Neuronet Artificial Immune System for Malicious Code Detection

Name Surname

University
adress
email@.com

Abstract

In this paper we present the intelligent adaptive self-learning and self-organizing system for malicious code detection based on integration of both methods: the Artificial Immune Systems and the Artificial Neural Networks. Such system is functioning according to basic principles of the artificial immune system where immune detectors have neuronet architecture and detect a malicious pattern by analyzing the structure of the executable code. The system is able to adapt to the continually changeable computer environment and detect not only known but unknown malicious code. The paper zeroed in on neuronet structure of immune detectors as main element for malware detection.

1 INTRODUCTION

At present time the most real antivirus software based on signature analysis. Signature-based approach have acceptable detection rate for known virus and relatively low false positives. Unfortunately the ability of signature-based system to detect new viruses is extremely poor.

The artificial immune system (AIS) is novel approach inspired by biological immune systems. It can be defined like computational system based on ideas biological immune system. There exist different models of AIS for malicious code detection (L.N. de Castro 2002, S. Hofmeyr 2000, Janeway 1993). Unfortunately there are some problems to use this approach for malware detection. As a rule AIS models use of binary or real strings structure of detectors. In this case it is difficult to train such detectors for qualitative malware detection. As a result such systems have high computational complexity and nonwell ability for novel malware detection. To overcome these problems we propose the neural network structure of immune detectors in AIS for malicious code detection.

The key idea of this paper is integration of advantages of artificial immune systems and neural networks for creating intelligent adaptive self organizing system for malicious code detection and recognition. Such system is characterized by ability of novel malware detection and low false positive and false negative rates.

The paper is structured as follows. Section 2 describes the immune detector as black box. Basic algorithms of learning and function of detector are given. Section 3 describes the choice of the type of artificial neural network as a basis for immune detectors. In section 4 the process of file scanning by neuronet immune detector is described. . In section 5 the experimental results of malicious code detection are given. Section 6 concludes this paper and point out our future work.

2 THE BLACK BOX OF IMMUNE DETECTOR

The AIS is highly parallel system consists of several important mechanisms (figure 1).

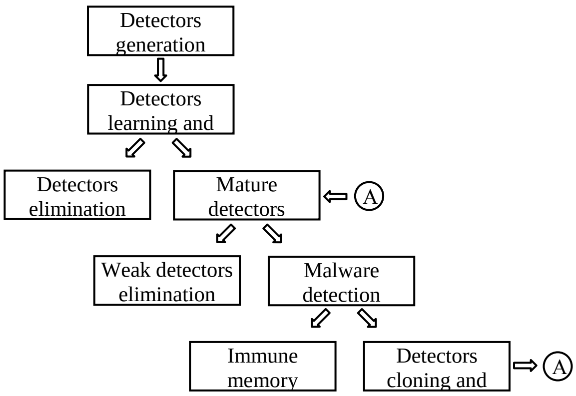

Figure 1: Model of the AIS.

The proposed AIS consists of the following blocks: block of detectors generation, block of detectors training, block of detectors selection, block of detectors elimination, block of infected files detection, block of detectors cloning and mutation, block of immune memory creation. The immune detectors play a main role in malicious code detection and the architecture of detectors is significant for successful detection. The computer system is changeable with time system. The new software is installed and uninstalled continually. Thereby the security system should correct identify legitimate software and detect malicious code both already known and novel. All this impose strong requirements on the immune detectors.

By way of immune detector we use the artificial neural network.

Initially let's represent neuronet immune detector as a black box with *n*-inputs and two outputs (Figure 2).

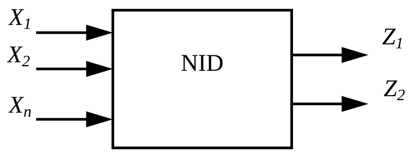

Figure 2: The neuronet immune detector.

