Available online at www.sciencedirect.com
ScienceDirect
Neurocomputing 69 (2006) 2429–2433
NEUROCOMPUTING
www.elsevier.com/locate/neucom

Letters

Applying genetic algorithm for classifying anomalous TCP/IP packets

Taeshik Shon[a,*], Xeno Kovah[b], Jongsub Moon[c]

[a]*IP Lab, NGT Team, Telecommunication R&D Center, Samsung Electronics Co., LTD., Dong Suwon P.O. Box 105, 416, Maetan-3dong, Yeongton-gu, Suwon-si, Gyeonggi-do 442-600, Korea*
[b]*Department of Computer Science and Engineering, University of Minnesota, 4-192 EE/CSci Building, 200 Union Street SE, Minneapolis, MN 55455, USA*
[c]*Graduate School of Information Security, Center for Information Security Technologies, Korea University, Anamdong, Sungbukgu, Seoul 136-701, Korea*

Received 1 October 2005; received in revised form 31 January 2006; accepted 31 January 2006
Communicated by R.W. Newcomb
Available online 21 June 2006

Abstract

We present a method for applying genetic algorithms in order to feature selection of TCP/IP packets. The proposed scheme creates an appropriate polynomial equation with weighted coefficients as an objective function for fitness evaluation. The coefficients of the proposed polynomial equation represent the anomaly score of each field. After the evolutionary process is complete, the selected features are used for preprocessing TCP/IP packets before applying TCP/IP packets directly into anomalous detection. To verify the efficiency of the proposed method, various machine learning algorithms were tested with and without a genetic algorithm selecting the optimal fields.
© 2006 Elsevier B.V. All rights reserved.

Keywords: Genetic algorithm; Intrusion detection; Network security

1. Introduction

Network security issues have become one of the most important challenges facing modern information-based societies. Often intrusion detection models rely on signature-based mechanisms which are useful for finding attacks whose patterns are already known, but are vulnerable to modified and newly created attack patterns. Learning algorithms offer a solution to this problem. For these to operate efficiently, however, appropriate features of network data must be selected as learning inputs. In this paper we select features for our problem domain using a genetic algorithm (GA), a stochastic search method that mimics the natural process of biological evolution. We use this approach to obtain the minimum set of fields in a TCP/IP packet, which when used as input for machine learning, yields the highest detection rate of anomalous TCP/IP packets. This paper is organized as follows: a general description of the GA approach, the proposed application of this method, experimental results, and conclusions.

2. Approaching with a genetic algorithm (GA)

GA is a model to mimic the behaviour of evolution processes in nature. It is known to be an ideal technique to find a solution of optimization problems [1,2]. To apply this ‘‘process of evolution’’ to our problem domain, we have to decide on the following 3 steps: presenting and initialization individual genes, modelling evaluation function, and deciding a specific function of genetic operators and their parameters.

For the first step, we transform TCP/IP packets into binary gene strings. We convert each of the TCP and IP header fields into a one bit binary gene value, ‘0’ or ‘1’. ‘1’ means that the corresponding field exists and ‘0’ means no. The initial population consists of a set of randomly generated 24 bits strings including both 13 bits of IP fields and 11 bits of TCP fields. Additionally the total number of individuals in the population should be carefully considered. If the population size is too small, all gene chromosomes will have the same gene string value soon and the genetic model cannot generate new individuals. In contrast, if the population size is too large, the model spends more time to calculate gene strings, negatively affecting the overall effectiveness of the method. In this paper, we chose 100 as a population size.

*Corresponding author.
E-mail addresses: ts.shon@samsung.com, 743zh2k@korea.ac.kr (T. Shon).

0925-2312/$ - see front matter © 2006 Elsevier B.V. All rights reserved.
doi:10.1016/j.neucom.2006.01.023

Addressing the second process, we create a fitness function for evaluating individuals. The fitness function consists of an object function $f(X)$ and its transformation function $g(f(x))$:

$$F(X) = g(f(X)). \tag{1}$$

In Eq. (1), the objective function's values are converted into a measure of relative fitness by the fitness function $F(X)$ through the transformation function $g(x)$. To create our own objective function, we use the anomaly and communication scores shown in Table 1. The anomaly scores refer to MIT Lincoln Lab data sets, covert channels, and other anomaly attacks [3,4]. The scores increase in proportion to the frequency of a field being used for anomaly attacks. Communication scores are divided into three kinds of scores in accordance with their importance during a communication. 'S' fields have static values, "De" field values are dependent on connection status, and field values for "Dy" can change dynamically. We can derive a polynomial equation which has the above-mentioned considerations as coefficients. As the combination of an anomaly score and a communication score, the coefficients of this derived polynomial equation have the property of being weighted sums. Our objective function $f(X)$ consists of two polynomial functions $A(X)$ and $N(X)$ shown in (2):

