Use of Genetic Algorithm in Network Security

L.M.R.J Lobo
Professor, Department of Computer Science & Engg.
Walchand Institute of Technology, Solapur, India

Suhas B. Chavan
MECSE (SEM IV), Department of Computer Science & Engg. Walchand Institute of Technology, Solapur, India

ABSTRACT
After overcoming some drawbacks from an improved genetic feedback algorithm based network security policy framework. A motivation was experienced for the need of a strong network security policy framework. In this paper a strong network model for security function is presented. A gene for a network packet is defined through fitness function and a method to calculate fitness function is explained. The basic attacks encountered can be categorized as buffer overflow, array index out of bound, etc. A stress is given on passive attack, active attack, its types and brute force attack. An analysis on recent attacks and security is provided. Finally the best policy using a comparator is found. The main aim of this paper is threat detection, time optimization, performance increase in terms of accuracy and policy automation. For experimenting real data set from the internet and local network is used. Jpcap, winpcap, Colasoft Capsa 7 open source software are used for implementation. 73% efficiency is achieved because we add 3 networking terms such as payload, checksum, and sequence number. A client server environment is made use of.

General Terms
Genetic algorithm, Network security, Crossover technique.

Keywords
Genetic algorithm, Network security policy framework, Fitness function.

1. INTRODUCTION
In a client server communication, attackers need only little information for doing fraudulent transaction. A genetic algorithm is an example of evolutionary computing models and optimization-type algorithm. Chromosomes, which are DNA strings, provide the abstract model for a living organism. Subsections of the chromosomes, genes, are used to define different traits of the individuals. During Reproduction genes of the parents are combined to form genes for the child.

Given an alphabet A, an individual or chromosome is a string I= I_1, I_2, I_3 …… In Where I € A. Each character in the string, Ij, is called a gene. A population, P, is a set of individuals.

Although individuals are often represented as bit strings, any encoding is possible. One of the most important components of a genetic algorithm is determining how to select an individual. A fitness function f is used to determine the best individuals in a population. This is then used in the selection process to choose parents.

A policy based system has three basic steps in which it works i.e. creation, assignment and execution of policy. These depend on the network event type and requirement in terms of security.

2. RELATED WORK
In improved genetic feedback algorithm based network security policy framework , the security model was much more simplified and implementable. Their future work included creating the network security framework and testing it on real data sets [1].

In "An Evolutional network security policy framework based on gene-feedback algorithm" paper, based on the historical security events, using genetic algorithm, they generated a rule base. When a new network event encountered, the analyzer judged whether the event was secure or not according to the rule base, and the policy system gave a policy decision too. Obviously, these two results differed. So the policies could be automatically adjusted, referring the genetic calculated results [2].

In a new interactive genetic algorithm (IGA) framework incorporating relevant feedback (RF) in which human evolution is regarded as not only the fitness function of the GA, but also the relevant score to instruct interactive machine learning. The two mapped functions between the low-level parameter space and the high-level users' psychological space built during interactions. The effectiveness of our approach was first evaluated through simulation tests using two benchmark functions. The experimental result showed that the convergence speed of the proposal was faster than that of normal IGA. Then the approach was applied to retrieve images with emotion semantics queries [3].

In "Designing rule base for genetic feedback algorithm based network security policy framework using state machine" paper, A genetic algorithm based policy management system judged the validity of network events according to the rules defined in the rule base. These rules were either IP address or some other parameters, such as a port number etc. This paper discussed the design and benefits of rule base which is based on finite state machine. Since a new networking event came, the process of judges the event should be less time consuming. This paper introduced how FSM could be used for representing and managing the rule base of a genetic feedback algorithm based network security framework in an efficient way [4].

In "A Policy-based Management System with Automatic Policy Selection and Creation Capabilities by using a Singular Value Decomposing Technique" paper, it described a novel method by which policies could be selected or created automatically based on events observed and knowledge learned. This new approach treats the observed event-policy relationship represented by an event-policy matrix as a statistical problem. Using Singular value decomposition (SVD) technique, implicit higher order correlations among

policies and their associated events were used to estimate the selection or creation of recommended policies based on events found in the observed event set [5].

