Network and Complex Systems
ISSN 2224-610X (Paper) ISSN 2225-0603 (Online)
Vol 2, No.4, 2012

Soft Computing and Artificial Intelligence Techniques for Intrusion Detection System

V. Bapuji[1*] R. Naveen Kumar[2] Dr. A. Govardhan[3] Prof. S.S.V.N. Sarma[4]

1. Department of Informatics, Kakatiya University, Warangal, India
2. Department of Informatics, Kakatiya Univeristy, Warangal, India
3. Department of CSE, Jawaharlal Technological University (JNTUH), Hyderabad, India
4. Department of CSE, Vaagdevi College of Engineering, Bollikunta, Warangal (A.P), India

* E-mail of the corresponding author: bapuji.vala@gmail.com

Abstract

The rapid development of computer networks and mostly of the Internet has created many stability and security problems such as intrusions on computer and network systems. Further the dependency of companies and government agencies is increasing on their computer networks and the significance of protecting these systems from attacks is serious because a single intrusion can cause a heavy loss or the consistency of network becomes insecure. During recent years number of intrusions has dramatically increased. Therefore it is very important to prevent such intrusions. The hindrance of such intrusions is entirely dependent on their detection that is a key part of any security tool such as Intrusion Detection System (IDS), Intrusion Prevention System (IPS), Adaptive Security Alliance (ASA), checkpoints and firewalls. Hence accurate detection of network attack is imperative. A variety of intrusion detection approaches are available but the main problem is their performance, which can be enhanced by increasing the detection rates and reducing false positives.

Keywords: IDS, Soft Computing, ANN, Genetic Algorithm

1. Introduction

An Intrusion Detection System (IDS) itself can be defined as the tools, methods, and resources to help identify, assess and report unauthorized or unapproved network activity. The intrusion detection part of the name is a bit of a misnomer as an IDS does not actually detect intrusions. It detects activity in traffic that may or may not be an intrusion. To be more specific an IDS is a specialized tool that knows how to parse and interpret network traffic and/or host activities. This data can range from network packet analysis to the contents of log files from routers, firewalls, and servers, local system logs and access calls, network flow data, and more. Furthermore an IDS often stores a database of known attack signatures and can compare patterns of activity, traffic, or behavior it sees in the data it's monitoring against those signatures to recognize when a close match between a signature and current or recent behavior occurs. At that point the IDS can issue alarms or alerts take various kinds of automated actions ranging from shutting down Internet links or specific servers to launching back traces and make other active attempts to identify attackers and collect evidence of their nefarious activities.

2. Soft Computing and Artificial Intelligence for Intrusion Detection System

Since most of the intrusions can be located by examining patterns of user activities and audit records many IDSs have been built by utilizing the recognized attack and misuse patterns. IDSs are classified based on their functionality as misuse/signature detectors and anomaly detectors. Misuse/signature recognition systems use well known attack patterns as the basis for detection. Anomaly detection systems make use user profiles as the basis for detection of any deviation from the normal user behavior is considered an intrusion.