The outputs of detector after presentation of all checking data in accordance can be obtained with the following expressions:

$$Z_1 = \begin{cases} 1, & if\ \ legitimate\ \ file \\ 0, & otherwise. \end{cases}$$
$$Z_2 = \begin{cases} 1, & if\ \ malicious\ \ file \\ 0, & otherwise. \end{cases} \qquad (1)$$

The training data set consist of legitimate and malicious files. Of course, the immune detectors will be more diverse, if the more various files are presented in the training data set. It is desirable to have also representatives of all types of malware such as worms, Trojan programs, macro viruses etc.

The neural network underlying the immune detector is trained by supervisor rule (Haykin 1999). Figure 3 illustrates the input samples for neural network training.

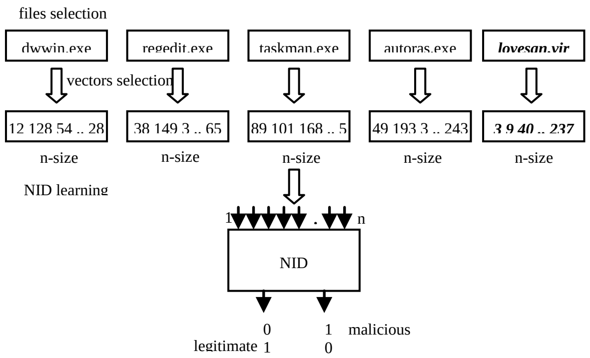

Figure 3: The mechanism of neural network training.

For generation legitimate samples using files selected from utilities of operating system Microsoft Windows as it is shown in Figure 3 (*dwwin.exe, regedit.exe, taskman.exe, autoras.exe*). The computer viruses are used for generation malicious samples, for instance *lovesan.vir* in figure.

Let T is set of legitimate files and F is set of malicious files. These sets are used for forming the learning sample for training of *i*-th detector (formula 2).

Accordingly, the set of desired output samples can be written by formula 3.

The desired output samples for *i* th detector are formed by formula 4.

$$X_i = \begin{bmatrix} X_i^1 \\ X_i^2 \\ \dots \\ X_i^L \end{bmatrix} = \begin{bmatrix} X_{i1}^1 & X_{i2}^1 & \dots & X_{in}^1 \\ X_{i1}^2 & X_{i2}^2 & \dots & X_{in}^2 \\ \dots & & & \\ X_{i1}^L & X_{i2}^L & \dots & X_{in}^L \end{bmatrix} \qquad (2)$$

$$l_i = \begin{bmatrix} l_i^1 \\ l_i^2 \\ \dots \\ l_i^L \end{bmatrix} = \begin{bmatrix} l_{i1}^1 & l_{i2}^1 \\ l_{i1}^2 & l_{i2}^2 \\ \dots & \\ l_{i1}^L & l_{i2}^L \end{bmatrix} \qquad (3)$$

where L is dimension of training set.

$$l_{i1}^k = \begin{cases} 1, & if\ \ X_i^k \in T \\ 0, & otherwise. \end{cases}$$
$$l_{i2}^k = \begin{cases} 1, & if\ \ X_i^k \in F \\ 0, & oteherwise. \end{cases} \qquad (4)$$

The detectors are trained to classify normal and abnormal patterns. The purpose of the training of each detector is to minimize the mean square error. The mean square error for *i*-th detector is defined as:

$$E_i = \frac{1}{2}\sum_{k=1}^{L}\sum_{j=1}^{2}(Z_{ij}^k - l_{ij}^k)^2, \qquad (5)$$

where Z_{ij}^k is *j*-th output unit of *i*-th detector for *k*-th pattern.

The mean square error is characterized the detector fitness to recognize malicious code. The fewer value of mean square error means the better fitness of the detector. Therefore the square error evaluates the quality of detector and can be used for selection of the best detectors.

The set of trained neural networks forms population of immune detectors which circulate in computer system and perform detection of malicious code. Each detector has an assigned lifetime that is decreased by each iteration of algorithm presented below. If the detector reaches the maturity age, it will be eliminated. By using of the various files for neural network training and random process of generation of input vectors the population of difference detectors is permits.

