GENETIC ALGORITHM IN COMPUTER NETWORK PROTECTION

Andon Lazarov[1*] and Petia Petrova[2]

[1] *Nikola Vaptsarov Naval Academy,*
73, V, Drumev St., 9002 Varna, Bulgaria,
e-mail: a.lazarov@naval-acad.bg

[2] *Burgas Free University,*
62, San Stefano St., 8001 Burgas, Bulgaria,
e-mail: ppetrova83@abv.bg

Abstract. The present article addresses main functionality and structure of the genetic algorithms applied for detection of malware network intrusions. The sequence of network connection characteristics is interpreted as a genetic chromosome or an intrusion detection rule. The network characteristics, attributes of the connection are chromosome's genes. An extended list of basic TCP/IP network characteristics and their structure is suggested and illustrated in examples of network connections. The syntax and structure of the chromosomes (rules) coded by terms of bits and integers are presented. The basic chromosome operations in generation of a particular population and experimental assessment of Firas Alabsi's Fitness function are demonstrated.

Keywords: genetic algorithm, network intrusion detection, chromosome structure, intrusion detection rule, Fitness function.

1. INTRODUCTION

The contemporary computer networks due to the vulnerability of their software components are subject to threats and attacks. It exposes the network functionality at great risk [1, 2]. To counteract intrusions and cyberattacks with viruses and malware, various antivirus software and techniques as intrusion detection systems, password and message encryption are applied to overcome threats and reduce the risk [3].

The issue of recognizing attacks and detecting intrusions into the computer network is particularly relevant. One approach to detecting and preventing computer intrusions is the application of genetic algorithms [4]. A genetic algorithm for detecting different types of intrusions in computer networks and

* Corresponding author.

DOI: 10.7546/EngSci.LIX.22.01.07

assessing the genetic evolution process in the intrusion detection is presented in [5]. The specific structure and basic characteristics make genetic algorithms suitable for detecting malicious intrusions into computer networks. The sequence of network characteristics (IP addresses, port addresses, durations, protocols, etc.) or attributes occupying certain positions in the communication pattern is interpreted as a chromosome, and the constituent components as genes. During the evolutionary process, recombination and mutation operators are used to simulate the natural reproduction and mutation of generated rules.

A software implementation of a genetic algorithm for detecting network intrusions is suggested in [6]. A genetic algorithm to select chromosomes based on their steady state is described in [7]. A genetic algorithm for classifying and detecting network intrusions based on address and temporal information and an evolutionary decision-making process are considered in [8]. A genetic algorithm to generate classification rules (chromosomes) and identify cyber-attacks based on network attributes, such as protocol type, duration, service function, is developed in [9].

A Fitness function is used to assess the suitability of the chromosomes. By assessing the Fitness function of a chromosome, a decision is made to detect intrusion. An original Fitness function using an award-punishment technique for the effective estimation of population chromosomes has been proposed in [10]. A structural description of the genetic algorithm and application of various Fitness functions for rules evaluation are presented in [11].

The elementary structure of the Fitness function used in the genetic algorithm to detect smurf (DOS) and probe attacks is shown in [12]. An analytical expression for calculating the Fitness function to assess the suitability of the rule is presented in [13]. The application of a genetic algorithm for achieving security in computer networks and the development of an exemplary encryption algorithm are discussed in [14].

A genetic algorithm and a tree structure for deciding on computer network intrusion by analyzing Sun data are discussed in [13, 15]. A system for detecting network intrusions by applying an algorithm based on the fuzzy logic is proposed in [16]. The genetic algorithm finds the optimal solution in penetration detection through a Fitness function, which evaluates the extent to which selected individuals (chromosomes) optimize the solution. A genetic algorithm pattern to realize self-organizing agile computer security is presented in [17]. A model of a system for detecting network intrusions by applying linear genetic programming and instruments of artificial intelligence are presented in [18, 19]. A review of algorithms for detecting intrusions into computer net-

works is suggested in [20]. Mathematical modelling of internet user's behavior in penetrating computer systems is presented in [21].

The goal of the present study is to define the syntax of the genetic algorithm, to analyze the network characteristics (genes) of the rules, and to describe the structure of the Fitness function evaluating chromosomes in the genetic algorithm applied for network intrusions detection.

