International Research Journal of Computer Science (IRJCS)
Issue 11, Volume 3 (November 2016)
ISSN: 2393-9842
www.irjcs.com

A Study of Intrusion Detection System using Advanced Genetic Algorithm

Aman V. Mankar *B.E. Computer Sci. & Engineering*
Tushar C. Ravekar *M.Tech. Information Security*
PRMIT&R, Badnera College of Engineering, Pune

Abstract* — As of today, we are relying more and more on Internet or network computer access, a growing problem intrusion into computer systems by unauthorized users has been observed. An intrusion is unauthorized access or attempted access into or unauthorized activity in a computer or information system. Intrusion detection technologies are therefore becoming extremely important to improve the overall security of computer systems. Intrusion detection is the process of identifying that an intrusion has been attempted, is occurring or has occurred. In this paper, we are focusing on Intrusion Detection System based on genetic algorithm (GA).

***Keywords*— Intrusion Detection System, Intrusion Prevention System, Genetic Algorithm, Threats, Attacks, Hacks, Firewall**

I. INTRODUCTION

Intrusion is some time also called as hacker or cracker attempting to break into or misuse your system. While introducing the concept of intrusion detection in 1980, defined an intrusion attempt or a threat to be the potential possibility of a deliberate unauthorized attempt to

- *Access information*
- *Manipulate information*
- *Render a system unreliable or unusable.*

Intrusion detection systems (IDS) do exactly as the name suggests: they detect possible intrusions. More specifically, IDS tools aim to detect computer attacks and/or computer misuse, and to alert the proper individuals upon detection. An intrusion detection system inspects all inbound and outbound network activity and identifies suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system. An IDS installed on a network provides much the same purpose as a burglar alarm system installed in a house. Through various methods, both detect when an intruder/attacker/burglar is present, and both subsequently issue some type of warning or alert.

IDPSs typically record information related to observed events, notify security administrators of important observed events, and produce reports. Many IDPSs can also respond to a detected threat by attempting to prevent it from succeeding. They use several response techniques, which involve the IDPS stopping the attack itself, changing the security environment (e.g., reconfiguring a firewall), or changing the attack's content.

Hence, we need IDS in our regular use of network as it may protect us from malicious activities which are invisible to us but they are lightly or severely harmful for us. So, IDS are important for home user, server, workstations, government security portal, etc.

Genetic Algorithm is selected because of some of its attributes, e.g., strong to noise, no gradient information is required to find a global optimal solution, self-learning, etc. Using Genetic Algorithms for intrusion detection has proven to be an effective approach. In this research, we implemented software based approach. The experimental results show that our approach is effective, and it has the litheness to either generally detect network intrusions or precisely classify the types of misuse intrusions.

Following paper is organised as follows. Section II gives an overview of intrusion detection system using Genetic Algorithm method. Section III gives a brief overview of how to implement it. Section IV discusses the experimental results. Section V gives a conclusion.

II. Overview of Genetic Algorithm

Genetic algorithms (GA) are a branch of evolutionary algorithms used in search and optimization techniques. The three dominant functions of a genetic algorithm i.e., selection, crossover and mutation correspond to the biological process: The survival of the fittest. In a genetic algorithm, there is a population of strings (called chromosomes or the genotype of the genome), which encode and indent solutions (called individuals, creatures, or phenotypes). Traditionally, solutions are represented in binary as strings of 0s and 1s, but other encodings are also possible. The evolution usually starts from a population of randomly generated individuals and evolves over generations.

In each generation, the fitness of every individual in the population is evaluated, multiple individuals are stochastically selected from the current population (based on their fitness), & modified (recombined and possibly randomly mutated) to form a new population. The new population is then used in the next iteration of the algorithm. Commonly, the algorithm terminates when either a maximum number of individuals are there in a generation, or a satisfactory fitness level has been reached for the population. If the algorithm has terminated due to a maximum number of individuals, a satisfactory solution may or may not have been reached.

