Intelligent Intrusion Detection System using SVM and Genetic Algorithm (SVM-GA)

Namita Parati 1, Sumalatha Potteti2
1Assistant Professor, Department of CSE, BRECW, Hyderabad, India, namianand006in@gmail.com
2Assistant Professor, Department of CSE, BRECW, Hyderabad, India, sumalatha.po@gmail.com

Abstract: Nowadays, much attention has been paid to intrusion detection system (IDS) which is closely linked to the safe use of network services. Intrusion detection system (IDS) is one of the emerging techniques for information security. Security mechanisms for an information system should be designed to prevent unauthorized access of system resources and data. Many intelligent learning techniques of machine learning are applied to the large volumes of data for the construction of an efficient intrusion detection system (IDS). Several machine-learning paradigms including neural networks, linear genetic programming (LGP), support vector machines (SVM), Bayesian networks, multivariate adaptive regression splines (MARS) fuzzy inference systems (FISs), etc. have been investigated for the design of IDS. This paper presents an overview of intrusion detection system and a hybrid technique for intrusion detection based on SVM algorithm and Genetic algorithm. SVM algorithm classifies the dataset into various categories to identify the normal/attacked packets where as genetic algorithm is used to generate a new data by applying mutation operation on the existing dataset to produce a new dataset. Thus this algorithm classifies KDD99 benchmark intrusion detection dataset to identify different types of attacks with high detection accuracy. The experimental result also shows that the accuracy of detecting attacks is fairly good.

Key words: Detection Accuracy, Genetic algorithms, Intrusion Detection System (IDS), Support vector machines (SVM).

INTRODUCTION

Information Security, intrusion detection is the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a resource. When Intrusion detection takes a preventive measure without direct human intervention, then it becomes an Intrusion-prevention system. Intrusion detection can be performed manually or automatically. Manual intrusion detection might take place by examining log files or other evidence for signs of intrusions, including network traffic. A system that performs automated intrusion detection is called an Intrusion Detection System (IDS). An IDS can be either host-based, if it monitors system calls or logs, or network-based if it monitors the flow of network packets. Modern IDSs are usually a combination of these two approaches [9]. Another important distinction is between systems that identify patterns of traffic or application data presumed to be malicious (misuse detection systems) and systems that compare activities against a 'normal' baseline (anomaly detection systems).

Intrusion Detection

Intrusion detection systems (IDS) are an essential part of the security infrastructure. They are used to detect, identify and stop intruders. The administrators can rely on them to find out successful attacks and prevent a future use of known exploits. IDS are also considered as a complementary solution to firewall technology as they recognize against the network that are missed by the firewall. Nevertheless, IDS are plagued with false positive alerts, letting security professionals to be overwhelmed by the analysis tasks [8]. Therefore, IDS employ several techniques in order to increase the detection probability of suspect threats while reducing the risk of false positives. While using pattern matching to detect intrusions, IDS users try to refine the attack signatures and limit the search to smaller traffic intervals. On the other hand, by using protocol analysis in the detection process, IDS rely on protocol specification in order to adequately analyze the traffic. In the intrusion detection field two different approaches can be observed: misuse detection and anomaly detection. The main idea behind misuse detection is to represent attacks in a form of a pattern or a signature in such a way that even variations of these attacks can be detected. Based on these signatures, this approach detects attacks through a large set of rules describing every known attack. The main disadvantage of the signature based approach is its difficulty for detecting unknown attacks. The main goal of the anomaly detection approach is to build a statistical model for describing normal traffic. Then, any deviation from this model can be considered an anomaly, and recognized as an attack. Notice that when this approach is used, it is theoretically possible to detect unknown attacks, although in some cases, this approach can lead to a high false attack rate. This ability to detect unknown attacks has been the cause of the increasing interest in developing new techniques to build models based on normal traffic behavior in the past years. An Anomaly-Based Intrusion Detection System is a system for detecting computer intrusions and misuse by monitoring system activity and classifying it as either normal or anomalous. The classification is based on heuristics or rules,
rather than patterns or signatures, and will detect any type of misuse that falls out of normal system operation. This is as opposed to signature based systems which can only detect attacks for which a signature has previously been created. In order to determine what attack traffic is, the system must be taught to recognize normal system activity. This can be accomplished in several ways, most often with artificial intelligence type techniques. Systems using neural networks have been used to great effect.

**Misuse-Based Detection.** The idea of misuse detection is to represent attacks in the form of a pattern or a signature so that the same attack can be detected and prevented in future. These systems can detect many or all known attack patterns, but they are of little use for detecting naive attack methods. The main issues of misuse detection is how to build signatures that include possible signatures of attacks build a signature that includes all possible variations of the pertinent attack to avoid false negatives, and how to build signatures that do not match non-intrusive activities to avoid false positive.