The rest of the article is organized as follows. In Section 2 the syntax and structure of the chromosomes (rules) are described. In Section 3 the coding of genes in the chromosome is discussed. In Section 4 the algorithm used to calculate the Fitness function and estimate chromosomes in the genetic algorithm is described. In Section 5 the basic chromosome operations in generating a particular population and experimental assessment of Firas Alabsi's Fitness function are presented. In Section 6 general conclusions are formulated and guidelines for future research in the field of detection and prevention of computer networks intrusions are provided.

2. SYNTAX AND STRUCTURE OF THE CHROMOSOME IN THE GENETIC ALGORITHM

The genetic algorithm is an evolutionary computational search and optimization process that is used to detect malicious intrusion. The algorithm operates with sequences of network characteristics – individuals, called chromosomes and mimics biological genetic processes. The intrusion detection is a process of generating a quality chromosome with genes, i.e. network characteristics maximally corresponding to the computer intrusion. Based on the initial group of chromosomes a population of chromosomes is generated by applying biological operations: selection, recombination and mutation. The quality of each population is evaluated through an objective Fitness function.

Genetic algorithms are applied to generate rules or chromosomes and estimate the type of the computer network traffic. The rules are used to distinguish normal and abnormal (intrusion) network connections. The rules stored in the knowledge (rules) database of the intrusion detection system can be defined by the predicate logic clause syntax [8, 11]:

$$\textit{If } \{\textbf{condition}\} \textit{ then } \{\textbf{act}\}, \tag{1}$$

where the part *if* of the clause defines a state (**condition**) described by network characteristics, such as source IP address, destination IP address, TCP and UDP port numbers, TCP/IP network protocols, connection duration, including indication of intrusion probability. In this part of the rule, the genetic

algorithm compares the current network characteristics with the network characteristic stored in the rules database of the intrusion detection system.

The characteristics in the conditional part are connected by a logical "and" operator. The **act** part of the statement (1) refers to an action defined by security rules, such as a warning report to the system administrator, stopping communication, logging a message in system monitored files, or all of the above. Some network features have a greater relative contribution to the definition of network access, connections and communication exchange.

3. NETWORK GENES AND CHROMOSOME CODING

The network genes (network characteristics) of chromosomes (network connections) can be represented by different types of data, binary numbers (bytes), decimal numbers or floating point numbers. This is due to the different format and range of data values for the different network characteristics. Names of the network characteristics – chromosome's attributes, the number of genes defining the attributes of chromosomes and their formats are given in the first, second and third columns of the Table 1, respectively. For example, the "Duration" characteristic has three components (hours, minutes and seconds), each of which is represented by one byte-type gene. Similarly, each of the characteristics "Protocol", "Source Port Number", "Destination port number" is encoded using one gene of integer type, and each of the characteristics "Source IPv4" and "Destination IPv4" have four components (a, b, c, d), each of which is represented by a single byte gene, "Source IPv6" and "Destination IPv6" have eight components (a, b, c, d, e, f, g, h), each of which is represented by a single byte gene.

The attribute "Attack_name" is located in the **act** part of the rule, which classifies the network characteristics at the learning stage or determines the nature of the communication at the intrusion detection stage.

An example of a rule that classifies network communication as a Neptune Denial of Service (DoS) attack is the following chromosome [6]:

if (duration = "0:10:6" and protocol ="finger" and source_port = 17965 and destination_port = 80 and source_ip = "10.18.50.20" and destination_ip = "172.16.102.49") then (attack_name ="Neptune").

The rule shows that if the network packet with a source IP address 10.18.50.20 and source port 17965, and is sent to a destination IP address 172.16.102.49 and destination port 80 using the finger protocol, and the communication duration is 10 min and 6 sec, then it is most likely network a *Neptune* attack that can shut down the destination host (packet destination).

Table 1. Attributes, genetic structure and codes of the network characteristics (chromosomes or rules)

No	Network chromosome's attributes (genes)	Genes' number	Code format
1	Source IP-v 4 address	4	a.b.c.d
2	Source IP-v 6 address	8	a.b.c.d.e.f.g.h
3	Destination IP-v 4 address	4	a.b.c.d
4	Destination IP-v 6 address	8	a.b.c.d.e.f.g.h
5	Source Port Number	1	Integer
6	Destination Port Number	1	Integer
7	Duration	3	h:m:s
8	State	1	Integer
9	Protocol	1	Integer
10	Number of Bytes Sent by Originator	1	Integer
11	Number of Bytes sent by Responder	1	Integer
12	Attack_name	1	Integer

Each rule is encoded as a chromosome using a fixed-length vector, where each network characteristic is encoded using one or more genes of different types (second and third columns of Table 1). The coded chromosome of the rule in the example above gets the form:

{1, 20, 6, 2, 17965, 80, 10, 10, 10, 10, 172, 16, 102, 49, 1}.