2.1 Functions performed during GA process:

1. Applying fitness function:
Fitness function (or objective function) defines the problem constraints; it measures the performance of all chromosomes in the population. Fitness function is the heart of all Genetic Processes. In our approach, we have used:

Fitness = (size * weight)

Where the size is the actual packet data size prescribed by the incoming packet data stream and weight is the vector applied by each chromosome.

2. Selection operator:
Selection Operator determines which chromosome(s) from the population will be chosen for recombination; depends on the fitness of the chromosome. The selected chromosomes are called parents. Selection methods are as follows:

- *Fitness-proportion selection.*
- *Roulette-wheel selection.*
- *Rank selection.*
- *Local selection.*
- *Steady state selection*

3. Crossover operator:
The parent's chromosomes are recombined by one of the crossover methods. It produces one or more new chromosomes called offspring. Such methods are: Single Point Crossover, Multipoint Crossover, Uniform Crossover and Arithmetic Crossover.

4. Mutation operator:
New genetic material could be introduced into the new population through mutation process. This will increase the diversity in the population. For each offspring mutation randomly alters some genes. A commonly used method for mutation is called single point mutation. Though, a special mutation types used for varies problem kinds and encoding methods. So, we are having Single point mutation and multi point mutation.

2.2 Genetic Algorithm Process:

GA evolves the population of chromosomes (individuals) as the process of natural selection. It generates new chromosomes during its process. GA process uses a set of genetic operators (selection operator, crossover operator and mutation operator), and evaluate chromosome using the fitness function.GA consists of population of chromosomes that reproduced over set of generations according to their fitness in an environment. Chromosomes that are most fit are most likely to survive, mate, and bear children.GA terminate the process by define fixed maximal number of generations or as the attainment of an acceptable fitness level, or if there are no improvements in the population for some fixed generations, or for any other reason. The standard GA processes is shown in the following figure.

It contains various steps which include: encoding chromosomes, generating initial population, fitness function evaluation, and then applying one of the operators. The process will stop when we get the best individuals.

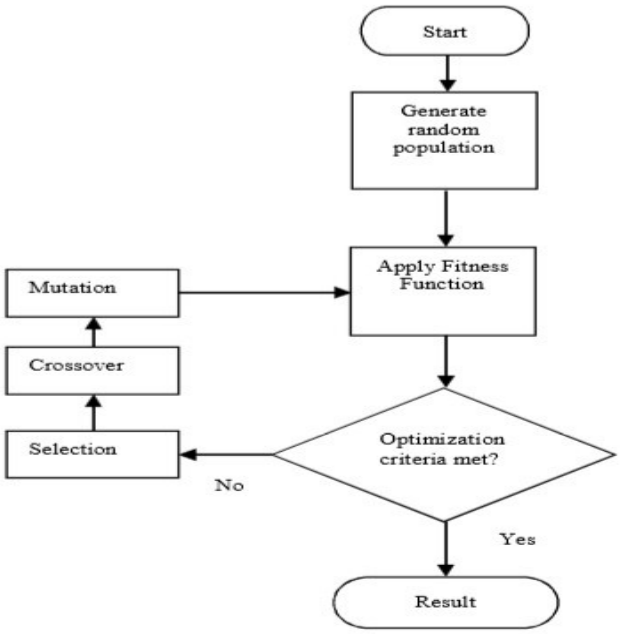


Figure 1: Genetic Algorithm Process

2.3 GA OPERATORS ENCODING OF THE CHROMOSOMES:
In the GA process, it is important to represent the data into some of the encoding formats. One outstanding problem associated with encoding is that some individuals correspond to infeasible or illegal solutions to a given problem. Various encoding methods have been created for particular problems to provide effective implementation of genetic algorithms. The encoding methods can be classified as follows:

A. BINARY ENCODING:
Binary encoding (i.e., the bit strings) are the most common encoding used for several of reasons. One is historical: in their earlier work, Holland and his students concentrated on such encodings and genetic algorithms practices have tended to follow this lead. Another reason for that was because much of existing GAs theories is based on the assumption of using binary encoding.