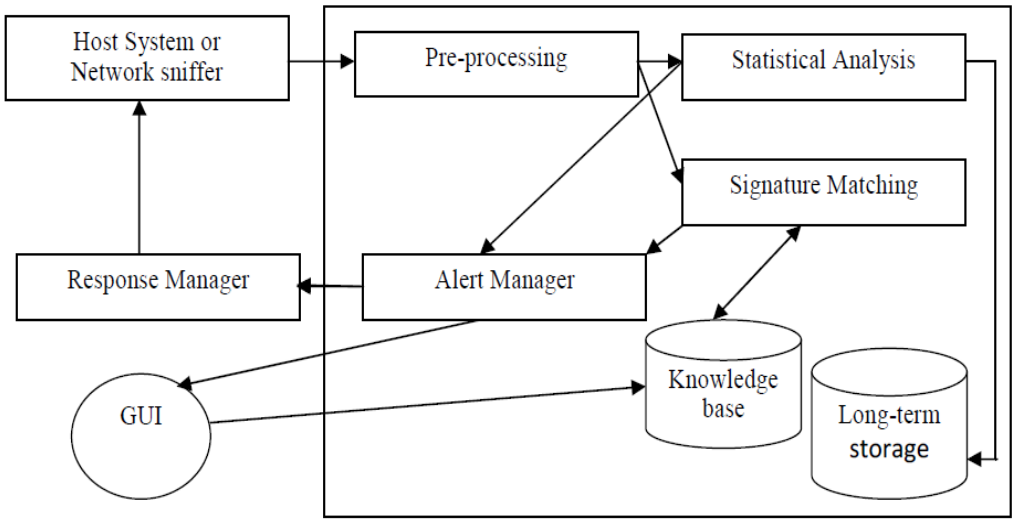

Figure 1. The Standard Intrusion Detection System.

For increasing IDS's "performance" the computer security researchers try to apply soft computing methods to IDS. Soft computing is the fusion of methodologies that were designed to model and enable solutions to real world problems, which are not modeled or too difficult to model mathematically. Soft computing is a general term for describing a set of optimization and processing techniques that exploit the tolerance for imprecision, uncertainty, partial truth and approximation to achieve robustness and low solution cost. An intelligent adaptable and cost effective tool that is capable of (mostly) real time intrusion detection is the goal of the researchers in IDSs. Various Artificial Intelligence (AI) techniques have been utilized to automate the intrusion detection process to reduce human intervention. And generally the techniques which implemented to IDS based on machine learning. Generally machine learning methods that used in IDS implementation are Artificial Neural Network (ANN) and Genetic Algorithms (GA) with addition techniques to improve that approach are Data Mining, Fuzzy Logic, and Probabilistic Reasoning.

3. Artificial Neural Networks

Artificial Neural Networks (ANN) often just called a "Neural Networks" (NN), are sets of mathematical models based on biological neural networks (nerve cells) assigning different weights to connections between elements within the neural network similarly to how electrical potentials for neurons are built up at synaptic junctions based on their frequency of firing. The more frequently a neighboring neuron fires the more electrical potential there is at the synapses of the neurons that react to this pattern. In neural networks elements that receive inputs from neighboring elements receive higher weights. It consists of an interconnected group of artificial neurons and processes information using a connectionist approach to computation. In most cases an ANN is an adaptive system that changes its structure based on external or internal information that flows through the network during the learning phase. In more practical terms neural networks are non-linear statistical data modeling tools. They can be used to model complex relationships between inputs and outputs or to find patterns in data.

Although complicated and still somewhat mysterious the neural networks approach can be applied to a wide range of pattern recognition problems intrusion detection included. The idea behind the application of soft computing techniques and particularly ANN in implementing IDSs is to include an intelligent agent in the system that is capable of disclosing the latent patterns in abnormal and normal connection audit records and to generalize the patterns to new (and slightly different) connection records of the same class. And the beauty of neural networks in intrusion detection is that no signatures or even rules are needed. To simply start feeding input data concerning network or host based

events to a neural network and it does the rest. Neural networks are therefore well suited to picking up new patterns of attacks readily although some learning time is required. The neural networks approach has been around for a long time and if anything it is likely to become more widely used and relied on in intrusion detection in the future as reliance on signatures diminishes.

Different types of Artificial Neural Network (ANN) exist for various purposes such as classification, prediction, clustering, association, etc. A feedforward ANN was the first and arguably simplest type of artificial neural network devised. In this network the information moves in only one direction forward from the input nodes through the hidden nodes (if any) and to the output nodes. There are no cycles or loops in the network) with the back propagation learning algorithm is commonly used for classification problems. Cyber attack detection can be considered a classification problem in that computer and network data is classified into attack or normal use. One advantage of a feedforward ANN for classification problems lies in its ability to learn a sophisticated nonlinear input-output function. The ANN learns signature patterns of cyber attacks and normal use activities from the training data and uses those signature patterns to classify activities in the testing data into attack or normal use.

A feedforward ANN has one or more hidden layers of processing units and one output layer of processing units. Processing units are connected between layers but not within a layer. Figure 2 shows a fully connected feedforward ANN with three inputs, one hidden layer of 4 processing units, and one output layer of 2 processing units. In Figure 2 each input is connected to each hidden unit and each hidden unit is connected to each output unit. Each connection has a weight value associated with it.