A rule defined by exemplary values of chromosomes attributes can be written as [8, 18]:

if {*the connection has following information: source IP address 218.24.107.52; destination IP address: 168.185.215.108; source port number: 53247; destination port number: 80; connection time: 514 seconds; the connection is stopped by the originator; the protocol used is TCP; the originator sent 6287 bytes of data; and the responder sent 45649 bytes of data*} *then* {*stop the connection*}.

The coded chromosome is expressed by the sequence:

/d, a, 1, 8, 6, b, 3, 4, a, 8, b, 9, d, 7, 6, c, 5, 3, 2, 4, 7, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0, 0, 5, 1, 4, 1, 1, 9, 0, 0, 0, 0, 0, 0, 6, 2, 8, 7, 0, 0, 0, 0, 0, 4, 5, 6, 4, 9/,

where the source and destination IP addresses are presented in a hexadecimal format.

The rule can be interpreted as follows:

– *if* network communication with the source IP address 218.24.107.52, destination IP address 168.185.215.108, source port number 42335, destination port number 80, duration 482 seconds, ending with state 11 (communication terminated by creator), using protocol type 9 (TCP), and source sends 6287 bytes of data, respondents send 45649 bytes of data, *then* this is suspicious behavior and can be identified as a potential intrusion.

The validity of the rule is assessed by matching a preliminary set of network attributes marked as abnormal or normal. If the rule is able to find abnormal behavior, an *Award* will be given to the current chromosome. If the rule corresponds to normal communication, a "Penalty" will be imposed on the chromosome [18].

First, rules (chromosomes) are randomly selected and compared with a test chromosome using Fitness function. Then, recombination and mutation are applied on selected chromosomes. Based on the value of the Fitness function, the population of chromosomes is assessed as a set of computer intrusion rules. At the end of the genetic process, the algorithm stops, the rules are selected and added to the database of the intrusion detection system.

The classifying rules for Denial of Serice (DoS) attacks (Smurf, mailbomb), Remote to Local (User) (R2L) attacks (warezmaster, multihop), User to Root (U2R) attacks (snmpguess, buffer-overflow), Probing attacks (ip-sweep, saint), derived from the DARPA-USA Penetration Database, have the following structure [9].

DoS

– Rule 1: *if duration = 0 and protocol_type = tcmp and dst_host_srv_count = 255 and then smurf*;
– Rule 2: *if duration = 1 V 5 V 11 and protocol_type-tcp and dst_host_srv_count >= 2^ <= 247 and then mailbomb.*

R2L

– Rule 3: *if duration = 0 V duration <= 289 and protocol_type = tcp and dst_host_srv_count >= 1^ <= 128 and then waremaster;*
– Rule 4: *if duration = 0 and protocol_type = icmp V top V udp and dst_host_srv_count >= 1^ <= 20 and then multihop.*

U2R

– Rule 5: *if duration = 0 V duration <= 289 and protocol_type = udp and src_bytes-I > and then snmpguess;*
– Rule 6: *if 1 and = 0 and protocol_type = tcp and dst_host_srv_count <= 100 and then buffer–overflow.*

Probe

– Rule 7: *if duration = 0 and protocol_type – icmp and dst_host_srv_count >= 1^ <= 255 and then ipsweep;*
– Rule 8: *if duration = 0 and duration <= 11 and protocol_type – icmp V tcp V udp and dst_host_srv_count >= 1^ <= 255 and then saint.*

The definition of the above rules uses the following notations "<="(less than or equal to), "=>" (equal or greater), ^ (logical "and").

4. FIRAS ALABSI ALGORITHM FOR FITNESS FUNCTION CALCULATION

Firas Alabsi's algorithm for calculating the Fitness function is based on the *Award Penalty* concept [13]. Chromosomes (rules) differ in strength and weakness. Therefore, the Fitness function takes into account, first, an *Award*, which should be as much as the strength of the chromosome, and, second, a *Penalty*, which should be as much as the chromosome weakness. Intrusions into the computer network are classified into five categories: Normal, DoS, Probe, U2R and R2L [12]. Five characteristics (genes) are used to compare chromosomes in each category. The original description of the five main characteristics for each category is as follows [13].