**Anomaly - Based Detection.** The idea here is that if we can establish a normal activity profile for a system, in theory we can flag all system states varying from the established profile as intrusion attempts. However, if the set of intrusive activities is not identical to the set of anomalous activities, the situation becomes more interesting instead of being exactly the same; we find few intrusive, though they are false positives. Actual intrusive activities that go undetected are called false negatives. This is a serious issue, and is far more serious than the problem of false positives. One of the main issues of anomaly detection systems is the selection of threshold levels so that neither of the above problems is unreasonably magnified. Anomaly detection is usually computationally expensive because of the overhead of keeping track of and possibly updating several system profiles. Recent proposed system Learning Rules for Anomaly Detection (LEARD) discovers relationships among attributes in order to model application protocols. Intrusion detection systems (IDS’s) are usually deployed along with other preventive security mechanisms, such as access control and authentication, as a second line of defense that protects information systems. There are several reasons that make intrusion detection a necessary part of the entire defense system. First, many traditional systems and applications were developed without security in mind. In other cases, systems and applications were developed to work in a different environment and may become vulnerable when deployed in the current environment. (For example, a system may be perfectly secure when it is isolated but become vulnerable when it is connected to the Internet). Intrusion detection provides a way to identify and thus allow responses to, attacks against these systems. Second, due to the limitations of information security and software engineering practice, computer systems and applications.

![Standard Intrusion Detection System](image)

**Table 1. Attack Classification**

<table>
<thead>
<tr>
<th>Denial of Service</th>
<th>Remote-to-Local</th>
<th>User to root</th>
<th>Probe</th>
</tr>
</thead>
<tbody>
<tr>
<td>Smurf</td>
<td>Phf</td>
<td>***</td>
<td>Satan</td>
</tr>
<tr>
<td>Samp</td>
<td>Rootkit</td>
<td>Nmap</td>
<td></td>
</tr>
<tr>
<td>Back</td>
<td>Load module</td>
<td>Butter overflow</td>
<td></td>
</tr>
<tr>
<td>Land</td>
<td>Multihop</td>
<td>Portswep</td>
<td></td>
</tr>
<tr>
<td>Neptune</td>
<td>Imap</td>
<td>Iphsweep</td>
<td></td>
</tr>
</tbody>
</table>

may have design flaws or bugs that could be used by an intruder to attack the systems or applications. As a result, certain preventive mechanisms (E.g., firewalls) may not be as effective as expected.

Intrusion detection complements these protective mechanisms to improve the system security. Moreover, even if the preventive security mechanisms can protect information systems successfully, it is still desirable to know what intrusions have happened or are happening, so that we can understand the security threats and risks and thus be better prepared for future attacks. Inspite of their importance, IDS’ are not replacements for preventive security mechanisms, such as access control and authentication. Indeed, IDSs themselves cannot provide sufficient protection for information systems. As an extreme example, if an attacker erases all the data in an information system, detecting the attacks cannot reduce the damage at all. Thus, IDSs should be deployed along with other preventive security mechanisms as a part of a comprehensive defense system.

Alternatively, IDSs may be classified into host-based IDSs, distributed IDSs, and network-based IDSs according to the sources of the audit information used by each IDS (Intrusion Detection Sys-tem). Host-based IDSs get audit data from host audit trails and usually aim at detecting attack [4][10][7][6].

**Input Data Description**

In the 1998 DARPA intrusion detection evaluation program, an environment was set up to acquire raw TCP/IP dump data
for a network by simulating a typical US Air Force LAN. The LAN was operated like a real environment, but being blasted with multiple attacks. For each TCP/IP connection, 41 various quantitative and qualitative features were extracted. Of this database a subset of 494021 data were used, of which 20% represent normal patterns. The four different categories of attack patterns are as follows.

- **Probing Attack**: It is a method of gathering information about a network of computers with an intention of circumventing its security controls.

- **Denial of Service Attack (DoS)**: It is a type of attack in which an attacker denies legitimate users access to machines or makes computing resources too busy to handle requests.

- **User to Root (U2R)**: In U2R the attacker first accesses the system with a normal user account by sniffing passwords or social engineering and then gains root access to the system by exploiting some vulnerability.

- **Remote to Local (R2L)**: R2L occurs when a user without an account has the ability to send packets to a machine gains local access as a user of that machine. Table 2 explains types of attacks.