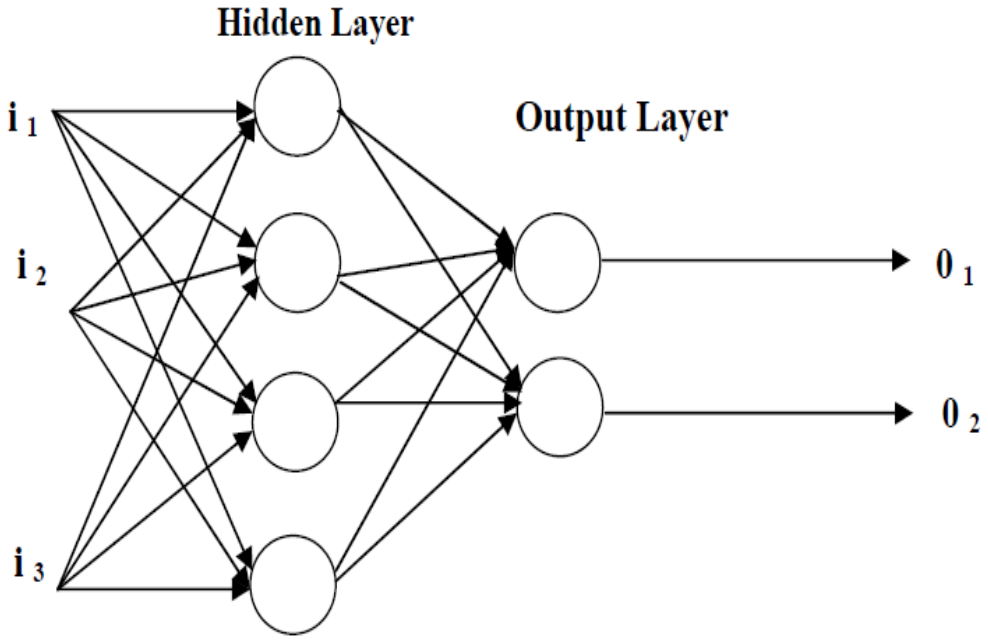

Figure 2. Example of a two-layered feedforward fully connected ANN.

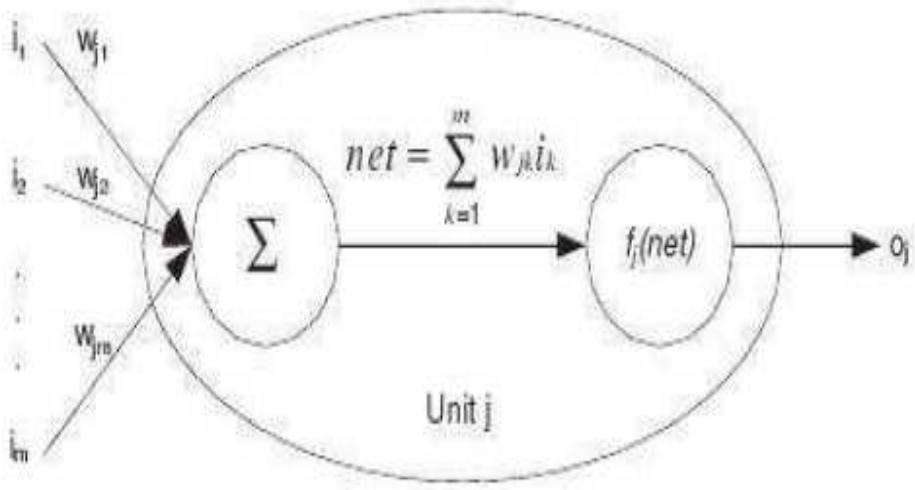

Figure 3.Structure of a single processing unit.