$$f(X) = A(X) + N(X), \tag{2}$$

Table 1
TCP/IP anomaly and communication score

Index number	Name of coefficients	Anomaly score[a]	Communication score[b]
01	a_{01} (version)	0	S
02	a_{02} (header length)	0	De
03	a_{03} (type of service)	0	S
04	a_{04} (total length)	0	De
05	a_{05} (identification)	2	Dy
06	a_{06} (flags)	5	Dy
07	a_{07} (fragment offset)	5	Dy
08	a_{08} (time to live)	1	Dy
09	a_{09} (protocol)	1	S
10	a_{10} (header checksum)	0	De
11	a_{11} (source address)	2	S
12	a_{12} (destination address)	1	S
13	a_{13} (options)	1	S
14	a_{14} (source port)	1	S
15	a_{15} (destination port)	1	S
16	a_{16} (sequence number)	2	Dy
17	a_{17} (acknowledge number)	2	Dy
18	a_{18} (offset)	1	Dy
19	a_{19} (reserved)	1	S
20	a_{20} (flags)	2	Dy
21	a_{21} (window)	0	S
22	a_{22} (checksum)	0	De
23	a_{23} (urgent pointer)	1	S
24	a_{24} (options)	1	S

[a]By anomaly analysis in [3,4].
[b]S: static, De: dependent, Dy: dynamic.

$A(X)$ is our anomaly scoring function and $N(X)$ is our communication scoring function. $X(x)$ is an individual with 24 attributes and x_i is one of the individual's attributes with an attribute's value a_i (as illustrated in Table 1). To prevent generating too many features from (2)'s fitness function, a bias term μ is used as follows:

$$\begin{aligned} f'(X(x)) &= f(X(x)) - \mu \\ &= A(X(x)) + N(X(x)) - \mu, \end{aligned} \tag{3}$$

where, μ is the bias term of new objective function $f'(X(x))$, and the boundary is $0 < \mu < \mathrm{Max}(f(X))$. In case of $A(X(x))$, we can derive the proper equation as follows:

$$\begin{aligned} A(X) &= A(X(x)) \\ &= a_i x_i + \cdots + a_2 x_2 + a_1 x_1, \quad i = \{1, \ldots, 24\}, \end{aligned} \tag{4}$$

where, x_i would denote the ith attribute. $\{a_i, \ldots, a_2, a_1\}$ is a set of coefficients in the polynomial equation (as illustrated in Table 1) and each coefficients represents anomaly scores. From Eq. (4), we use the bias term to satisfy condition (5). Thus, we can choose a reasonable number of features without over-fitting and we can derive the new anomaly scoring function (6) with the bias term μ_A as follows:

$$\begin{aligned} A(X) &= a_i x_i + \cdots + a_2 x_2 + a_1 x_1 \\ &< \mathrm{Max}(A(X)), \end{aligned} \tag{5}$$

$$\begin{aligned} & A'(X) = (a_i x_i + \cdots + a_2 x_2 + a_1 x_1) - \mu_A, \\ & \quad 0 < \mu_A < \mathrm{Max}(A(X)) \\ & \quad \therefore 0 < A'(X) < \mathrm{Max}(A(X)). \end{aligned} \tag{6}$$

For $N(X(x))$, we also develop an appropriate function with the same derivation as Eq. (6).

$$\begin{aligned} N(X) &= N(X(x)), \alpha = 1, \beta = 2, \gamma = 3, i = \{1, \ldots, 24\} \\ &= a_i x_i + \cdots + a_2 x_2 + a_1 x_1 \\ &= (a_1 x_1 + a_3 x_3 + a_9 x_9 + a_{11} x_{11} + a_{12} x_{12} + a_{13} x_{13} + a_{14} x_{14} \\ &\quad + a_{15} x_{15} + a_{19} x_{19} + a_{21} x_{21} + a_{23} x_{23} + a_{24} x_{24}) \\ &\quad + (a_2 x_2 + a_4 x_4 + a_{10} x_{10} + a_{22} x_{22}) + (a_5 x_5 + a_6 x_6 \\ &\quad + a_7 x_7 + a_8 x_8 + a_{16} x_{16} + a_{17} x_{17} + a_{18} x_{18} + a_{20} x_{20}) \\ &= \alpha(x_1 + x_3 + x_9 + x_{11} + x_{12} + x_{13} + x_{14} + x_{15} + x_{19} \\ &\quad + x_{21} + x_{23} + x_{24}) + \beta(x_2 + x_4 + x_{10} + x_{22}) \\ &\quad + \gamma(x_5 + x_6 + x_7 + x_8 + x_{16} + x_{17} + x_{18} + x_{20}), \end{aligned} \tag{7}$$