Normal

– hot indicators: Number of "hot" indicators;
– destination bytes: Number of bytes sent from the destination system to the host system;
– source bytes: Number of bytes sent from the host system to the destination system;
– compromised conditions: Number of compromised conditions;
– dst_host_rerror_rate: % of connections that have REJ errors from a destination host.

Probe

– dst_host_diff_srv_rate: % of connections to different services from a destination host;
– rerror_rate: % of connections that have REJ errors;
– srv_diff_host_rate: % of connections that have same service to different hosts;
– logged in: binary decision;
– service: type of service.

DoS

– count: Number of connections made to the same host system in a given interval of time;
– compromised conditions: Number of compromised conditions;
– wrong_fragments: *no* of wrong fragments;
– land: 1 if connection is from/to the same host/port; 0 otherwise;
– logged in: 1 if successfully logged in; 0 otherwise.

U2R

– root shell: 1 if root shell is obtained; 0 otherwise;
– dst_host_srv_serror_rate: % of connections to the same service that have SYN errors from a destination host;
– *no* of file creations: *no* of file creation operations;
– serror_rate: % of connections that have SYN errors;
– dst_host_same_src_port_rate: % of connections to same service ports from a destination host.

R2L

– guest login: 1 if the login is a "guest" login; 0 otherwise;
– *no* of file access: *no* of operations on access control files;
– destination bytes: Number of bytes sent from the destination system to the host system;
– fail logins: *no* of failed login attempts;
– logged in: binary decision;

where *no* denotes a number.

Five tables are built, one table for a category. Each table has a name (type of category) and includes three columns for the parameters A, AB and Fitness function, and five rows for network characteristic. The data in column A and column AB have the following dynamics of change in their values. For example, each characteristic (gene) of DoS category must have values in a certain range in the **condition** part of the rule or to be equal to a specific value in order to evaluate in the **act** part of the rule the incoming sequence as DoS. If the network characteristics are in the **condition** part of the rule (chromosome) and the category name is in the **act** part of the rule (chromosome), then each sequence in the knowledge database is compared with all incoming sequences.

If the **condition** and **act** of the selected sequence in the database are equal to the c**ondition** and **act** of the incoming sequence being compared, then this will increase the value of AB in column AB of the selected sequence by 1. Otherwise, if the **condition** of the selected sequence is equal to the **condition** of the compared sequence, but the actions of the two records do

not match, then the value in column A of the selected sequence will increase by 1.

In other words, if c & a of the selected chromosome = c & a of the compared chromosome, then $AB = AB + 1$. Otherwise, if c of the selected chromosome = c of the compared chromosome, but a of the selected chromosome $\neq a$ of the compared chromosome, then $A = A + 1$ (c is the condition and a is the action).

Fitness function *Award-Penalty* is defined by the following empirically derived analytical expression [10]:

$$\text{Fitness} = 2 + ((AB)/(AB + A)) - A/(AB + A) + AB/X - A/Y, \quad (2)$$

where X is the maximum value of AB in the population of chromosomes, Y is the maximum value of A in the population of chromosomes.

The ratio $(AB/(AB + A))$ reflects the strength of the chromosome. The ratio defined by the expression $(A/(AB + A))$ reflects the weakness of the chromosome. The AB/X ratio determines the chromosome strength relative to the strongest chromosome in the population. $AB/X = 0$ in the worst case (if $AB = 0$) and $AB/X = 1$ in the best case (if AB accepts the highest value of AB in the population). AB/X is added on the right-hand side of the Fitness feature to *Award* the chromosome.

The ratio A/Y determines the chromosome weakness relative to the weakest chromosome in the population. $A/Y = 0$ at best (if $A = 0$) and $A/Y = 1$ at worst (if A assumes the highest value in the population), so the A/Y value must be subtracted from the Fitness function to "punish" the chromosome.

Analysis of the Fitness function

Consider the Fitness function defined by expression (2) written without the coefficient 2. The chromosome with the highest value of AB and $A = 0$ has a Fitness = 2. The chromosome with a highest value of A and $AB = 0$ has a Fitness = -2. However, the Fitness function must rate each chromosome (rule) with a non-negative value. To obtain a non-negative Fitness value of a chromosome it is necessary to add a coefficient 2 on the right-hand part of the expression (2). Then, the range of the Fitness values is 0.4.

