders can be classified into two categories: outsiders and insiders. Outsiders are intruders who approach your system from outside of your network and who may attack your external presence (ie. deface web servers, forward spam through e-mail servers, etc.) They may also attempt to go around the firewall and attack machines on the internal network. Insiders, in contrast, are legitimate users of your internal network who misuse privileges, impersonate higher privileged users, or use proprietary information to gain access from external sources (Axelsson 2000).

A single intrusion of a computer network can result in the loss or unauthorized utilization or modification of large amounts of data and causes users to question the reliability of all of the information on the network. There are numerous methods of responding to a network intrusion, but they all require the accurate and timely identification of the attack. Intrusion detection systems (IDS) have emerged to detect actions which endanger the integrity, confidentiality or availability of a resource as an effort to provide a solution to existing security issues.

The act of detecting actions that attempts to compromise the confidentiality, integrity or availability of a resource (Kruegel C et.al 2002). Intrusion is the act of violating the security policy or legal protections that pertain to an information system. An intruder is somebody ("hacker" or "Cracker") attempting to break into or misuse your system (Kayacik et.al 2003).

Stateless firewalls suffer from several significant drawbacks that make them insufficient to safeguard networks by themselves. The major drawbacks to stateless firewalls are

- ✓ They cannot check the data (payload) that packets contain.
- ✓ They do not retain the state of connections

- ✓ The TCP can only be filtered in the 0th fragments
- ✓ Public services must be forwarded through the filter

There are two general categories of attacks which intrusion detection technologies attempt to identify - anomaly detection and misuse detection, Anomaly detection identifies activities that vary from established patterns for users, or groups of users. Anomaly detection typically involves the creation of knowledge bases that contain the profiles of the monitored activities. The second general approach to intrusion detection is misuse detection. This approach involves the comparison of a user's activities with the known behaviors of attackers attempting to penetrate a system (Axelsson 2000). While anomaly detection typically utilizes threshold monitoring to indicate when a certain established metric has been reached, misuse detection approach frequently utilize a rule-based approach. When applied to misuse detection, the rules become scenarios for network attacks. The intrusion detection mechanism identifies a potential attack if a user's activities are found to be consistent with the established rules (Bass T. 2000).

The majority of currently existing IDS face a number of challenges such as low detection rates and high false alarm rates (Wlodzislaw D. 2003), which falsely classifies a normal connection as an attack and therefore obstructs legitimate user access to the network resources. These problems are due to the sophistication of the attacks and their intended similarities to normal behavior.

To overcome low detection rate and high false alarm problems in currently existing IDS, we propose a Genetic Algorithm based Intrusion detection system to enhance the performance of intrusion detection for rare and complicated attacks.

The remainder of the paper is organized as follows; Section 2 presents related works of intrusion detection systems with Genetic Algorithm. Section 3 presents the data set which we used for training and testing. Section 4 introduces our proposal system. Section 5 shows the experiments and results and in Section 6 are the conclusions and future works.

2. Literature Survey

In this paper a prototype Intelligent Intrusion Detection System was developed by vautkeskhovich in 2009 to demonstrate the effectiveness of data mining techniques that utilize Fuzzy Logic and Genetic Programming. The system provides a high degree of detection for both anomaly and misuse. Genetic algorithms are used to tune the fuzzy membership function to improve performance and also to provide data mining components with the set of features from the audit data. The aim was to design and abstract IDS which is adaptive, accurate and flexible. In this experiment, GA was found very effective in selecting set of features to identify different types of intrusions.

Chittur's paper in 2001 presents a novel approach to detect the intrusions by using a Genetic Algorithm approach. The goal from this paper is to test whether Genetic Algorithm is a feasible option for model generation in an artificial intelligence- based Intrusion Detection System, designed to replace or reinforce fingerprinting systems. According to the researcher, one network connection and its related behavior can be translated to represent a rule. These rules are used to judge whether or not a real-time connection is considered an intrusion. These rules are modeled as chromosomes. The generated rule set can be used as knowledge inside the Intrusion Detection System for judging whether the network connection and related behaviors are potential intrusions.