where N is a set of communication scores, and the coefficients α, β, γ mean static (S), dependent (De) and dynamic (Dy), respectively (as illustrated in Table 1). In Eq. (7), we give the bias term by the same method as (5) and (6) as follows:

$$\begin{aligned} N(X) &= \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) \\ &< \mathrm{Max}(N(X)), \end{aligned} \tag{8}$$

$$\begin{aligned} & N'(X) = \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - \mu_N, 0 < \mu_n < \mathrm{Max}(N(X)) \\ & \quad \therefore 0 < N'(X) < \mathrm{Max}(N(X)), \end{aligned} \tag{9}$$

where $x_\alpha, x_\beta, x_\gamma$ are a set of elements with the coefficient α, β, γ, respectively. From Eqs. (6) and (9), we can derive our entire objective Eq. (10) as follows:

$$\begin{aligned} f'(X(x)) &= A'(X) + N'(X) \\ &= (a_i x_i + \cdots + a_2 x_2 + a_1 x_1) - \mu_A + \alpha(x_\alpha) + \beta(x_\beta) \\ &\quad + \gamma(x_\gamma) - \mu_N = (a_i x_i + \cdots + a_2 x_2 + a_1 x_1) + \alpha(x_\alpha) \\ &\quad + \beta(x_\beta) + \gamma(x_\gamma) - (\mu_A + \mu_N) = (a_i x_i + \cdots + a_2 x_2 \\ &\quad + a_1 x_1) + \alpha(x_\alpha) + \beta(x_\beta) + \gamma(x_\gamma) - \mu, \\ &\therefore 0 < f'(X(x)) < \mathrm{Max}(f(X(x))). \end{aligned} \tag{10}$$

The last step for genetic modelling is to define a specific function of genetic operators and their related parameters. After finishing a genetic procedure, to examine the effective of GA for TCP/IP anomaly classification we use three well-known learning algorithms, such as back propagation network (BPN), Radial Basis Function (RBF), *k*-Nearest Neighbour (*k*-NN), and Support Vector Machine (SVM) [3].

3. Experimental methods and results

In order to find reasonable genetic parameters, we make preliminary tests using the typical genetic parameter values mentioned in the literature [5]. Table 2 describes the four preliminary test parameters.

Fig. 1 shows four graphs of GA feature selection with the fitness function (10) according to Table 2 parameters. In Cases #1 and #2, the resultant graph seems to have rapidly converging values because of too low reproduction rate, and too high crossover and mutation rate respectively. In Case #3, the graph seems to show constant values because of too high reproduction rate. Finally, the fourth

Table 2
Preliminary test parameters of GA

	Size of population	Reproduction rate (Pr)	Crossover rate (Pc)	Mutation rate (Pm)	Final fitness value
Case #1	100	0.100	0.600	0.001	11.72
Case #2	100	0.900	0.900	0.300	14.76
Case #3	100	0.900	0.600	0.100	25.98
Case #4	100	0.600	0.500	0.001	19.12