The neuronet architecture of AIS allow to construct such adaptive security system, which able to detect novel virus patterns for which no signature exists.

The procedure of building and performance of neuronet immune system can be represented as follows:

1. The generation of an initial population of detectors. It should be noted that each detector represents the neural network with random weights:

$$D = \left\{ D_i, \quad i = \overline{1, r} \right\}, \tag{6}$$

where D_i is i-th neural immune detector, r is the number of detectors.

2. The training of the neuronet immune detectors. Training data set are generated by random way from legitimate and malicious files or their signatures. The desired output units are obtained in accordance with equation 4. After the training certain amount of detectors are obtained, which are used in the testing stage.

3. The selection of the best neural detectors by test data set. The goal of this process is to eliminate bad (unsuitable) detectors, which have insufficient ability to training and false positive rate. Each detector is verified by test data set, which consists of legitimate files. As a result for each detector is determined mean square error E_i in accordance with equation 5. Detectors would be selected with zero mean square error:

$$D_i = \begin{cases} 0, & if \ E_i \neq 0 \\ D_i, & otherwise. \end{cases} \tag{7}$$

where 0 characterizes operation of elimination of detector.

4. Each detector is provided with lifetime and scans files chosen by random from all files of computer system.

5. The scanning of chosen file by detector. As a result output values of detectors Z_{i1}, Z_{i2}, where $i=1,r$, are defined.

6. If i-th detector does not detect virus in scanning file, i.e. $Z_{i1}=1$ и $Z_{i2}=0$, then it choose next file for inspection. If the lifetime of a detector is ended, it is eliminated from the detectors set and new detector is created.

7. If i-th detector detect virus in file, i.e. $Z_{i1}=0$ и $Z_{i2}=1$, then it activates alarm. Then cloning and mutation of given detector is performed. As a result the set of clones are generated and each clone is trained by using detected infected file (mutation). Finally we can get the set of clones, which are aimed to detect given virus

$$D_i = (D_{i1}, D_{i2}, \dots, D_{in}). \tag{8}$$

8. The best clone detectors selection by fitness function which characterizes the level of detection of malicious cod. The mean square error for each clone is calculated, using detected virus file. IF $E_{ij} > E_i$, тhen detector has passed selection. Here E_{ij} – mean square error for j-th clone of i-th detector.

9. The set of clones scanning the file system with purpose of given malicious code detection and elimination.

10. The creation of immune memory detector. The best neuronet immune detector is defined, which have shown the perfect results during detection of given computer virus. Detectors of immune memory live in system long time and provide the protection against repeated infection.

Figure 4 shows the performance of the immune detectors based on neural network architecture.

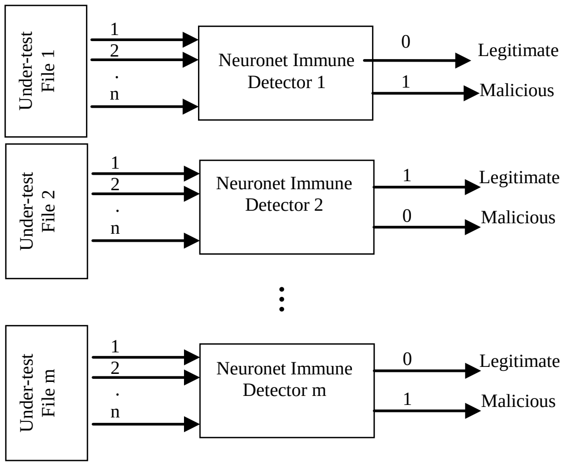

Figure 4: The function of the set of neuronet immune detectors.

Let's note the basic differences of the proposed and known algorithms. In our case each immune detector is completely independent object, i.e. itself chooses the scanning area. For this purpose it receives the list of files stored on a hard disk and randomly chooses a file from this list. After checking of file detector selects by random next file from the existing list. The procedure is continuing until the detector does not detect malicious code or lifetime of the detector is ends. The key advantage of proposed neuronet AIS is the ability of novel malicious code detection, for which no signature exists.