B. REAL-NUMBER ENCODING:
Real number encoding is best used for function optimization problems. It has been widely confirmed that real number encoding performs better than binary encoding for function optimization and constrained optimizations problems. In real number encoding, the structure of genotype space is identical to that of the phenotype. Therefore, it is easy to form effective genetic operators by borrowing useful techniques from conventional methods.

C. INTEGER OR LITERAL PERMUTATION ENCODING:
Integer or literal permutation encoding is best used for combinational optimization problems because the essence of this kind of problems is to search for the best permutation or combination of items subject to constrains.

III. IMPLEMENTATION OF A GENETIC ALGORITHM

The scope of experiment is focused to generate list of IP addresses and their packets which are vulnerable to the server or destined system. The testing is done on the entries generated by the firewall system of machine in pfirewall.log file. The training is done on the predefined data rules. The pfirewall.log file contains the entries of incoming packets with various fields like date/time, Action, Protocol, Source IP, Destination IP, src port, dest port, size, flag, ack, type and info. These entries are made available on firewall setting which are available for both successful connection and dropped packets. But for making the connection profile we have used only 5 important fields of it. These are source ip, destination ip, src-port, dst-port and size. The size of pfirewall.log file may also vary with requirements

For this experiment, we have used java as the frontend to make coding part and to write different algorithms and classes. The training data is stored into the wamp server which is used as the backend to the system. Wamp server is able to store the different structures of dataset tables. For this experiment, we used windows based HP computer with i5 processor system having 500Gb hard disk space and 2 GB RAM to execute the computer program.

The generated rules are stored in a rule base in the following form:

```
if
 {condition}
then
 {act}
```

For example, a rule can be defined as:

```
if
 {the connection has following information:
 source IP address 124.12.*.*; destination IP address: 130.18.206.55;
 destination port number: 21; connection
 time: 10 seconds; protocol = UDP}
then
 {attack type = port-scan;
 stop the connection}
```

Numeric representation for Crossover and Mutation

```
{124, 12, -1, -1, 10, 251, -1, -1, 21, 10, 1, 2, 999}
```

Here 1 and 2 represents the protocol UDP and attack type port-scan and 999 represents stopping the connection.
Explanation**:** if there exists a network connection request with source IP address 124.12.*.*, destination IP address 10.251.*.*, destination port number 21, connection time 10 seconds and protocol = UDP, then stop the connection establishment – since IP address 124.12.5.18 is recognized by the IDS as a blacklisted IP address for performing "port-scan". Thus, service request initiated from it, is rejected.

3.1 System Architecture

The detail proposed architecture is shown in figure 4. It starts from initial population generation from **pfirewall.log** file generated by the firewall system. The packets are the filtered out on the basis of rules. Then the précised data packets go through several steps namely selection, crossover and mutation operation. These processes get generate best individuals.