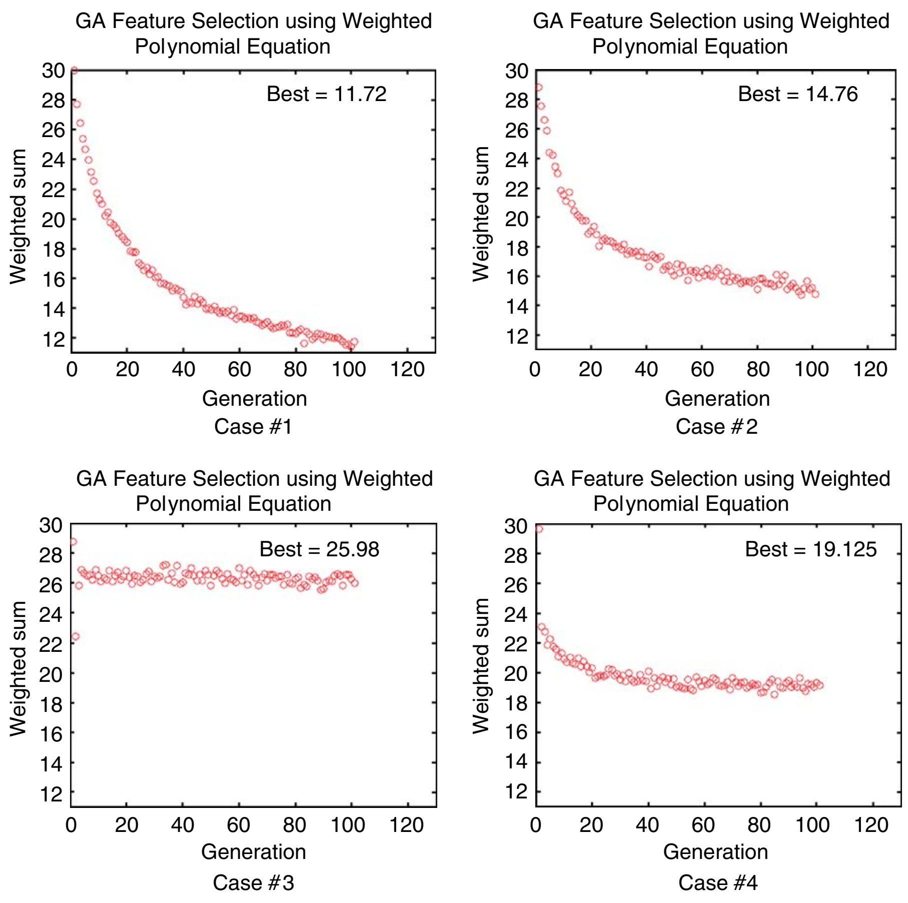

Fig. 1. Evolutionary process according to preliminary test parameters.

Table 3
GA field selection results of preliminary Test #4

Generation units	Number of selected fields	Number of selected fields	CR (%)	FP (%)	FN (%)	PT (ms)
01–15	19	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,21,22,23,24	96.68	1.79	7.00	2.27
16–30	15	2,5,6,7,8,9,10,11,12,16,17,20,21,23,24	95.00	0.17	16.66	1.90
31–45	15	2,5,6,7,8,9,10,11,12,16,17,20,21,23,24	95.00	0.17	16.66	1.90
46–60	18	1,2,5,6,7,8,9,10,11,12,13,14,17,19,20,22,23,24	95.12	0.00	16.66	1.84
61–75	17	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21	73.17	0.00	91.60	0.30
76–90	17	2,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21	73.17	0.00	91.60	0.30
91–100	15	3,5,6,7,9,12,13,16,17,18,19,21,22,23,24	97.56	0.00	8.33	1.74

*CR: correction rate, FP: false positive, FN: false negative, PT: processing time (SVM learning time with 2000 packets).

Table 4
Our experiment results using GA feature selection

	Parameters	Detection rate (%)		False positive (%)		False negative (%)	
		Test 1	Test 2	Test 1	Test 2	Test 1	Test 2
Neural networks	BPN	85.5	79.1	12.1	15.8	12.4	15.1
	RBF	96.3	88.4	2.7	10.1	11.0	11.5
k-NN	$k = 10$	92.2	90.5	7.1	18.2	10.7	11.3
	$k = 15$	94.8	91.1	4.6	17.8	10.6	11.1
SVM	RBF kernel	98.6	93.6	2.5	11.2	11.0	15.2

graph of Case #4 seems to be converging with appropriate values. The detailed results of Case #4 are described in Table 3. Through the above experiment results using soft margin SVM learning, we knew the final generation was well-optimized. The generation 91–100 showed the best correction rate and relatively fast processing time. Moreover, by comparing the generation 16–30 with the 46–60 generation, fewer fields do not always guarantee faster processing because the processing time is also dependant on the value of the fields.

For the real verification of this research, we need to preprocess real-world datasets by the resultant fields of Table 3. The real-world data sets were prepared 100,000 packets for normal and 10,000 packets for abnormal. In normal data sets, we gathered training data from the Information Systems Technology Group (IST) of MIT Lincoln Laboratory [4]. To evaluate preprocessed packets, we applied to three machine learning algorithms such as BPN [6], RBF [7], *k*-NN [8], and SVM [2]. For each of the learning algorithms, we set a variety of parameters. For BPN, the learning rate is 0.02 and maximum error rate is 0.2. Also, we used the number of 1000 epoch less. For RBF, we use the PNN parameters of the Matlab package such as "spread"—spread of radial basis functions, default = 0.1. A PNN can be worked as a kind of radial basis network suitable for classification problems. For the *k*-NN, the values of *k* were set to 10,000 as the training data set, $k = 10$ and 15. *k* can be adjusted to the overall size of the data. Finally, in case of SVM, we used the SVM light package [9] for supervised SVM classification with RBF kernel. Each test means Test 1 with GA feature selection and Test 2 without GA feature selection.