3 THE ARCHITECTURE OF NEURONET IMMUNE DETECTOR

The choice of the structure of immune detectors directly influences on the classification quality of unknown patterns and malware detection. The AIS is characterized by continuous evolution of immune detectors which getting through several stages during lifecycle (figure 1). Untrained detectors incapable of correct classification of legitimate and malicious code and require the learning process. The complexity of learning process is in direct proportion with the size of learning sample. Therefore we should choose the type of artificial neural network that characterized by minimal size of learning sample. As a basis of the neuronet immune detectors we chose backpropagation neural network (Haykin 1999). In contrast to another types of neural networks such as multi-layer perceptron and multi-recurrent neural networks, backpropagation neural network is characterized by minimal size of learning sample (Haykin 1999).

Figure 5 shows the structure of neuronet immune detector (NID) which consists of three layers of neurons and arbiter.

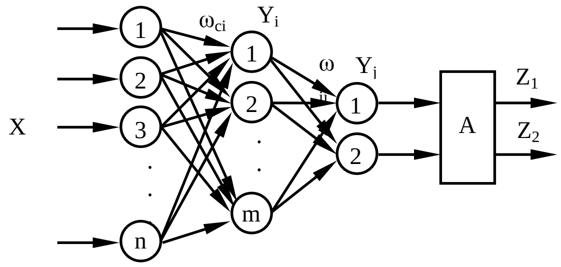

Figure 5: The structure of NID.

The first layer of neurons is input layer and distributes input signals to the neurons of hidden layer. The number of neurons of input layers is equal to size of sliding window *n* (NID scans files by a sliding window method).

The second layer consists of Kohonen neurons (Kohonen 1982) which using the competitive learning rule (Kohonen 1982) and functions by "winner-takes-all" (Kohonen 1982) scheme. Kohonen layer performs clusterization of the input patterns and clusters of different patterns are formed where each cluster corresponds with one's own neuron. The number of neurons equals *m* and

$$m = p + r, \tag{9}$$

where p – the number of the first neurons which corresponding to legitimate files; r – the number of last neurons, their activity characterizes the class of malicious files.

The ratio of p to r should be multiple of 4 to 1 (for example $p = 8, r = 2$)

$$\frac{p}{r} = \frac{4}{1}. \tag{10}$$

This ratio related to the algorithm of forming of learning sample, had received from experiments and showed best results.

The third layer consist of two linear neurons which using linear activation function [activation function]. The activity of the first neuron is characterizes the "clear" legitimate pattern while activation of the second neuron points to malicious pattern. In general case the output value of j-th neuron of third layer described by equation 11.

If the winning neuron of Kohonen layer has number k then the output value of j-th neuron is calculated by formula 12.

$$Y_j = \sum_{i=1}^{m} \omega_{ij} \cdot Y_i, \tag{11}$$

where ω_{ij} – weighting coefficient between i-th neuron of Kohonen layer and j-th neuron of linear layer; m - the number of neurons of Kohonen layer.

$$Y_j = \omega_{kj} \cdot Y_k. \tag{12}$$

Eventually, for the correct mapping of input patterns into two classes the matrix of weights of third layer should form as follows:

$$\begin{aligned} &\omega_{kj} = 1 \text{ if } k = 1,2,..,p \text{ and } j=1, \text{ or } k=p+1,..,r \text{ and } j=2 \\ &\omega_{kj} = 0 \text{ if } k = 1,2,..,p \text{ and } j=2, \text{ or } k=p+1,..,r \text{ and } j=1 \end{aligned} \tag{13}$$

For example, if $p = 8$ and $r = 2$ then the matrix of weights looks like that

$$W^T = \begin{bmatrix} 1111111100 \\ 0000000011 \end{bmatrix} \tag{14}$$

The arbiter performs the procedure of making of final decision about the class (legitimate or malicious) of the under test file. The output values of detector are formed after analysis of all windows of the under test file and described by equation 15.