5. BASIC OPERATIONS ON CHROMOSOMES OF A GIVEN GENERATION AND EXPERIMENTAL EVALUATION OF FIRAS ALABSI FITNESS FUNCTION

5.1. Selection

When generating each succeeding generation, part of the resulting population is chosen to create a new generation. The decisions for the selection of individuals (chromosomes or rules) are derived through a Fitness-based process, which ensures that individuals with higher values of Fitness function are selected with a high probability. From the population, pairs of chromosomes are selected to be the parents of the next population, i.e. the chromosome pairs to recombine [17].

5.2. Recombination

Recombination creates one or more new generations of parent chromosomes to produce better chromosomes with high Fitness function values.

5.3. Mutation

The mutation randomly changes the new offspring. This is done to prevent all decisions in the chromosome population from falling into the local optimum.

The evaluation of current chromosome generation by Fitness function and test chromosome are shown in Fig. 1. New chromosome generation is produced by recombination and mutation in binary format. The recombination chromosomes are indicated by ♂ and ♀. The invalid chromosome is denoted by •. The unchanged chromosome is denoted by ○. The mutating chromosome is denoted by ⚇. The fitness function for the current chromosome and the test chromosome is estimated as the ratio of the number of matching bits in the chromosomes to full the number of bits in the chromosome.

To confirm the evaluation of a chromosome with a given Fitness function, the chromosome must be tested with another Fitness function. The results of the two Fitness functions are compared. *Support Confidence Framework* as a second control Fitness function is used [11]:

$$\text{Support} = |A \text{ and } B| / N, \tag{3}$$

$$\text{Confidence} = |A \text{ and } B| / |A|, \tag{4}$$

$$\text{Fitness} = w_1 * \text{Support} + w_2 * \text{Confidence}, \tag{5}$$

Current Generation	FF	Crossover&Mutation	New Generation
1011101010001010	0.63♀		1011101010001010
1000011110101000	0.43●	1011100100001001	1011100100001001
1100101000101011	0.50●	0101111010001010	0101111010001010
0101110100001001	0.75♂		0101110100001001
0101000111010101	0.56♌	0101000011010101	0101000011010101
0010111000101011	0.56○		0010111000101011

Test Chromosome
0101111010001101

Fig. 1. Evaluation of current chromosome generation through Fitness function (FF) and test chromosome, recombination and mutation, and new generation of chromosomes in binary format

where Support shows the relative number of AB repetitions within all chromosomes (rules) in the population. Confidence shows the relative number of AB within those rules that have the same condition. w_1 and w_2 are threshold or weight values to balance the Support value with the Confidence value. It is accepted $w_1 = 0.0257$ and $w_2 = 0.9843$.

If the Fitness function of the rule (chromosome) P is greater than the Fitness function of the rule (chromosome) Q according to the first Fitness algorithm, then the Fitness function of the rule P is also greater than the Fitness function of the rule Q according to the second Fitness algorithm. For each chromosome in the population, there are two results R_1 and R_2, defined by the equations [14]:

$$R_1 = \frac{Fv_1}{\max Fv_1 \text{ in Population}}, \tag{6}$$

$$R_2 = \frac{Fv_2}{\max Fv_2 \text{ in Population}}, \tag{7}$$

where Fv_1 is the Fitness function based on the *Award-Penalty* Fitness algorithm, Fv_2 is the Fitness function based on the *Support Confidence Framework* Fitness algorithm applied on the same chromosome. The quality of the Fitness

function, i.e. it gives a good result, is confirmed if the values of R_1 and R_2 are close to each other.

Estimates of A, AB and the Fitness function with application of Firas Alabsi algorithm, obtained with data of Normal, DoS, R2L, U2R, Probe network communications, realized with randomly generated network characteristics of five chromosome structures for each category, are given in Tables 2–6.