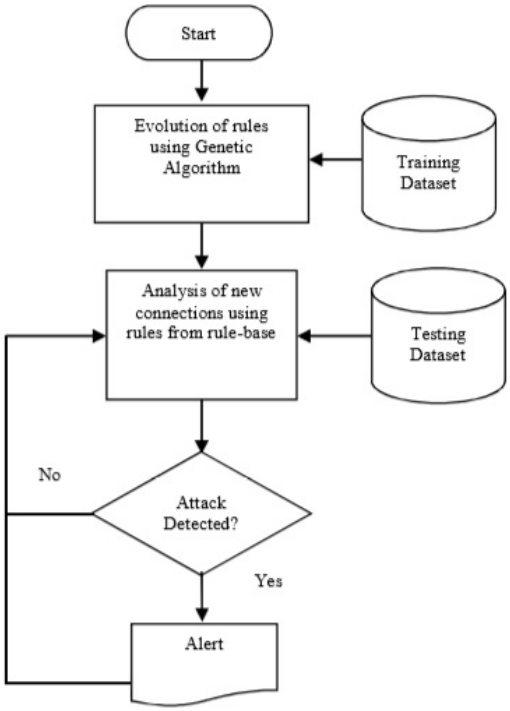
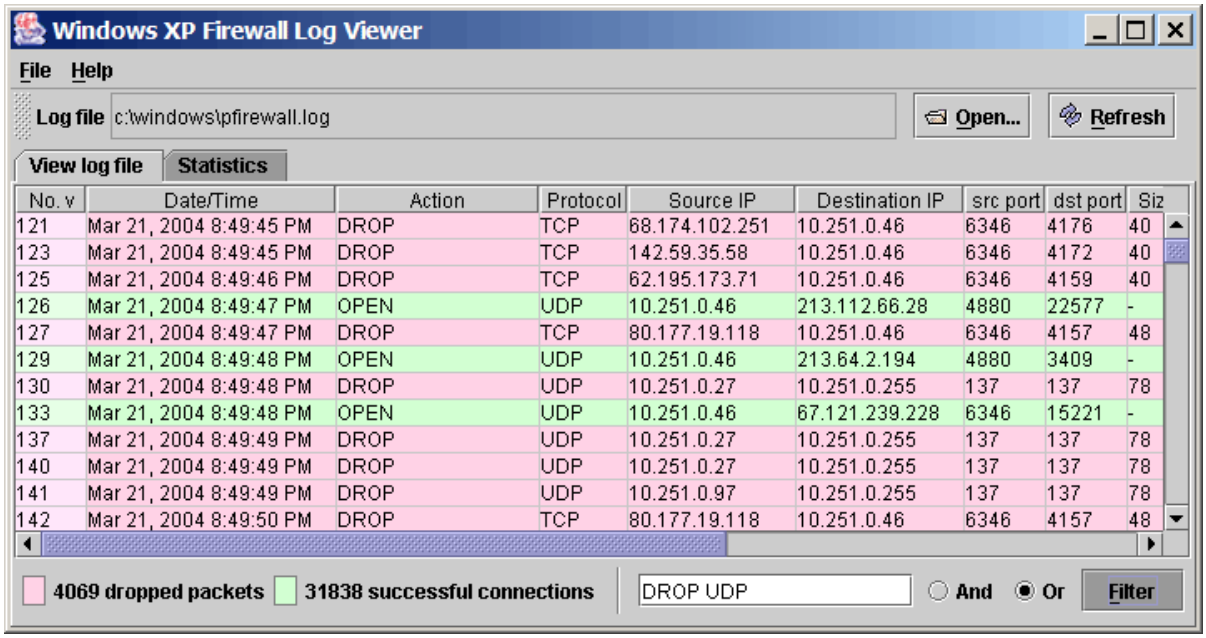


Figure 2: Architecture of Genetic Algorithm

Then the whole set of new generation and old ones are saved as a new possible rule of the malicious activities. If the matching pattern or suspicious behavior is found, then the Alert is sent to the administrator.

IV. EXPERIMENTAL RESULTS

From the above experiment, we have able to create a rule base that could successfully categories harmful and harmless connection types. We have shown the resultant figures below by applying 100 connection entries respectively to the proposed system. After that we were able to get around 96% of accuracy to classify the connections types.



No. v	Date/Time	Action	Protocol	Source IP	Destination IP	src port	dst port	Siz
121	Mar 21, 2004 8:49:45 PM	DROP	TCP	68.174.102.251	10.251.0.46	6346	4176	40
123	Mar 21, 2004 8:49:45 PM	DROP	TCP	142.59.35.58	10.251.0.46	6346	4172	40
125	Mar 21, 2004 8:49:46 PM	DROP	TCP	62.195.173.71	10.251.0.46	6346	4159	40
126	Mar 21, 2004 8:49:47 PM	OPEN	UDP	10.251.0.46	213.112.66.28	4880	22577	-
127	Mar 21, 2004 8:49:47 PM	DROP	TCP	80.177.19.118	10.251.0.46	6346	4157	48
129	Mar 21, 2004 8:49:48 PM	OPEN	UDP	10.251.0.46	213.64.2.194	4880	3409	-
130	Mar 21, 2004 8:49:48 PM	DROP	UDP	10.251.0.27	10.251.0.255	137	137	78
133	Mar 21, 2004 8:49:48 PM	OPEN	UDP	10.251.0.46	67.121.239.228	6346	15221	-
137	Mar 21, 2004 8:49:49 PM	DROP	UDP	10.251.0.27	10.251.0.255	137	137	78
140	Mar 21, 2004 8:49:49 PM	DROP	UDP	10.251.0.27	10.251.0.255	137	137	78
141	Mar 21, 2004 8:49:49 PM	DROP	UDP	10.251.0.97	10.251.0.255	137	137	78
142	Mar 21, 2004 8:49:50 PM	DROP	TCP	80.177.19.118	10.251.0.46	6346	4157	48