$$\begin{aligned} Z_1 &= \begin{cases} 1, & \text{if legitimate file} \\ 0, & \text{otherwise.} \end{cases} \\ Z_2 &= \begin{cases} 1, & \text{if malicious file} \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{15}$$

4 THE PROCESS OF FILE SCANNING BY NEURONET IMMUNE DETECTOR

The learned and selected NID scans the file memory and performs the function of malicious code detection. It should be noted that every NID is an independent autonomous agent which chose the target files for scanning by oneself.

The process of file scanning is implements by sliding window method. The size of window can be varying between 128 and 512. These values are taken from the traditional method of malware detection based on signature implementation where similar sizes of signatures guaranties exactly malicious code detection. The NID pass the file and makes decision about maliciousness of the code.

The algorithm of function of the NID can be described as set of the next steps:

1. The initial values of neurons in third layer are settled

$$\overline{Y_1}(k-1)=0,$$
$$\overline{Y_2}(k-1)=0. \quad (16)$$

2. The input patterns k ($k = 1, L$, where L – the set of pattern from scanning file) from under-test file are given sequentially by the sliding window and the following values are calculates:

a. Euclidean distance between input pattern and weights of Kohonen layer neurons;

b. the winning neuron with index k

$$D_k = \min_j D_j. \quad (17)$$

c. the values of neurons in third layer (by equation 12);
d. the quantity of legitimate and malicious fragments of under-test file

$$\overline{Y_1}(k)=\overline{Y_1}(k-1)+Y_1^k,$$
$$\overline{Y_2}(k)=\overline{Y_2}(k-1)+Y_2^k. \quad (18)$$

3. The belonging probability of under-test file to legitimate or malicious class is calculates

$$P_T=\frac{\overline{Y_1}}{L}\cdot 100\%,$$
$$P_F=1-P_T=\frac{\overline{Y_2}}{L}\cdot 100\%, \quad (19)$$

where P_T – the probability of legitimate file; P_F – the probability of malware.

5. On the grounds of calculated probabilities the decision of belonging of under-test file to one of the two classes is accept correspond to next equation

$$Z_1=\begin{cases}1, & if\ P_T>80\%\\ 0, & otherwise.\end{cases}$$
$$Z_2=\begin{cases}1, & if\ P_F>20\%\\ 0, & otherwise.\end{cases} \quad (20)$$

This value of the threshold is conditioned by the specific of the learning sample forming and shows the best results [our paper].

Thus the space of output values of the arbiter can be represents in a tabular form (table 1).

Table 1: The arbiter output values space.

Z_1	Z_2	Class
1	0	Clear
0	1	Malicious
0	0	Undefined

6. If $Z_1 = 0$ and $Z_2 = 0$ then another NID for scanning this file is assign.

Let's consider the process of function "mature" NID by example where two files *write.exe* and *Virus.Win32.VB.d* are scanning. The NID has next architecture: $n = 256$, $m = 10$, $b = 2$, where n – the number of input neurons; m – the number of Kohonen neurons; b – the number of output neurons. For the learning of the NID the next four legitimate file – *forcedos.exe, rspndr.exe, share.exe, lpq.exe* and one malicious file – *Net-Worm.Win32.Bozori.k* are used. For checking of *write.exe* the NID should forms the next quantity of windows (NID scans files by a sliding window method)

$$L = S - n + 1 = 2402 - 256 + 1 = 2147, \quad (21)$$

where S – file size; n – window size.