As shown in figure 3, the output value of a processing unit is computed from the inputs to the unit as follows:

$$O_j = f_j(net_j) \qquad \text{(Figure 2)}$$

$$net\ j = \sum_{k=1}^{m} wjk\ ik \qquad \text{(Figure 3)}$$

Where Oj is the output value of unit j, fj is the activation function of unit j, ik is the k th input to unit j, Wjk is the weight of the connection from the k th input to unit j. A sigmoid function is an example of the activation function with the following form which produces an output value in (0, ~):

$$f(net_j) = \frac{1}{1 + e^{-net_j}}. \qquad \text{(activation function)}$$

The activation function can be the same for all the units in the ANN or can be different for different units. Generally the ANN aims at approximating the function between the inputs and the outputs. The connection weights of an ANN are typically initialized to random values. Using the initial connection weights the ANN may not produce the target outputs for the given inputs. Hence the initial connection weights need to be adjusted by using a training data set of input-output pairs to learn the input-output function.

Some IDS designers exploit ANN as a pattern recognition technique. Pattern recognition can be implemented by using a feedforward neural network that has been trained accordingly. During training the neural network parameters are optimized to associate outputs (each output represents a class of computer network connections, like normal and attack) with corresponding input patterns (every input pattern is represented by a feature vector extracted from the characteristics of the network connection record). When the neural network is used it identifies the input pattern and tries to output the corresponding class. When a connection record that has no output associated with it is given as an input, the neural network gives the output that corresponds to a taught input pattern that is least different from the given pattern.

The most commonly reported application of neural networks in IDSs is to train the neural net on a sequence of information units each of which may be an audit record or a sequence of commands. The input to the net consists of the current command and the past *w* commands (*w* is the size of window of commands under examination). Once the net is trained on a set of representative command sequences of a user it constitutes (learns) the profile of the user and when put in action, it can discover the variance of the user from its profile. Usually recurrent neural networks are used for this purpose.

4. Genetic Algorithm

The Genetic algorithm (GA) as a search technique used in computing to find exact or approximate solutions to optimization and search problems. Genetic algorithms are categorized as global search heuristics. Genetic algorithms are a particular class of evolutionary algorithms (also known as evolutionary computation) that use techniques inspired by evolutionary biology such as inheritance, mutation, selection, and crossover (also called recombination).

The process of a genetic algorithm usually begins with a randomly selected population of chromosomes. These chromosomes are representations of the problem to be solved. According to the attributes of the problem, different positions of each chromosome are encoded as bits, characters, or numbers. These positions are sometimes referred to as genes and are changed randomly within a range during evolution. The set of chromosomes during a stage of evolution are called a population. An evaluation function is used to calculate the "goodness" of each chromosome. During evaluation the two basic operators such as crossover and mutation are used to simulate the natural reproduction and mutation of species. The selection of chromosomes for survival and combination is biased towards the fittest chromosomes.

A typical genetic algorithm requires two things to be defined:

1. A genetic representation of the solution domain.
2. A fitness function to evaluate the solution domain.

A standard representation of the solution is as an array of bits. Arrays of other types and structures can be used in essentially the same way. The main property that makes these genetic representations convenient is that their parts are easily aligned due to their fixed size that facilitates simple crossover operation. Variable length representations may also be used, but crossover implementation is more complex in this case. Tree like representations are explored in Genetic programming and graph form representations are explored in Evolutionary programming.

The fitness function is defined over the genetic representation and measures the quality of the represented solution. The fitness function is always problem dependent. For instance in the knapsack problem we want to maximize the total value of objects that we can put in a knapsack of some fixed capacity. A representation of a solution might be an array of bits where each bit represents a different object and the value of the bit (0 or 1) represents whether or not the object is in the knapsack. Not every such representation is valid, as the size of objects may exceed the capacity of the knapsack. The fitness of the solution is the sum of values of all objects in the knapsack if the representation is valid or 0 otherwise. In some problems it is hard or even impossible to define the fitness expression in these cases interactive genetic algorithms are used.