In "Storing Scheme for State Machine Based Rule Base of Genetic Feedback Algorithm Based Network Security Policy Framework Depending on Memory Consumption" paper. Three types of storing schemes are discussed namely "Time Efficient Storing" which gave constant search time, second is "Trade off Storing" in this scheme how a trade-off could be done between space and time complexities to get a system according to availability of resources. The time complexity of this scheme would be in the range of 2n. Where n=2, 3…10 and the value of n is indirectly proportional to the space used. Whenever a "*" was encountered in the rule base it was stored in a simple linked list. In the data field "*" was stored and the link field would point to the next table or next link. For example whenever the system has to check whether the rule was present or not, it will see the start link and go to the index location 'a' (i.e. Considering IPV4 addresses in the format a.b.c.d) and if the location at index 'a' points to a table then it will go to that table. If index "d" points to the "END" link then the system terminated with successful search. In Trade off method they have stored the entries in the array of link list and simple links called spares with each link consisting of three fields one is Boolean data field and two pointer fields.

The size of each array would be 2n, where n = 0 to 8. Worst case time complexity in the case of trade off sharing could be calculated by using simple formula i.e. Worst case time complexity = (256/2n) *p Where, n = 0 to 8, depending on the size of the array, and P is a number of parts in IP address i.e. p = 4 If IP version 4 is used and p=16 is IP version 6 is used. In Space efficient storing the link have three parts i.e. value part which stored value between 0-255 and two pointer fields one pointer pointing the next value at the same level and the second pointer pointing the values at the next level. Here same level means the same octet of the address and next level means the next octet of the address. I.e. if the system is in 'a' position of the IP address then it would go to the 'b' octet of the address.

Selection of storing scheme could be done on the basis of number of entries in the rule base i.e. if the number of entry was more than half of the total possible IP addresses then time efficient storing should be given preference [6].

In An Artificial intelligence perspective on autonomic computing policies, three policies were described i.e. Action policy, goal policy, Utility function policy. In action policy should be taken whenever the system is in a given current state. Typically this took the form of IF (Condition) THEN (Action), where Condition specified either a specific state or a set of possible states that all satisfy the given Condition. An implementing utility function required optimization algorithms. Because utility functions were a function of the states, it might appear easy and natural to use optimization to directly identify the most desirable state as a Goal from which actions could be derived via planning and/or modeling [7].

Reinforcement learning was the problem faced by an agent that learnt behavior through trial-and-error interactions with a dynamic environment. In this literature they have discussed central issues of reinforcement learning, including trading off exploration and exploitation, establishing the foundations of the field via Markov decision theory, learning from delayed reinforcement, constructing empirical models to accelerate learning, making use of generalization and hierarchy, and coping with hidden state. It concluded with a survey of some implemented systems and an assessment of the practical utility of current methods for reinforcement learning [8].

In Improved Algorithms for Finding Gene Teams and Constructing Gene Team Tree paper, a discusseion on gene team was covered. A set of gene that appear in two or more species, possibly in a different order yet with the distance between adjacent genes in the team for each chromosome always no more than a certain threshold \delta. A gene team tree was a succinct way to represent all gene teams for every possible value of \delta. In this paper, improved algorithms are presented for the problem of finding the gene teams of two chromosomes and the problem of constructed a gene team tree of two chromosomes [9].

In RFC 2753 a framework for policy-based admission control was given. Networking terms such as administrative domain, network element or node, policy, policy decision point, policy enforcement point has been discussed [10].

In Performance Evaluation of a Genetic Algorithm Based Approach to Network Intrusion Detection System. They provided an intrusion detection system (IDS), genetic algorithm (GA) applied to a network intrusion detection system. Parameters and evaluation process for GA was discussed in detail and implemented**.** This approach used in information theory to filter the traffic data and thus reduce the complexity. They used a linear structure rule to classify the network behaviors into normal and abnormal behaviors. [11].
In Network Routing Protocol using Genetic Algorithms. This paper developed a genetic algorithm to solve a network routing protocol problem. The algorithm has to find the shortest path between the source and destination nodes. In the literature, the routing problem was solved using search graph techniques to find the shortest path. Dijkstra's algorithm was one of the popular technique to solve this problem. The developed genetic algorithm is compared with Dijkstra's algorithm to solve routing problem. Simulation results carried out for both algorithms using MATLAB. The results affirmed the potential of the proposed genetic algorithm. The obtained performance was similar to Dijkstra's algorithm. [12].
In network security policy framework and analysis paper. They have used signature rules for infected packet analysis. They have tested it on real data sets available on the internet with some automated software. As future expansion in this work, latest attack prevention can be dealt with [13].