The Abraham's paper in 2004 presents a soft computing approach to detect intrusions in a network. The experimental results clearly show that soft computing approach could play a major role for intrusion detection. Author investigated the suitability of linear genetic programming (LGP) technique to model fast and efficient intrusion detection systems. Linear Genetic Programming examines the evolution of imperative computer programs written as linear sequences of instructions. In contrast to functional expressions or syntax trees used in traditional Genetic Programming (GP), Linear Genetic Programming (LGP) employs a linear program structure as genetic material whose primary characteristics are exploited to achieve acceleration of both execution time and evolutionary progress.

GA can be used to evolve simple rules for network traffic; these rules are used to differentiate normal connections from anomalous connections, the rule (Li.W. 2004) used in form of (If {condition} then {act}) where the condition refers to a match between current network connection and the rules in network based IDS, such as source and destination IP addresses and port numbers (used in TCP/IP network protocols), duration of the connection, protocol used, etc., indicating the probability of an intrusion. The act field 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.

3. KDD Data set

In 1998, the Defense Advanced Research Projects Agency (DARPA) intrusion detection evaluation created the first standard corpus for evaluating intrusion detection systems. The 1998 off-line intrusion detection evaluation was the first in a planned series of annual evaluations conducted by the Massachusetts Institute of Technology (MIT) Lincoln Laboratories under DARPA sponsorship. The corpus was designed to evaluate both false alarm rates and detection rates of intrusion detection systems using many types of both known and new attacks embedded in a large amount of normal background traffic. Over 300 attacks were included in the 9 weeks of data collected for the evaluation. These 300 attacks were drawn from 32 different attack types and 7 different attack scenarios as shown in KDD achieve.

Initial observations of the evaluation results for the 1998 competition concluded that most IDSs can easily identify older, known attacks with a low false-alarm rate, but do not perform as well when identifying novel or new attacks. Several additional intrusion detection contests, such as DARPA 1999 and KDD Cup 1999, used similar data sets to evaluate results in intrusion detection research. The DARPA 1999 evaluation used a similar structure for the contest, but included Widows NT workstations in the simulation network. These evaluations of developing technologies are essential to focus effort, document existing capabilities, and guide research.

The DARPA evaluation focused on the development of evaluation corpora that could be used by many researchers for system designs and refinement. The evaluation used the Receiver Operating Characteristic (ROC) technique to assess intrusion detection systems. The ROC approach analyzed the tradeoff between false alarm rates and detection rates for detection systems. ROC curves for intrusion detection indicate how the detection rate changes as internal thresholds are varied to generate fewer more or fewer false alarms to tradeoff detection accuracy against analyst workload.

The training data was about four gigabytes of compressed binary tcpdump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records. Scoring focused on the systems ability to detect novel attacks in the test data that was a variant of a known attack labeled in the training data. The KDD ’99 training datasets contained a total of 24 training attack types, with an additional 14 attack types in the test data only.

Participants were given a list of high-level features that could be used to distinguish normal connections from attacks. A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to and from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes. Three sets of features were made available for analysis. First, the ``same host" features examine only the connections in the past two seconds that have the same destination host as the current connection, and calculate statistics related to protocol behavior, service, etc. The similar ``same service" features examine only the connections in the past two seconds that have the same service as the current connection. "Same host" and "same service" features are together called time-based traffic features of the connection records. Some probing attacks scan the hosts (or ports) using a much larger time interval than two seconds, for example once per minute. Therefore, additional features can be constructed using a window of 100 connections to the same host instead of a time window. This yields a set of so-called host-based traffic features. Finally, domain knowledge can be used to create features that look for suspicious behavior in the data portions, such as the number of failed login attempts. These features are called “content” features.