Once the genetic representation and the fitness function defined GA proceeds to initialize a population of solutions randomly then improve it through repetitive application of mutation, crossover, inversion and selection operators. The GA's flowchart (Figure 4) for its iterative process like this:

A basic genetic algorithm Example of a simple GA's pseudo code:

```
1: InitPopulation(P)
2: Fitness(P)
3: while MaxGenerationNotReached do
4:    for i = 0 to xfactor do
5:       p1 = Selection(P)
6:       p2 = Selection(P)
```

```
7:        (o1, o2) = crossover(p1, p2)
8:        crowding(p1, p2, o1, o2)
9:     end for
10:    for i = 0 to dfactor do
11:        p = Selection(P)
12:        Dropping(p)
13:    end for
14:    for i = 0 to mfactor do
15:        p = Selection(P)
16:        Mutation(p)
17:    end for
18:    Fitness(P)
19: end while
20: SelectBestIndividual(P)
```

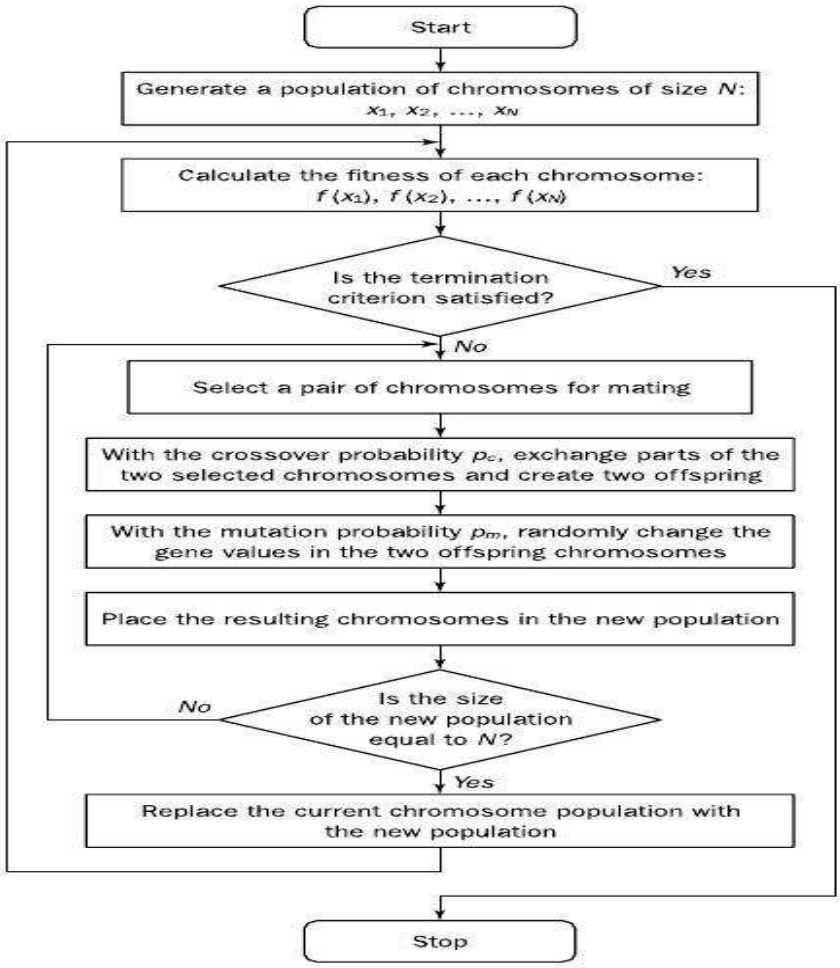

Figure 4. Genetic Algorithm Flowchart for Iterative Process

The GA is used for problem solving and the three factors will have to impact on the effectiveness of the algorithm they are: 1) The selection of fitness function 2) The representation of individuals and 3) The values of the GA parameters.

The Genetic algorithms can be used to evolve simple rules for network traffic. These rules are used to differentiate normal network connections from anomalous connections. These anomalous connections refer to events with probability of intrusions. The rules stored in the rule base are usually in the following form:

if { condition } then { act }

The condition usually refers to a match between current network connection and the rules in 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 usually refers to an action defined by the security policies within an organization such as reporting an alert to the system administrator stopping the connection, logging a message into system audit files or all of the above.

The final goal of applying GA is to generate rules that match only the anomalous connections. These rules are tested on historical connections and are used to filter new connections to find suspicious network traffic.