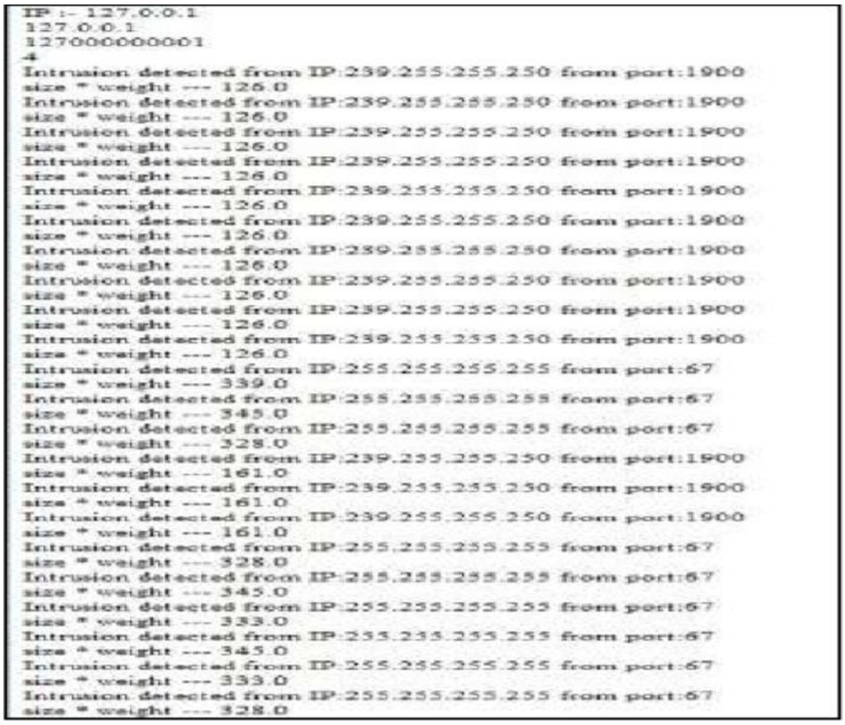
Figure 3: pFirewall.log file by Windows Firewall & entries taken by the proposed system for filtration.

SETTING TYPE	VALUES
Encoding Scheme	Binary Encoding
Population size	100
Evolution generation	50
Selection	Fitness-proportion
Crossover	One point
Mutation	Real number
Generations	50
Fitness function	Weight * pkt_size

Figure 4: Parameter setting for Genetic Algorithm

121	Mar 21, 2004 8:49:45 PM	DROP	TCP	68.174.102.251	10.251.0.46	6346	4176	40
123	Mar 21, 2004 8:49:45 PM	DROP	TCP	142.59.35.58	10.251.0.46	6346	4172	40
125	Mar 21, 2004 8:49:46 PM	DROP	TCP	62.195.173.71	10.251.0.46	6346	4159	40
126	Mar 21, 2004 8:49:47 PM	OPEN	UDP	10.251.0.46	213.112.66.28	4880	22577	-
127	Mar 21, 2004 8:49:47 PM	DROP	TCP	80.177.19.118	10.251.0.46	6346	4157	48
129	Mar 21, 2004 8:49:48 PM	OPEN	UDP	10.251.0.46	213.64.2.194	4880	3409	-
130	Mar 21, 2004 8:49:48 PM	DROP	UDP	10.251.0.27	10.251.0.255	137	137	78
133	Mar 21, 2004 8:49:48 PM	OPEN	UDP	10.251.0.46	67.121.239.228	6346	15221	-
137	Mar 21, 2004 8:49:49 PM	DROP	UDP	10.251.0.27	10.251.0.255	137	137	78