The four (KDD archieve 1999) different categories of attack patterns are:

a. Denial of Service (DOS) Attacks: A denial of service attack is a class of attacks in which an attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine. Examples are Apache2, Back, Land, Mail bomb, SYN Flood, Ping of death, Process table, Smurf, Syslogd, Teardrop, Udpstorm.

b. User to Superuser or Root Attacks (U2Su): User to root exploits are a class of attacks in which an attacker starts out with access to a normal user account on the system and is able to exploit vulnerability to gain root access to the system. Examples are Eject, Ffbconfig, Fdformat, Loadmodule, Perl, Ps, Xterm.

c. Remote to User Attacks (R2L): A remote to user attack is a class of attacks in which an attacker sends packets to a machine over a network-but who does not have an account on that machine; exploits some vulnerability to gain local access as a user of that machine. Examples are Dictionary, Ftp_write, Guest, Imap, Named, Phf, Sendmail, Xlock, Xsnoop.

d. Probing (Probe): Probing is a class of attacks in which an attacker scans a network of computers to gather information or find known vulnerabilities. An attacker with a map of machines and services that are

available on a network can use this information to look for exploits. Examples are Ipsweep, Mscan, Nmap, Saint, Satan.

4. Proposed System

Our methodology is divided in to three phases as follows. Features were reduced based on Correlation and optimized rules were developed using Genetic Algorithm and new attack patterns were detected by learning the patterns from the Neural Network Radial Basis Function Networks as shown in Figure 1.

As the first step in our work is to cope with the speed problem mentioned above, we have used the results obtained in our previous work where we deployed Information Gain based Mutual Information, in order to extract the most relevant features of the data. In this way the total amount of data to be processed is highly reduced. As an important benefit of this arises the high speed of training the system thus providing high refreshing rate of the rule set.

Subsequently, these features are used to form rules for detecting various types of intrusions using Genetic Algorithm. This permits the introduction of higher level of generality and thus to higher detection rates. The procedure starts from an initial population of randomly generated individuals. Then the population is evolved for a number of generations while gradually improving the qualities of the individuals in the sense of increasing the fitness value as the measure of quality. The final step is to train the network using Radial Basic Function Network to detect the unknown attack.

5. Genetic Algorithm

Genetic algorithms are now widely applied in science and engineering as adaptive algorithms for solving practical problems. Certain classes of problem are particularly suited to being tackled using a GA based approach.

The general acceptance is that GAs are particularly suited to multidimensional global search problems where the search space potentially contains multiple local minima. Unlike other search methods, correlation between the search variables is not generally a problem. The basic GA does not require extensive knowledge of the search space, such as likely solution bounds or functional derivatives. A task for which simple GAs are not suited is rapid local optimization; however, coupling the GA with other search techniques to overcome this problem is trivial.

Since the variable values are represented in binary, there must be a way of converting continuous values into binary, and visa versa. Quantization samples a continuous range of values and categorizes the samples into non-overlapping sub-ranges. Then a unique discrete value is assigned to each sub-range. The difference between the actual function value and the quantization level is known as the quantization error. Quantization begins by sampling a function and placing the samples into equal quantization levels. Any value falling within one of the levels is set equal to the mid, high, or low value of that level. In general, setting the value to the mid value of the quantization level is best, because the largest error possible is half a level. Rounding the value to the low or high value of the level allows a maximum error equal to the quantization level.

GA evolves a population of initial individuals to a population of high quality individuals, where each individual represents a solution of the problem to be solved as shown in figure 2. Each individual is called chromosome, and is composed of a predetermined number of genes. The quality of each rule is measured by a fitness function as the quantitative representation of each rule's adaptation to a certain environment. The procedure starts from an initial population of randomly generated individuals. Then the population is evolved for a number of generations while gradually improving the qualities of the individuals in the sense of increasing the fitness value as the measure of quality. During each generation, three basic genetic operators are sequentially applied to each individual with certain probabilities, i.e. selection, crossover and mutation.

Each individual in the population consists of a fixed number of rules. In other words, the individual itself is a complete solution candidate. In our current implementation, we set this fixed number as 10 which well satisfy the requirement of our testing databases. The antecedent of a certain rule in the individual is formed by a conjunction of n attributes, where n is number of attributes being mined. K bits will be used to stand for an attribute if this attribute has k possible values. Continuous attributes will be partitioned to threshold-based Boolean attribute in which the threshold is a boundary (adjacent examples across this boundary differ in their target classification) that maximizes the information gain. Therefore two bits will be used for a continuous attribute. The consequent of a rule is not explicitly encoded in the string. In contrast, it is automatically given based on the proportion of positive/negative examples it matches in the training set.