Table 2. Values of A, AB and Fitness function of experimental records (chromosomes) for Normal, DoS, R2L, U2R, Probe simulated network communications

Normal		
A	AB	Fitness
6	5	1.910
0	12	3.210
69	3	0.136
2	56	3.900
1	27	3.390

Table 3. Values of A, AB and Fitness function of experimental records for DoS simulated network communications

DoS		
A	AB	Fitness
7195	31	0.0921
349	371	2.9820
4	9	2.4080
12	54	2.7800
87	226	3.0410

Table 4. Values of A, AB and Fitness function of experimental records for R2L simulated network communications

R2L		
A	AB	Fitness
5	21	2.6410
1	7	2.7500
148221	19	0.0235
513	12	1.0570
19	817	3.9540

The results in Tables 2–6 can be analyzed as follows. The maximum value of the Fitness function for the chromosome from categories Normal, R2L, U2R,

Table 5. Values of A, AB and Fitness function of experimental record for U2R simulated network communications

U2R		
A	AB	Fitness
14	2	1.7070
3	56	3.8980
71	9	1.3850
129101	4	0.0715
25	16	2.0660

Table 6. Values of A, AB and Fitness function of experimental records for Probe simulated network communications

Probe		
A	AB	Fitness
84	915	3.8310
92	18	1.3400
9	4	1.6200
141802	11	0.0122
3	28	2.8370

Probe is obtained at the maximum value of AB. It means a maximum match of the two parts of the clause *if* (**condition**) *then* **act** for the current sequence of network characteristics (compared chromosome), and the sequence of chromosome characteristics from the database with an a priori defined category.

The minimum Fitness function value for the chromosome of the five categories is obtained at the maximum value of A. It means a maximum match of only the **condition** part of the clause *if* (**condition**) *then* **act** for the current sequence of network characteristics (compared chromosome) and the sequence of network characteristics (compared chromosome) from the database with an a priori defined category.

The experimental results for the DoS category show that the maximum value of the Fitness function is obtained at $AB = 226$, and not at the maximum value of $AB = 371$ for which the Fitness function is 2.982. This is due to the high value of $A = 349$, which significantly increases the *Penalty* on the value of the Fitness function.

6. CONCLUSIONS

The stages of the genetic algorithm used to detect and recognize attacks in computer networks are described and illustrated. The syntax and structure of

different rules (chromosomes), as well as the coding of network characteristics (genes) are discussed.

A Firas Alabsi's algorithm is applied to calculate the Fitness functions and evaluate network sequences, rules in the network intrusion detection.

The idea of applying genetic algorithms in systems for detecting and preventing intrusions into computer networks can be further developed in the direction of creating databases of communication characteristics that define different types of computer attacks. For the assessment of current and new intrusions, the use of integrated and complementary Fitness functions of suitability is envisaged, which will increase the accuracy of the chromosomes' selection and the effectiveness intrusion detection and prevention in computer networks.

REFERENCES

[1] I. Popchev and I. Radeva, Risk Analysis – an Instrument for Technology Selection, *Engineering Sciences* (2019) **LVI** (4), 5–20, ISSN: 1312-5702, e-ISSN: 2603-3542, DOI: 10.7546/EngSci.LVI.19.04.01.

[2] I. Popchev, I. Radeva, and I. Nikolova, Aspect of the Evolution from Risk Management to Enterprise Global Risk Management, *Engineering Sciences* (2021) **LVIII** (1) 16–30, ISSN: 1312-5702, e-ISSN: 2603-3542, DOI: 10.7546/EngSci.LVIII.21.01.02.

[3] D. Dimov and Y. Tsonev, Result Oriented Time Correlation between Security and Risk Assessments, and Individual Environment Compliance Framework, in: Proceedings of International Conference Europe Middle East & North Africa Information Systems and Technologies to Support Learning Smart Innovation, Systems and Technologies, 2019, pp. 373–383, ISBN: 978-3-030-03576-1, DOI:10.1007/978-3-030-03577-8_42.

[4] A. Toskova, V. Toskov, S. Stoyanov, and I. Popchev, Genetic Algorithm for Learning Humanoid Robot – *Comptes rendus de l'Academie Bulgare des Sciences* (2019) **72** (8) 1102–1110, ISSN: 1310-1331(Print), ISSN: 2367-5535 (Online), DOI: 10.7546/CRABS:2019.08.13.

[5] M. Sazzadul Hoque, Md. A. Mukit, and Md. A. N. Bikas, An Implementation of Intrusion Detection System Using Genetic Algorithm, *International Journal of Network Security & Its Applications* (*IJNSA*) (2012) **4** (2) 109–119, ISSN: 0974-9330, DOI: 10.5121/ijnsa.2012.4208.