In An Implementation of intrusion detection system using genetic algorithm. Secured data communication over the internet and any other network was always under threat of intrusions and misuses. So Intrusion Detection Systems have become a needful component in terms of computer and network security. They presented an Intrusion Detection System (IDS), through a genetic algorithm (GA). This detected efficiently and applied to various types of network intrusions. Parameters and evaluation processes for GA were discussed in details and implemented. This approach used evolution theory to inform evolution in order to filter the traffic data and thus reduce the complexity. To implement and measure the performance of our system we used the KDD99 benchmark datasets and obtained reasonable detection rate. [14].

3. GENETIC ALGORITHM

A pseudocode may be used to feature the steps taken in a Genetic Algorithm approach.

Input:

P //Initial population

Output:

P' //Improved population

Genetic algorithm:

repeat

N=|P|

P'=Ø;

repeat

i_1, i_2 = select (P);

o_1, o_2 = cross (i_1 , i_2);

o_1 = mutate (o_1);

o_2 = mutate (o_2);

P'= P'U {o_1, o_2 };

until |P'| = N;

P = P';

until termination criteria satisfied;

Algorithm.1: Working of Genetic Algorithm

Initially a population of individuals, P is created. Although different approaches can be used to perform this step, they typically are generated randomly. From this population, a new population, P', of the same size is created. The algorithm repeatedly selects individuals from whom create new ones. These parents i_1, i_2 are then used to produce two offspring, o_1, o_2 using a crossover process. Then mutants may be generated.

The process continues until the new population satisfies the termination condition. In genetic algorithms, reproduction is defined by precise algorithms that indicate how to combine the given set of individuals to produce new ones. These are called crossover algorithm. Given two individual parents from the population, the crossover technique generates new individuals (offspring or children) by switching subsequences of the strings. Table 1 illustrates the process of crossover.

3.1 SINGLE CROSSOVER

Table 1: Reproduction (Crossover Algorithms)

000 | 000	000 | 111
111 | 111	111 | 000
Parents	Children

3.2 MULTIPLE CROSSOVER

Table 2: Reproduction (Crossover Algorithms)

000 | 000 | 00	000 | 111 | 00
111 | 111 | 11	111 | 000 | 11
Parents	Children

The locations indicating the crossover points are shown in the Table 1 with the vertical lines. In Table 1 a crossover is achieved by interchanging the last three bits of the two strings. In Table 2 the center three bits are interchanged.

As in nature however, mutations sometimes appear, and these may be present in genetic algorithms. The mutation operation randomly changes characters in the offspring. A very small probability of mutation is set to determine whether a character should change.