Using the above three algorithms, we obtained the resultant data of Table 4. The correct percentage in detection was 85.5% in BPN, 96.3% in RBF, 92.2% and 94.8% in *k*-NN for $k = 10$ and 15, 98.65% in SVM respectively. By these five experiments, we can see that the overall Test 1's accuracy with GA is superior to that of Test 2's accuracy without. The average of Test 1 experiments is 92.2% of correction percentage and 6.63% of false positive; on the other hand, the average of Test 2 experiments is 87.23% of detection rate and 10.48% of false positive. Moreover, SVM presents the best performance. The most meaningful point from these experiments is to show the efficiency and performance of classifying and feature selecting with the genetic process.

4. Conclusions

The most attractive idea in this letter is to use GA for selecting optimum fields from a TCP/IP header for anomaly detection, and verify the proposed method using a variety of learning algorithms such as BPN, RBF, *k*-NN, and SVM. We proposed a weighted polynomial equation as the objective function of evolutionary process. The coefficient values of the proposed equation are an anomalous score of each TCP/IP fields in accordance with various anomaly attacks. After using GA with the proposed objective function, we found some of TCP/IP fields to be the selected optimal subset for our problem domain. The data in Table 3 showing increased detection rates with the addition of GA feature subset selection leads us to believe that this technique is of great practical value. In the future, we hope to make the genetic model more elaborate and verify it under more realistic environments.

Acknowledgement

This work was supported by the Ministry of Information Communications, Korea, under the Information Technology Research Centre Support Program supervised by the IITA.

References

[1] J. Holland, Adaptation in Natural and Artificial Systems, The University of Michigan Press, Ann Arbor, MI, 1995.
[2] X.-Z. Wang, Q. He, D.-G. Chen, D. Yeung, A genetic algorithm for solving the inverse problem of support vector machines, Neurocomputing 68 (2005) 225–238.
[3] K. Ahsan, D. Kundur, Practical data hiding in TCP/IP, in: Proceedings of the Workshop on Multimedia Security at ACM Multimedia, 2002, French Riviera, France, 2002, pp. 7–15.
[4] Lincoln Laboratory, MIT, "DARPA Intrusion Detection Evaluation," http://www.ll.mit.edu/IST/ideval/index.html
[5] M. Mitchell, An Introduction to Genetic Algorithms, MIT Press, Cambridge, MA.
[6] P.D. Wasserman, Advanced Methods in Neural Computing, Van Nostrand Reinhold, New York, 1993, pp. 35–55.
[7] M.I. Jordan, C.M. Bishop, Neural networks, ACM Comput. Surv. (CSUR) 28 (1) (1996) 73–75.
[8] S. Belkasim, M. Shridhar, M. Ahmadi, Pattern classification using an efficient KNNR, Pattern Recognition 25 (10) (1992) 1269–1274.
[9] SVM light package, http://svmlight.joachims.org/

Taeshik Shon is a senior engineer of IP Lab at TN R&D Center of Samsung Electronic Co., Ltd. He received his Ph.D. degree in Information Security at Korea University. He got a BK21 scholarship from the Ministry of Education in Korea, from 2000 to 2002. While he was working toward his Ph.D. degree at Korea University, he got a KOSEF scholarship to be a research scholar in Digital Technology Center at the University of Minnesota, USA, from February 2004 to February 2005. He was awarded the Gold Prize at the Sixth Information Security Best Paper Award of Korea Information Security Agency (KISA) in 2003, got an IFIP TC6 Student Grant at IntellComm 2004 and was awarded the Honorable Prize at the 24th Student Best Paper Award of Microsoft-KISS in 2005. His research interests include computer networks, network security, machine learning and pattern recognition.

Xeno Kovah is a master course student of Information Networking Institute at the Carnegie Mellon University. He received his B.S. degree at the department of computer science at the University of Minnesota (2005). He is interested in information security.

Jongsub Moon received a B.S. and M.S degree in Computer Science from Seoul National University, Seoul, Korea, in 1976 and 1980, respectively. He received his Ph.D. degree in Computer Science from the Institute of Illinois Technology, Illinois, USA, in 1985. He has become a professor in the Department of Computer Science and Information Engineering and Graduate School of Information Security at Korea University, Korea since February 1991. His areas of research include artificial neural network and communication, security.