[6] R. H. Gong, M. Zulkernine, and P. Abolmaesumi, A Software Implementation of a Genetic Algorithm Based Approach to Network Intrusion Detection, in: Proceedings of the Sixth International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing and First ACIS International Workshop on Self-Assembling Wireless Networks (SNPD/SAWN'05), 0-7695-2294-7/05, 2005.

[7] O. Chandrakar, R. Singh, and L. B. Barik, Application of Genetic Algorithm in Intrusion Detection System, *Control Theory and Informatics* (2014) **4** (1) 50–57, ISSN: 2224-5774, e-ISSN: 2225-0492.

[8] E. Bader and H. H. O. Nasereddin, Using Genetic Algorithm in Network Security, *IJRRAS* (2010) **5** (2) 148–154, ISSN: 2076-734X, e-ISSN: 2076-7366.

[9] B. Uppalaiah, K. Anand, B. Narsimha, S. Saraj, and T. Bharat, Genetic Algorithm Approach to Intrusion Detection System, *International Journal of Computer Science and Technology* (*IJCST*) (2012) **3** (1) 156–160, ISSN: 2229-4333, e-ISSN: 0976-8491.

[10] F. Alabsi and R. Naoum, Fitness Function for Genetic Algorithm Used in Intrusion Detection System, *International Journal of Applied Science and Technology* (2012) **2** (4) 129–134, ISSN: 2221-0997, e-ISSN: 2221-1004.

[11] Vr. Yewale, V. Jethani, and T. Ghorpade, Applying Genetic Algorithm to Intrusion Detection System, *International Journal of Science and Research* (*IJSR*) (2015) **4** (4) April 524–529, ISSN: 2319-7064.

[12] A. Goya, A Genetic Algorithm Based Network Intrusion Detection System (2007), https://pdfs.semanticscholar.org/6c8e/6708a1a737a9a5509de2fba46f8de1aff7e3.pdf.

[13] P. A. Diaz-Gomez and D. F. Hougen, Improved Off-line Intrusion Detection Using Genetic Algorithm, in: Proceedings of the Seventh International Conference on Enterprise Information Systems, 2005, http://www.cameron.edu/∼pdiaz-go/Art_ICEIS.pdf.

[14] S. Chowdhury, S. Kumar Das, and A. Das, Application of Genetic Algorithm in Communication Network Security, *International Journal of Innovative Research in Computer and Communication Engineering* (2015) **3** (1) 274–280, e-ISSN: 2320-9801, ISSN: 2320-9798.

[15] Ch. Sinclair and L. Pierce, An Application of Machine Learning to Network Intrusion Detection, in: Proceedings 15$^{\text{th}}$ Annual Computer Security Applications Conference (ACSAC'99), 1999, **6** (10), DOI: 10.1109/CSAC.1999.816048.

[16] R. Shanmugavadivu and N. Nagarajan, Network Intrusion Detection System Using Fuzzy Logic, *Indian Journal of Computer Science and Engineering* (*IJCSE*) (2011) **2** (1) 101–111, e-ISSN: 0976-5166, ISSN: 2231-3850.

[17] Messenger, R. Dove, Basic Genetic Algorithm Pattern for Use in Self-organizing Agile Security, in: Proceedings of IEEE International Carnahan Conference on Security Technology (ICCST), 2012, **15** (18).

[18] W. Li, Using Genetic Algorithm for Network Intrusion Detection, https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.89.3125&rep=rep1&type=pdf.

[19] V. Bapuji, R. N. Kumar, A. Goverdan, and S. Sharma, Soft Computing and Artificial Intelligence Techniques for Intrusion Detection System, *Networks and Complex Systems* (2012) **2** (4), ISSN: 2224-610X, e-ISSN: 2225-0603.

[20] Sh. Devi and R. Nagpal, Intrusion Detection System Using Genetic Algorithm – a Review, *International Journal of Computing & Business Research*, in: Proceedings of I-Society, Talwandi Sabo Bathinda (Punjab) (2012), ISSN (Online): 2229-6166.

[21] A. Lazarov. Y. Tsonev, and P. Petrova, Mathematical Modelling of Internet User's Behaviour in Penetrating Computer Systems, in: Proceedings of DIGILIENCE 2021, June 2021, the Springer series "Communications in Computer and Information Science" (in print).

Received February 07, 2022