4. SYSTEM ARCHITECTURE

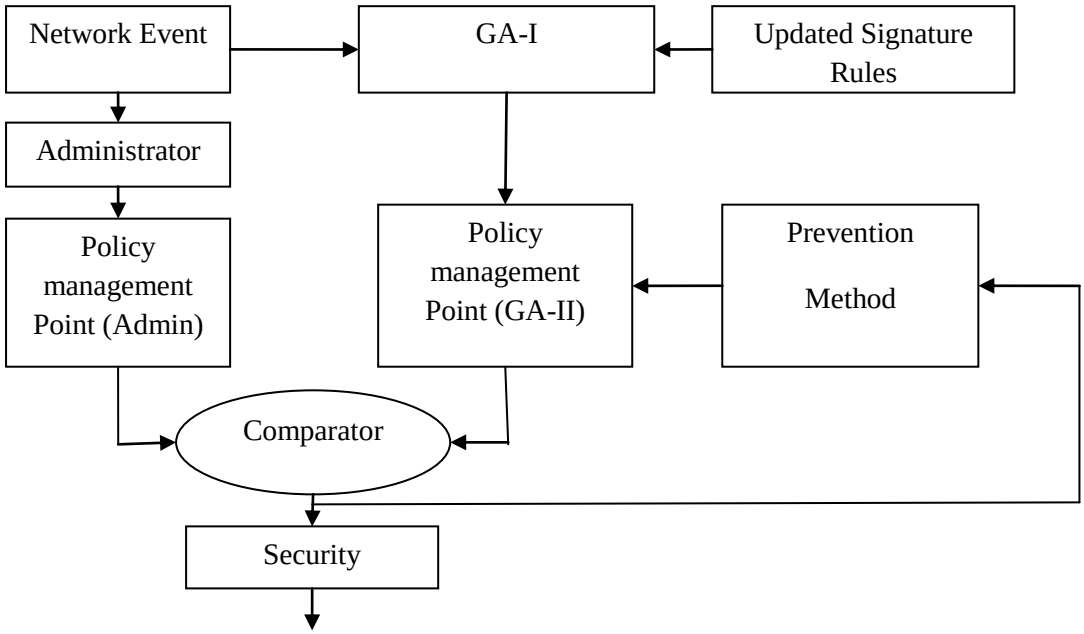

Figure 1: Architecture of Proposed System

Architecture of Proposed system is shown in Figure 1. It consists of following components.

4.1.1 Network Event Module

In this module Administrator selects the network event or packets. Here each network event is analyzed and passes network event to the GA-I.

4.1.2 Administrator

The user should be login with administrator so Jpcap and WinPcap tool can be detected the packets otherwise we could not be detect any packets.

4.1.3 Policy Management Component for Administrator

In this module Administrator selects the policy and passes that policy to the comparator.

4.1.4 Genetic Algorithm Component (GA-I)

GA-I used for packet analysis. We have analyzed server side packets also. We used vector class and analyzed n number of packets.

4.1.5 Policy Management Component for Genetic Algorithm Component (GA-II)

Basically GA-II used for policy selection and prevention purposes. Here types of attacks are analyzed and its vulnerability checks.

4.1.6 Comparator

In this component we are comparing two policies administrator policies and policy management point (GA-II). We are recycling policy and passing the comparator output to a network and we are choosing best of best policy.

4.1.7 Prevention Method Component

In this component prevention method is applied to infected networking terms and passes the policy to Policy Management Point (GA-II).

4.1.8 Security Component

In this component security has been applied for the best of best policy. The database has been updated whenever we are analyzing lots of packets through the network.

The subcomponents of Genetic Algorithm are shown in figure 1.2

The network security policy framework and analysis consist of following subcomponents:-

4.2.1 Gene Designer

Gene designer is used to get input as a network packet. The properties can be source and destination IP address, port number, size of the packet, in case of security breach the level of threat and damage caused, depending on type of security breach.

4.2.2 Genetic Operation Unit

In this component model genetic operations such as crossover, mutation, and selection apply to the initial set of the population selected by the administrator.

4.2.2 Gene Pool

In this component the all the gene selected during genetic operation based on their fitness score are stored along with their fitness value for future references.

4.2.2 Gene Comparator

In this component the gene generated by gene designer is compared with the gene present in the gene pool. If the gene is present in the gene pool then the output of the component is

forwarded to Event action model. If the gene is not present then the fitness value of the gene is calculated in the fitness calculator.

4.2.3 Fitness Calculator

Here the fitness of the gene is calculated and if the score is more than the threshold value decided for the fitness function then the gene added in the gene pool and the output is sent to the network report generator.

Here we used a genetic algorithm (GA-I) for packet analysis and (GA-II) for policy selection and prevention method.

In packet capturing Jpcap and WinPcap tool is used for implementation for this paper. These are the open source software. We used Jpcap 0.7, Netbeans-7. 0.1-male-windows WinPcap-4-1-2 and jdk-7 (Advance Java) used for implementation of this paper. Database saved in ms access.