Table 2. Four types of attacks in KDD’99 DataSet

<table>
<thead>
<tr>
<th>Attack Type</th>
<th>Attack Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probing</td>
<td>probe, map, portmap, banner</td>
</tr>
<tr>
<td>DoS</td>
<td>bad, land, neptune, ps, guess_passwd, showmem</td>
</tr>
<tr>
<td>U2R</td>
<td>rootkit, ped, load, dod, bufferover, d0w</td>
</tr>
<tr>
<td>R2L</td>
<td>ftp-, rlogin, spy, phf, guess_passwd, map, wanted:1, username:1, mailrelay</td>
</tr>
</tbody>
</table>

**Support vector machines (SVM)**

An SVM maps input (real-valued) feature vectors into a higher-dimensional feature space through some nonlinear mapping. SVMs are developed on the principle of structural risk minimization. Structural risk minimization seeks to find a hypothesis for which one can find lowest probability of error whereas the traditional learning techniques for pattern recognition are based on the minimization of the empirical risk, which attempt to optimize the performance of the learning set. Computing the hyper plane to separate the data points i.e. training an SVM leads to a quadratic optimization problem. SVM uses a linear separating hyper plane to create a classifier but all the problems cannot be separated linearly in the original input space. SVM uses a feature called kernel to solve this problem.

The Kernel transforms linear algorithms into nonlinear ones via a map into feature spaces. There are many kernel functions; including polynomial, radial basis functions, two layer sigmoid neural nets etc.

![Fig. 2. Separating Hyperplane with SVM](image)

The user may provide one of these functions at the time of training the classifier, which selects support vectors along the surface of this function. SVMs classify data by using these support vectors, which are members of the set of training inputs that outline a hyper plane in feature space consider each data instance to be an n-dimensional vector of attribute values. In an SVM, a data point is viewed as a vector in the d-dimensional feature space. Assume that all data points belong to either class A or class B.

**Genetic Algorithm**

Genetic algorithm is one of the components of evolutionary computation technique. A simple genetic algorithm may consist of a population generator and a selector, a fitness estimator and three genetic operators namely selection, mutation and crossover. The mutation operator inverts randomly chosen bits with a certain probability. The crossover operator combines parts of the species of two individuals, generates two new off springs, which are used to replace low fitness individuals in the population. After a certain number of generations, the search process will be terminated. A genetic algorithm (or GA for short) is a programming technique that mimics biological evolution as a problem-solving strategy. Given a specific problem to solve, the input to the GA is a set of potential solutions to that problem, encoded in some fashion, and a metric called a fitness function that allows each candidate to be quantitatively evaluated. These candidates may be solutions already known to work, with the aim of the GA being to improve them, but more often they are generated at random.

The GA then evaluates each candidate according to the fitness function. In a pool of randomly generated candidates, we choose promising candidates toward solving the problem. These promising candidates are kept and allowed to reproduce. Multiple copies are made of them, but the copies may not perfect; random changes are introduced during the copying process. These digital off spring then go on to the next generation, forming a new pool of candidate solutions, and are subjected to a second round of fitness evaluation. Those candidate solutions which were worsened, or made no better, by the changes to their code are again deleted; but again, purely by chance, the random variations introduced into the population may have improved some individuals, making them into better, more complete or more efficient solutions to the problem at hand. Again these winning individuals are selected and copied over into the next
generation with random changes, and the process repeats. The expectation is that the average fitness of the population will increase each round, and so by repeating this process for hundreds or thousands of rounds, very good solutions to the problem can be discovered.\textsuperscript{[3],[5]}

Methods of Change. Once selection has chosen fit individuals, they must be randomly altered in hopes of improving their fitness for the next generation. There are two basic strategies to accomplish this, they are:

1. **Mutation**: By applying random changes to a single individual in the current generation to create a child.

2. **Crossover**: By selecting vector entries, or genes, from a pair of individuals in the current generation and combines them to form a child.

\textbf{Method Applied in this Paper}. The classic example of a mutation operator involves a probability that an arbitrary bit in a genetic sequence will be changed from its original state. A common method of implementing the mutation operator involves generating a random variable for each bit in a sequence. This random variable tells whether or not a particular bit will be modified. This mutation procedure, based on the biological point mutation, is called single point mutation.

\textbf{Initialization}. Initially many individual solutions\textsuperscript{[3]} are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions.

\textbf{Selection}. During each successive generation, a proportion of the existing population is selected to breed a new generation. Individual solutions are selected through a fitness-based process, where fitter solutions (as measured by a fitness function) are typically more likely to be selected. Certain selection methods rate the fitness of each solution and preferentially select the best solutions. Fitness scaling converts the raw fitness scores that are returned by the fitness function to values in a range that is suitable for the selection function. The selection function uses the scaled fitness values to select the parents of the next generation. The selection function assigns a higher probability of selection to individuals with higher scaled values.\textsuperscript{[12]} Initially many individual solutions are randomly generated to form an initial population. The population size depends on the nature of the problem, but typically contains several hundreds or thousands of possible solutions.