It is very important to define a good fitness function that rewards the right kinds of individuals. We try to consider affecting factors as complete as possible to improve the results of classification. Our fitness function is

defined as following:

Fitness = Error rate + Entropy measure + Rule consistency (1)

If a rule matches a certain example, the classification it gives is its consequent part. If it doesn't match, no classification is given. An individual consists of a set of rules, the final classification predicted by this rule set is based on the voting of those rules that match the example. The classifier gives all matching rules equal weight. For instance, in an individual (which has ten rules here), one rule doesn't match, six rules give positive classification and three rules give negative classification on a training example, then the final conclusion given by this individual on that training example is positive. If a tie happens (i.e., four positive classifications and four negative classifications), the final conclusion will be the majority classification in the training examples. If none of the rules in the individual matches that example, the final conclusion will also be the majority classification in the training examples. The error rate measure of this individual is the percent of misclassified examples among all training examples.

The rationale of using entropy measure in fitness function is to prefer those rules that match less examples whose target values are different from rule's consequent. High accuracy does not implicitly guarantee the entropy measure is good because the final classification conclusion of a certain training example is based on the comprehensive results of a number of rules. It is very possible that each rule in the individual has a bad entropy measure but the whole rule set still gives the correct classification. Keeping low entropy value of an individual will be helpful to get better predicting results for untrained examples.

6. Experiments and Results

In this section, we summarize our experimental results to detect Anomaly intrusion detections using Distributed Time-Delay Artificial Neural Network over KDD99 dataset. The full training set of the KDD99 dataset has 4,898,431 connections covering normal network traffic and four categories of attacks For our experiments, the training dataset consist of 50000 patterns (8000 patterns for each class of DoS, R2L, U2R, Probe, Normal), and testing dataset consist of 10000 patterns (2500 patterns for each class). We are only interested in knowing to which category (Normal, DoS, R2l, U2R, Probe) a given connection belonged. The accuracy of each experiment is based on detection rate on train and test dataset, where

Detection rate = (number of correctly classified instance / number of instance in the test dataset) (2)

Some of the rules which are extracted as shown in figure 3 for DoS attacks is listed below:

Rule 1:

```
If protocol_type=tcp then
   If service=http then back
      Else if service= private then neptune
         else  if service =finger | telnet then
            if count=1 then land
               if  count >=1&& <=302 then Neptune
```

Rule 2:

```
If protocol_type=tcp then
   If service=http then
     If logged_in=1 then
        If dst_host_count =255 then
           If dst_host_same=0 then back
```

Rule 3:

```
If  Protocol_type =tcp then
      if service = http then back
```

Rule 4:

```
if Src_bytes = 54540 || Src_bytes >= 16384 && Src_bytes <= 53816 then
        if dst_bytes =8314 || dst_bytes >=1460 && <= 8315 then back
```

Rule 5:

```
if Logged_in =1 then
        if count >=1 && <=7 then back
```

Rule 6:

```
if Srv_count >=1 && <=46 then
         if Dst_host_count =255 then
               if Dst_host_same =0  then back
```

Rule 7:

```
If  Protocol_type =icmp then
        if service = ecr_i then
           if Src_bytes = 1480||564 then
              if dst_bytes =0 then
                 if Logged_in =0 then pod
```

Out of the Six Old Dos Attacks, only 3 attacks namely back attack, land attack, and Neptune attacks have more than 90% detection rate as depicted in the bar diagram Figure 4. However, other attacks have the detection rate of less than 50%. The pod attack, smurf attack, and smurf attack are the attacks mainly using ICMP. The packets using ICMP are included as smaller part in the DARPA Training Set than the DoS attacks using UDP or TCP. Thus, the information about the packets using ICMP cannot sufficiently influence the correlation, adopts the concept of the information entropy. Similarly, the information of the neptune attack is not enough to be reflected because most packets in neptune attack are destined to the telnet port, but the large number of packets destined to the telnet port are also contained in the data of negative class.