Figure 5: Highlighted entries in blue are eliminated by the rule base.



```
IP :- 127.0.0.1
127.0.0.1
127000000001
4
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 126.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 339.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 345.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 328.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 161.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 161.0
Intrusion detected from IP:239.255.255.250 from port:1900
size * weight --- 161.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 328.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 345.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 333.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 345.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 333.0
Intrusion detected from IP:255.255.255.255 from port:67
size * weight --- 328.0
```

Figure 6: Final list of IP addresses generated by GA.

V. CONCLUSIONS

In this paper, we present and implemented an Intrusion Detection System by applying genetic algorithm to efficiently detect various types of network intrusions. A simple, efficient and flexible fitness function was used to evaluate goodness of each rule (chromosome). Depending on the selection of fitness function weight values, the generated rules can be used to either generally detect network intrusions or precisely classify the types of intrusions.

The three factors which have impact on the effectiveness of the genetic algorithm are selection of fitness function, representation of individuals and values of the GA parameters. The determination of these factors often depends on applications. Designing accurate fitness function is the major challenge for solving a particular problem. Different models for designing fitness function have been discussed. Using GA for intrusion detection has proven to be a cost-effective approach. One of the major advantages of this technique is due to the fact that in the real world, the types of intrusions change and become complicated very rapidly. The GA based detection system can upload and update new rules to the systems as the new intrusions become known. Therefore, it is cost effective and adaptive.

VI. REFERENCES

[1].T. Lunt, A. Tamaru, F. Gilham, R. Jagannathan, P. Neumann, H. Javitz, A. Valdes, and T. Garvey. ―A real-time intrusion detection expert system (IDES)‖ - final technical report. Technical report, Computer Science Laboratory, SRI International, Menlo Park, California, February 1992.

[2].Vivek K. Kshirsagar, Sonali M. Tidke & Swati Vishnu "Intrusion Detection System using Genetic Algorithm and Data Mining: An Overview" *International Journal of Computer Science and Informatics Vol-1*, Iss-4, 2012

[3].Bezroukov, Nikolai. 19 July 2003. "Intrusion Detection (general issues)." *Softpanorama: Open Source Software Educational Society. URL:* http://www.softpanorama.org/Security/intrusion_detection.shtml

[4].Mit H. Dave, Dr. Samidha Dwivedi Sharma Improved Algorithm for Intrusion Detection Using SNORT *International Journal of Emerging Technology and Advanced Engineering Volume 4*, Issue 8, August 2014

[5].A. Ahmad Sharifi, B. Akram Noorollahi, and Farnoosh Farokhmanesh "Intrusion Detection and Prevention Systems (IDPS) and Security Issues" *International Journal of Computer Science and Network Security, VOL.14 No.11, November 2014*

[6].Wei Li ―Using Genetic Algorithm for network intrusion detection‖ SANS Institute 2004.

[7].B. Upalhaiah, K. Anand, B. Narsimha, S. Swaraj, T. Bharat, ―Genetic Algorithm Approach to Intrusion Detection System‖ ISSN: 09768491 (online) | ISSN: 2229-4333 (print), IJCST VOL3, ISSUE 1, JAN-MARCH 2012

[8].Shaik Akbar, Dr. J. A. chandulal, Dr. K. Nageswara Rao, G. Sudheer Kumar, ―troubleshooting technique for intrusion detection system using genetic algorithm‖, IJWBC, vol1(3), december 2011

[9].Shrinivasa K G, Saumya chandra, Sidharth Kajaria, Shilpita mukharjee, ―IGIDS: Intelligent intrusion detection system using Genetic Algorithm‖, 978-1-4673-0126-8/11/2011 IEEE.