5. Conclusions

The implementation of Soft Computing and Artificial Intelligence methods/techniques are used widely in Intrusion Detection System are gaining its ability to learn and evolve which makes them more accurate and efficient in facing the enormous number of unpredictable attacks. We investigate with Soft Computing paradigm of evolutionary computation techniques for synthesizing intrusion detection programs on Mobile Ad hoc Networks. We evolve the programs to detect such as Ad hoc Flooding, Route Disruption and Dropping Attacks against Mobile Ad hoc Networks. Since prevention techniques cannot be sufficient and new intrusions continuously emerges, IDS is an indispensable part of a security system. IDS detect possible violations of a security policy by monitoring system activities and response. If we detect an attack once it comes in to the network, a response can be initiated to prevent or minimize damage of the system. The two major techniques for machine learning were highlighted, with the use of Genetic Algorithm and Artificial Neural Network providing intrusion system with extra intelligence. Since

References

Nong Ye Secure Computer and Network Systems: Modeling, Analysis and Design- Wiley Publishers age 258-259.

Andrew G. Barto and Michael I. Jordan. Gradient following without back-propagation in layered networks. In Proceedings of the IEEE First Annual Conference on Neural Networks, pages II629–II636. IEEE Publishing Services, New York, 1987.

Kenji Doya. Metalearning and neuromodulation. Neural Networks, 15:495–506, 2002.Kevin Gurney. Simon Haykin. Neural Networks: A Comprehensive Foundation. Prentice Hall, 2 edition, 1999.

Anders Holst. The Use of a Bayesian Neural Network Model for Classification Tasks. PhD thesis, Studies of Artificial Neural Systems, Nada, KTH, September 1997.

Pietro Mazzoni, Richard A. Andersen, and Michael I. Jordan. A more biologically plausible learning rule for neural networks. In Proceedings of the National Academy of Sciences, volume 88, pages 4433–4437, May 1991.

Fausett, Laurene: Fundamentals of Neural Networks: Architectures, Algorithms, and Applications. Prentice Hall, NJ, USA 1994.

Dote, Yasuhiko; S.Taniguchi, and T. Nakane:Intelligent Control Using Soft Computing" Proceedings of the 5th Online World Conference on Soft Computing in Industrial Applications (WSC5). 2000.

Kacpzyk, Janusz (Editor):Advances in Soft Computing. Springer-Verlag, Heidelberg, Germany, 2001.

V. Bapuji is currently pursuing Ph.D. degree in Computer Science at Kakatiya University Warangal (A.P), India. He has 12 years of teaching experience in Post Graduate level. His research interests include Mobile Ad Hoc Networks Security, Soft Computing, and Design efficient accurate reliable low cost IDS. His work focuses on the security and fault tolerance of routing protocols, with an emphasis on solutions to support Mobile Ad hoc Networking. E-mail: bapuji.vala@gmail.com

R. Naveen Kumar is a Ph.D. candidate, Department of Informatics Kakatiya University Warangal (A.P), India. He has 14 years of teaching experience. His interests include: Mobile Ad Hoc Networks Security Vulnerabilities and Key management, Cross Layer Design. E-mail: naveensmitha@gmail.com

Dr. A. Govardhan is a Professor, Department of computer science and Engineering at Jawaharlal Nehru Technological University (JNTUH) Hyderabad (A.P), India. His research interests are in the area of Mobile Computing, Data management in wireless mobile environments and Data Mining. He has served on several program committees of conferences in the area of mobile computing, data management and sensor networks. He is a Senior Member of IEEE, ACM and various organizations. He has organized several workshops, delivered numerous tutorials at major IEEE and ACM conferences, and serves as editor of several journals and magazines. E-mail: govardhan_cse@yahoo.co.in

S.S.V.N. Sarma was a Professor (From 1975 to 2010). He has 40 years of experience and worked as Head, Department of Informatics, Chairman Board of Studies, and Dean Faculty of Science, Kakatiya University, Warangal (A.P) India. Presently he is working as a Dean, Academic Affairs at Vaagdevi College of Engineering, Bollikunta, Warangal (A.P), India. His current research interests include Distributed systems, Mobile computing, Computer networks, Computer security, and Performance evaluation. He has published over 120 refereed articles in these areas. He has organized several workshops, delivered numerous tutorials at major IEEE and ACM conferences, and serves as Editor of several International/National journals and magazines. He is a fellow member of professional organizations. E-mail: ssvn.sarma@gmail.com