The figure 5 represents the detection rates of the new kinds of DoS attacks. As you can see in this graph, the apache-2 attack is detected for 100% detection rate in spite of new kinds of attack. This is because the patterns of the encoded data for apache-2 attack have similar patterns to the old DoS attacks. However the other attacks such as the mailbomb attack, process table attack, and UDP storm are rarely detected because the patterns of the encoded data are very different from the patterns of the old DoS attacks.

Similarly for other types of attacks U2R, R2L, Probe also, our methodology shows a very good Result, having a detection rate of 96.5 %, 92.1%, 95.9% respectively.

In practical evaluation, we usually use the rule base instead of single rule to the performance of Intrusion Detection System based on Genetic Algorithm. We execute 5000 runs to evaluate the performance of our system, since we get different rules every time. We use as evaluation metrics the False Positive Rate and the Detection Rate. Generally speaking we say that an Intrusion Detection approach is good if it has high detection with low False Positive Rate. The distribution of Detection rate for the rule base in 5000 run is illustrated in the figure 6. When there are more rules in the rule base, we have a high detection rate and thus will possibly have a low false positive rate.

There are however some limitations with this evaluation method. First, it does not consider the completeness of the testing data set. The training set only contains a total of 24 attack types with an additional 14 types included in the testing set. Even if the rules perform well under the testing data set they may possible carry errors that are not identified due to the incompleteness of the testing data set. A workable alternative consists of improving the completeness of the testing set and increasing the coverage of rules.

7. Conclusions and Future works

In this paper, Intrusion Detection System overview is presented and various approaches for applying genetic algorithms for network intrusion detection are discussed. GA as evolutionary algorithms was successfully used in different types of IDS. Using GA returned impressive results, the best fitness value was very closely to the ideal fitness value. GA is a randomization search method often used for optimization problem. GAs implemented successfully in IDS in an offline training system for deriving rules from historical data, and in an online detection system that uses the generated rules to classify incoming network connections in real time environment. More techniques are to be investigated for enhancement of the performance of GA based Intrusion Detection System.

Future work includes creating a standard test data set for the genetic algorithm proposed in this work and applying it to a test environment. Detailed specification of parameters to consider for genetic algorithm should be determined during the experiments. In the future, we will hope to detected attackers in each class of (DoS, R2L, U2R, probing) combine Artificial neural network methods and fuzzy logic to improve the accuracy of IDS. Combining knowledge from different security sensors into a standard rule base is another promising area in this work.

References

[1] Abraham A. 2004, Evolutionary Computation in Intelligent Network Management, *Evolutionary Computing in Data Mining, Springer,* 189—210.

[2] Axelsson S. 2000, Intrusion Detection Systems: A Survey and Taxonomy, *Technical Report, Dept. of Computer Engineering*, Chalmers University.

[3] Bass T. 2000, Intrusion detection systems and multisensor data fusion, *Communications of the ACM*,

Vol.43 , 99–105.

[4] Chittur. A. 2001, Model Generation for an Intrusion Detection System Using Genetic Algorithms, *Ossining High School*, Ossining NY.

[5] Kayacik G., Zincir-Heywood N., Heywood M.2003, On the Capability of an SOM-based Intrusion Detection System, *Proceedings of International Joint Conference on Neural Networks*.

[6] Kruegel C and F.Valeur. 2002, Stateful Intrusion Detection for High-Speed Networks, *Proceedings of the IEEE Symposium on Research on Security and Privacy*, 285–293.

[7] Li .W. 2004, Using Genetic Algorithm for Network Intrusion Detection, *Proceedings of the United States Department of Energy Cyber Security Group*.

[8] The KDD Archive. KDD99 cup dataset, 1999. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[9] Vaitsekhovich, L. 2009.,Intrusion detection in TCP/IP networks using immune systems paradigm and neural network detectors, Brest State Technical University, *XI International PhD Workshop*. http://www.cs.ucc.ie/misl/publications/files/idssteinebach.pdf.