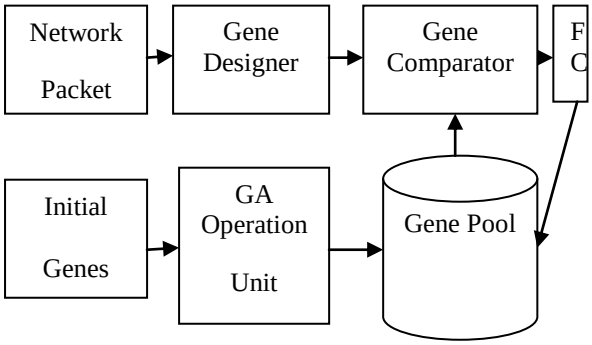

Figure 2 Components of proposed System

FC: -Fitness Calculator

FITNESS FUNCTION

There are many parameters that can influence the effectiveness of the genetic algorithm. The evaluation function is one of the most important and difficult parameters in genetic algorithms. First we define a formula to calculate whether a field of the connection matches the pre-classified data set. In a chromosome header fields are taken. The data type can be character, integer, double or an object. If a particular packet has been matched to a number of rules decided and we are calculating difference zero or one, then best delta value can be computed.

$$\text{Match\_Value} = \sum_{i=0}^{n} \text{match} * \text{weight}_i$$

Where "n" is the number of genes present in each chromosome. In our case gene means property that each network packet is to be checked for, here each network packet is equal to a chromosome.

Some of the properties which might be considered as genes for a network event are as follows:

1. Source IP address.

2. Destination IP address

3. Source port number

4. Destination port number

5. Size of packet

6. Number of hops between the source and destination.

7. Time to Live (TTL)

8. Packet type

9. Payload

10. Checksum

11. Sequence number

Here, THREAT is taken as the fitness function. So the THREAT value of each event is being calculated and if it is above the threshold decided, then the packet is considered as dangerous.

$$\text{THREAT\_VALUE} = \sum_{i=0}^{n} \text{THREAT}_{\text{MATCH}} * \text{weight}_i$$

Table 2 Comparison

Existing system	Proposed system
8 networking parameter	11 networking parameter
Signature rules predefined	Updated Signature rules are predefined.
Latest attack detection and prevention cannot be dealt with.	Latest attack detection and prevention method can be dealt with.

5. TYPES OF ATTACK DETECTION

Most of the attack in an internet is based on hacking IP address in TCP (Transmission Control protocol), UDP (User datagram Protocol), ICMP (Internet Control Message Protocol), ARP (Address Resolution Protocol); We have developed attacker module and analyzed TCP, UDP, ICMP and ARP. Here we have analyzed which IP addresses Block and ICMP Block in an internet or LAN (Local Area Network). In the process module we are taking appropriate action for particular attack. In GA II analyzer module packet id and fitness value is stored. In our database packet ID and which packet you are checking its weight and threat value is stored. In our database packet ID and which packet you are checking its weight and threat value is stored. How many packets we have analyzed this is checked by enabling flag one in analyzed column. Weight and threat value is stored for UDP, TCP, ICMP and ARP Protocol. In GA II Process module Packet ID and fitness value is stored for any web site or we can say particular network.

In process module whether the packet has been detected if the checked value is one in GA II. We are storing ID, PacketID, and fitness value in this module. Packet ID and fitness value is stored and which action we have to take whether it is IP Block or ICMP Block this overall content will be stored in policy

module for the purpose of prevention method. We have analyzed the active attack, passive attack.

In ICMP (Internet Control Message Protocol) Source IP address remains same for Destination IP address this is also one type of attack.

6. RESULT

6.1 Efficiency of System.

Testing Network Security Policy Framework and Analysis uses real data set is a difficult task. When infected packets found our system must be updated just like antivirus software. The following table shows the efficiency of the system.

$$\text{Efficiency of System} = \frac{\text{Total no of networking parameter checked in previous system}}{\text{Total no of networking parameter checked in our system}} * 100$$

$$\text{Efficiency of System} = \frac{8}{11} * 100$$

Efficiency of System = 73%.

Above Efficiency shows that our developed system is working 73% correctly.

Figure 3 indicates the pie chart for the efficiency of our system is indicated in blue portion (73 %). We have checked 11 networking parameter while the purple portion (27.28%) is indicating we haven't checked all the networking parameter.

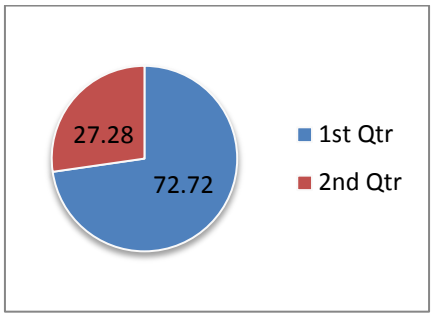

Figure 3 Efficiency of our system

7. CONCLUSION AND FUTURE WORK

In network security policy, framework and analysis will run on the web server and its function to detect attack, threat value and weight. Network security policy framework and analysis are carried out using genetic algorithm. After detecting an attack, we have to choose best of best policy. Experimental results show the performance and effectiveness of our system and demonstrate the usefulness of Genetic Algorithm. Comparative studies reveal that the Accuracy of the system is close to 73 percent over a wide variation in the input data and results are good as compared to previous papers due to increasing the networking parameter. We have checked 11 networking parameter

Following work would be possible.