\textbf{ARCHITECTURE}

\textbf{Components in the Architecture}

\textbf{KDD Dataset}: This is the initial dataset obtained from the MIT Lincoln labs.

\textbf{Pre-Processor}: The pre-processor processes the large chunks of data and we obtain the minimized dataset with particular required no. of attributes.

\textbf{SVM Algorithm Processor}: This processor applies the SVM Algorithm on the input dataset to classify it into various categories.

\textbf{Classified Dataset}: Now we obtain the classified dataset for which we compute the false positive and false negative rates.

\textbf{Genetic Algorithm Processor}: This takes the classified dataset as input and builds a new dataset using the Genetic Algorithm by observing the different variations in the dataset.

\textbf{New Dataset}: After the implementation of Genetic Algorithm we have the new dataset for which we again compute the false positive and false negative rates.

\textbf{Comparative Study and Final Results}: We show a comparative study of the false positive and false negative rates of the classified dataset and the newly obtained dataset and show the final results.\[7\]\[3\]

\textbf{IMPLEMENTATION & EXPERIMENTAL RESULTS}

This section evaluates the performance of the individual SVM and proposed hybrid SVM-GP for detection. In the individual SVM and proposed hybrid SVM-GP algorithms, the k-fold method is used to evaluate the accuracy of
classification, and output the best test accuracy and decision rules. This study set k as 10; that is, the data was divided into 10 portions. Nine portions of data are retrieved as training data and the other one is used for testing data. In experiments, the parameter C and of SVM varies from 0.01 to 50,000.

<table>
<thead>
<tr>
<th>Class</th>
<th>Hybrid intelligent system</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>True classification (TP - FN)</td>
</tr>
<tr>
<td>Normal</td>
<td>27619</td>
</tr>
<tr>
<td>Probe</td>
<td>5045</td>
</tr>
<tr>
<td>DOS</td>
<td>379508</td>
</tr>
<tr>
<td>U2R</td>
<td>35</td>
</tr>
<tr>
<td>R2L</td>
<td>47</td>
</tr>
<tr>
<td>Average precision</td>
<td>99.96412139</td>
</tr>
</tbody>
</table>

Table 3. The Classification Precision for Hybrid IDS

A set of data is selected to train the process and the algorithm. Then we have another set called the test set with which the SVM Algorithm is implemented. Thus after classifying the dataset into attack and normal packets using SVM Algorithm, we calculate the false positive, false negative rates and the detection accuracy. We then apply Genetic algorithm to obtain a new generation of dataset by selecting the required attributes from the already existing dataset. Here, we use mutation as the reproduction operator. Table 3 shows classification precision for hybrid IDS. Now we use this dataset to again implement the SVM Algorithm and classify the dataset into attack and normal packets. Also we calculate the false positive, false negative rates and the detection accuracy.

Finally we show a comparative analysis of the false positive, false negative rates and the detection accuracy.

CONCLUSION

In this research, we have investigated some new techniques for intrusion detection and evaluated their performance based on the benchmark KDD Cup 99 Intrusion data. We have explored SVM and GP as intrusion detection models. Next we designed a hybrid SVM-GP model and an ensemble approach with SVM, GP and SVM-GP models as base classifiers. Empirical results reveal that GP gives better or equal accuracy for Normal, Probe, U2R and R2L classes. The hybrid SVM-GP approach improves or delivers equal performance for all the classes when compared to a direct SVM approach. The Ensemble approach gave the best performance for Probe and R2L classes. The ensemble approach gave 100% accuracy for Probe class, and this suggests that if proper base classifiers are chosen 100% accuracy might be possible for other classes too. Finally, we propose a hierarchical intelligent IDS model to make optimum use of the best performances delivered by the individual base classifiers and the ensemble approach.

ACKNOWLEDGMENT

It is not only customary but necessary for a researcher to mention his/her indebtedness to those who had helped in carrying out and enhance the research work. Right from the moment of selecting the subject of research till the writing of the conclusions, a student has to turn to guide who helps in carrying the student by hand along the road to the destination. I am, therefore, grateful to my guide, A.G. Joshi, who helped me a long to complete this project.

I pay my deep regards to my caring Husband Mr. Pratyush Anand, my Parents and my loving Friends for their support and wishes which made this tedious work easy and successful.

Finally, I would like to extend my thanks to all those who have contributed, directly or indirectly to make this project successful.

REFERENCES