[10] Włodzisław D, W. Tomasz, B. Jacek and K. Adam. 2003, Feature Selection and Ranking Filters. http://metet.polsl.katowice.pl/~jbiesiada/prace/selekcja/03-Istambul.pdf.

Dr. S. Selvakani Kandeeban received the MCA degree from Manonmanium Sundaranar University and M.Phil degree from Madurai Kamaraj University. Presently she is working as an Professor & Head, MCA Dept in Francis Xavier Engineering College, Tirunelveli. Previously she was with Jaya College of Engineering and Technology as an Assistant Professor, MCA Department. She has presented 4 papers in National Conference and 1 paper in international conference. She has published 2 paper in National journal and 11 papers in International Journal. She received her PhD degree in Computer Science from Mother Teresa Women's University in the Year 2011. Her area is Network Security.

Dr. R. S Rajesh received his B.E and M.E degrees in Electronics and Communication Engineering from Madurai Kamaraj University, Madurai, India in the year 1988 and 1989 respectively, and completed his Ph.D in Computer Science and Engineering from Manonmaniam Sundaranar University in the year 2004. In September 1992 he joined in Manonmaniam Sundaranar University where he is currently working as an Associate Professor in the Computer Science and Engineering Department.

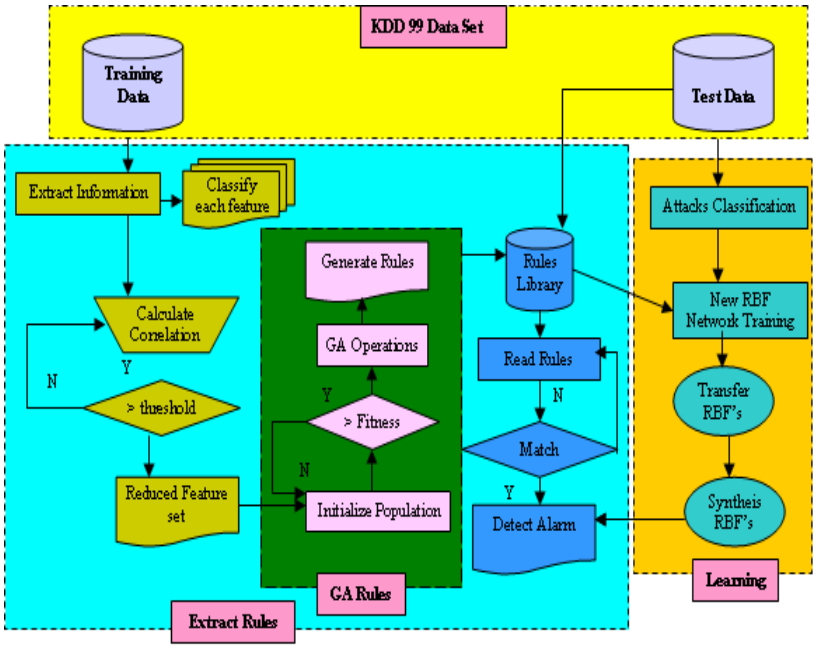

Figure 1: System Flow

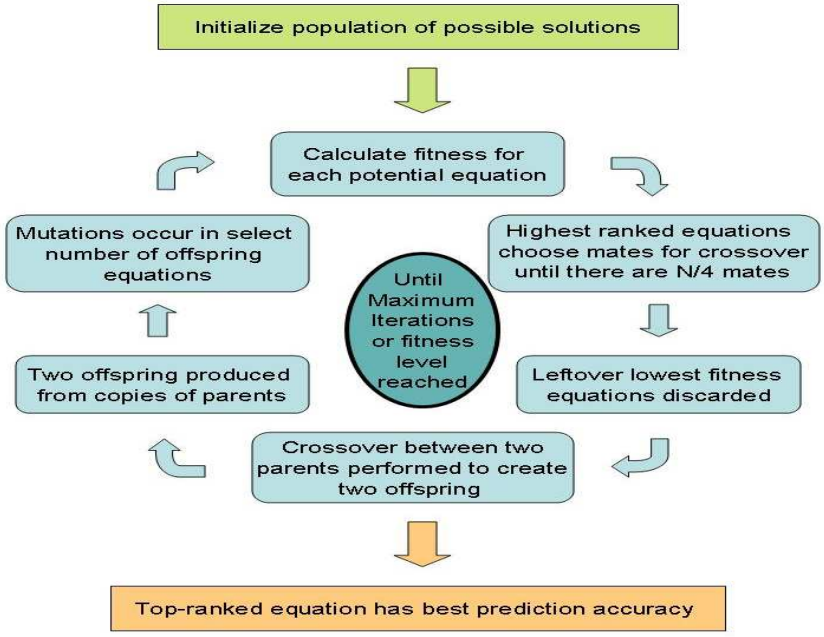

Figure 2: Genetic algorithm structure

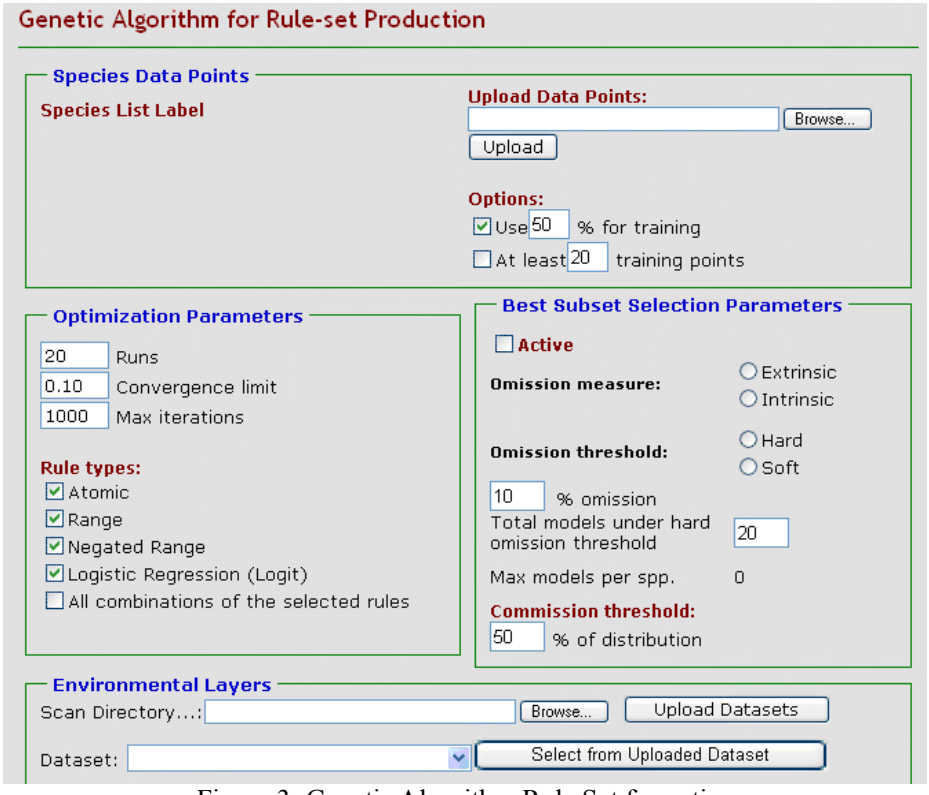

Figure 3: Genetic Algorithm Rule Set formation

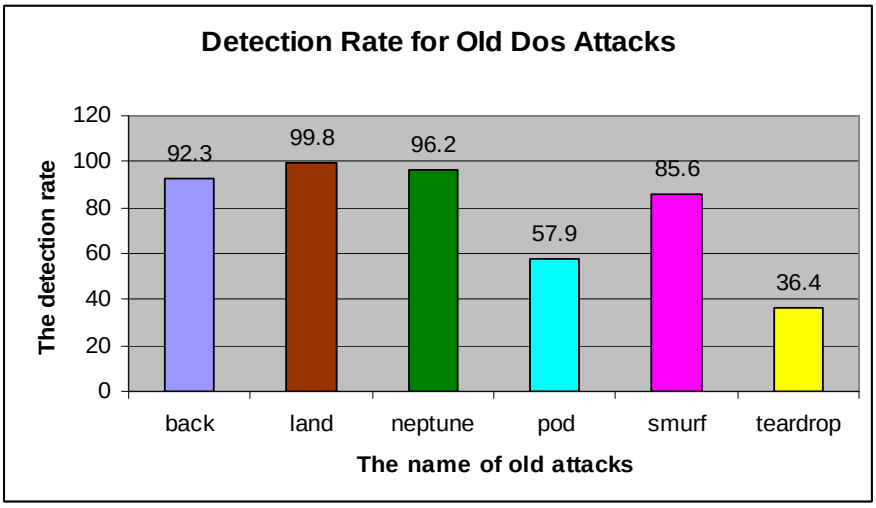

Figure 4: Detection rate for old DoS attacks

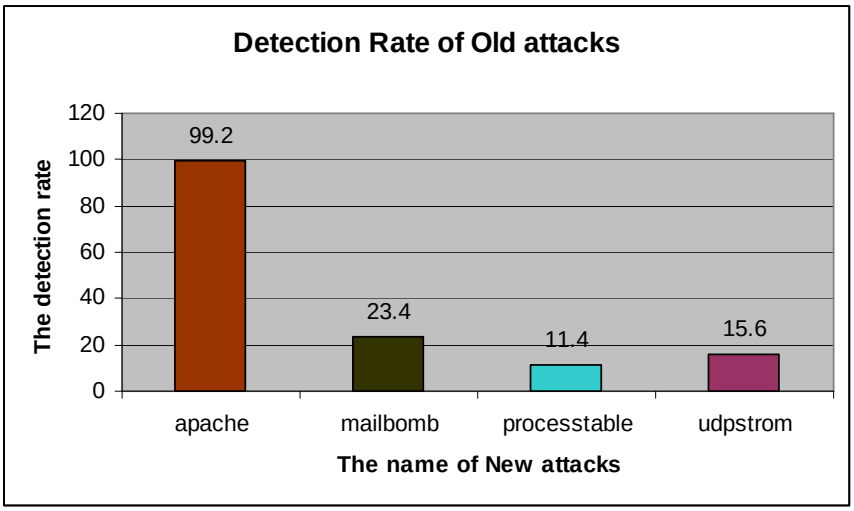

Figure 5: Detection rate for new DoS attacks

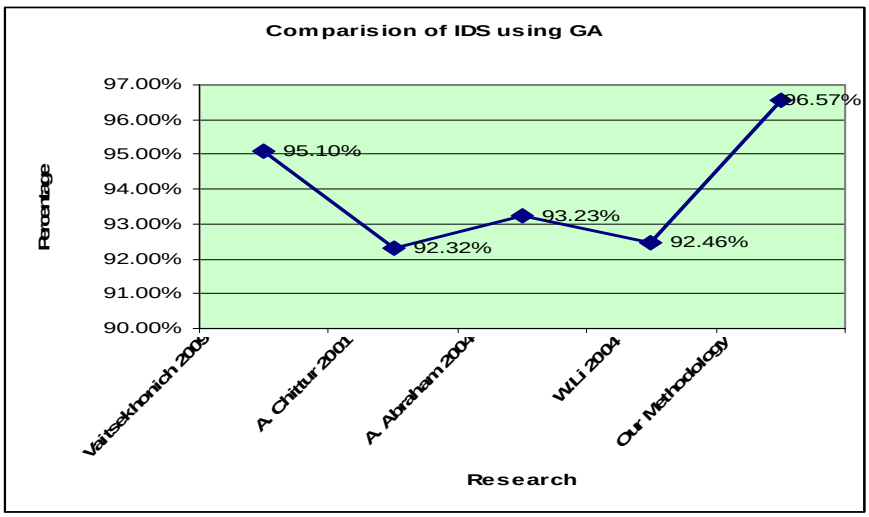

Figure 6: Detection rate for new DoS attacks