1. A new model that is much more simplified and implementable was implemented , future work to this includes creating the network security framework and testing it on real data set available on the internet.
2. In this model approximately eleven signature rules are predefined. As future expansion in this work, latest attack prevention can be dealt with.
3. It's possible to implement the overall thesis through cluster but it will require large time complexity as compared to the genetic algorithm.

8. REFERENCES

[1] Atish Mishra, Arun Kumar Jhapate, Prakash Kumar ,"Improved Genetic Feedback Algorithm Based Network Security Policy Framework" in Second International Conference on Future Networks on page 8-10. http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?armunber=5431893

[2] Chen Xiao-su Wu Jin-hua Ni Jun 2007 in Wireless Communications, Networking and Mobile Computing, WiCom 2007.Internatinal Conference on page 2278-2281.

[3] Hang-Fei Wang, Xu-Fa Wang and Jia Xue. "An Improved Interactive Genetic Algorithm Incorporating Relevant Feedback"

[4] Atish Mishra, Arun Kumar Jhapate and Prakash Kumar, 2009 "Designing Rule base for Genetic Feedback Algorithm Based Network Security policy Framework using State Machine" on pp 415-417 at International Conference on Signal Processing Systems.

[5] Hoi Chan; Kwok, T.; IBM Thomas J. Watson Res. Center, Hawthorne, NY June 2006 "A Policy-Based Management System with Automatic Policy Selection and creation capabilities by using a Singular Value Decomposition Technique" .

[6] Atish Mishra, Prakash Kumar 2009 "Storing Scheme for State Machine Based Rule Base of Genetic Feedback Algorithm Based Network Security Policy Framework Depending on Memory Consumption" at IACSIT vol, 3 Singapore.

[7] Jeffrey O. Kephart and William E.Walsh IBM Thomas J. Watson "An Artificial Intelligence Perspective on Automatic Computing Policies" at Research Center Yorktown Heights, New York 10598.

[8] L.P.Kaelbling, M. L. Littman, A. W. Moore. "Reinforcement Learning: A Survey"

[9] Biing Feng Wang, Chien-Hsin Lin. "Improved Algorithms for Finding Gene Teams and Constructing Gene Team Trees"

[10] R. Yavatkar,D.Pendarakis, R. Guerin, January 2000, RFC 2573, "A Framework for Policy-based Admission Control", http://www.faqs.org/rfcs/rfc2753.html.

[11] B. Abdullah*, I. Abd-alghafar**, Gouda I. Salama** and A. Abd-alhafez , "Performance Evaluation of a Genetic Algorithm Based Approach to Network Intrusion Detection System" in 13th International Conference on AEROSPACE SCIENCES & AVIATION TECHNOLOGY, ASAT- 13, May 26 – 28, 2009. http://www.mtc.edu.eg/ASAT13/pdf/CE14.pdf

[12] Gihan Nagib and Wahied G. Ali, "Network Routing Protocol using Genetic Algorithms" in International Journal of Electrical & Computer Sciences IJECS-IJENS Vol: 10 No: 02 10 March 2010. http://www.ijens.org/104302-8686%20IJECS-IJENS.pdf

[13] Suhas Chavan, L.M.R.J Lobo "Network Security Policy Framework and Analysis" in International Journal of Computer Application special issue on Network Security and Cryptography NSC (1):55-58, December 2011. http://www.ijcaonline.org/specialissues/nsc/number1/4325-spe014t

[14] Mohammad Sazzadul Hoque1, Md. Abdul Mukit2 and Md. Abu Naser Bikas "An implementation of intrusion detection system using genetic algorithm" in International journal of network security & its applications (ijnsa), Vol.4, No.2, march 2012 http://www.airccse.org/journal/nsa/0312nsa08.pdf