In the result the NID classified *1084* fragments to legitimate class and *343* fragments to malicious class and probabilities of legitimate and malicious is

$$P_T=\frac{\overline{Y_1}}{L}\cdot 100\%=\frac{1804}{2147}\cdot 100\%=84\%,$$
$$P_F=1-P_T=\frac{\overline{Y_2}}{L}\cdot 100\%=\frac{343}{2147}\cdot 100\%=16\%. \quad (22)$$

Then the arbiter makes a decision about maliciousness of the file:

$$Z_1 = 1, \text{ since } P_T > 80\%,$$
$$Z_2 = 0, \text{ since } P_F < 20\%. \quad (23)$$

Write.exe is thereby legitimate.

In case of *Virus.Win32.VB.d* the number of windows is equal to

$$L = S - n + 1 = 33330 - 256 + 1 = 33075, \quad (24)$$

In the result the NID classified *21499* fragments to legitimate class and *11576* fragments to malicious class and probabilities of legitimate and malicious is

$$P_T=\frac{\overline{Y_1}}{L}\cdot 100\%=\frac{21499}{33075}\cdot 100\%=65\%,$$
$$P_F=1-P_T=\frac{\overline{Y_2}}{L}\cdot 100\%=\frac{11576}{33075}\cdot 100\%=35\%. \quad (25)$$

Then the arbiter makes a decision about maliciousness of the file:

$$Z_1 = 0, \text{ since } P_T < 80\%,$$
$$Z_2 = 1, \text{ since } P_F > 20\%. \quad (26)$$

Virus.Win32.VB.d is thereby malicious.

The examined example shows that the developed algorithms of learning, selection and function of the NID allow to detect new malicious code.

5 EXPERIMENTAL RESULTS

Experimental conditions:

1) The set *Cl* of legitimate files and the set *Ml* of malicious files are formed. For legitimate set 50 different executive system utilities of Microsoft Windows are chosen. There are, for example: *regedit32.exe, control.exe, diskcopy.exe, write.exe* etc. For malicious files 50 different malware consisting of different type of malicious software such as Trojans, worms and viruses are chosen. For example: *Email-Worm.Win32.NetSky.q, Net-Worm.Win32.Lovesan, Trojan-Proxy.Win32.Agent.x, Virus.Win32.Hidrag.d* etc.

2) The neuronet immune detectors with different parameters (number of input neurons – *n*, number of hidden neurons – *m*, learning sample size – *n*k*) was generated.

4) The learning of neuronet immune detectors. The learning sample for one detector is formed by next algorithm: 4 files from clear set and 1 file from malicious set are chosen by random. Then from every file *k* fragments (where *k* = 5 or 10) by *n* (where *n* = 128, 256 and 512) length are chosen.

5) From set of trained (mature) detectors the best detectors from every kind of parameters are chosen, which check all files from clear and malicious sets.

The results of files scanning by best neuronet immune detectors are showed in table 2.

Table 2: The results of malicious code detection.

Detectors, n/m/k	False alarm	Virus detection
128/10/5	0	6
256/10/5	1	17
512/10/5	1	22
128/10/10	0	4
256/10/10	0	5
512/10/10	0	2
128/5/5	2	3
256/5/5	1	20
512/5/5	0	6
128/5/10	2	1
256/5/10	0	4
512/5/10	1	4

As can be seen some neuronet immune detectors make a false alarm. False alarm occurred when neuronet immune detector classify file from clear set as malicious. For elimination of such unsuitable detectors the stage of selection of detectors is using.

Table 3 shows the process of malware detection by different neuronet immune detectors. As can be seen one neuronet immune detector is capable to detect several different malware.

Table 3: The process of malicious code detection.

Malware	D1	D2	D3	D4	D5	D6	D7
Worm.Brontok.q	+					+	+
Worm.NetSky.q	+						
Worm.Rays				+			
Worm.Bozori.a	+						
Worm.Bozori.k	+						
Packed.Tibs	+						
Trojan.Dialer.eb						+	
Trojan.Service.bl					+		
Trojan.Service.gi		+	+	+		+	+
Trojan.Small.dde	+						
Trojan.Lager.d					+	+	
Virus.Bee		+	+	+			
Virus.Neshta.a		+	+	+		+	+
Virus.VB.d		+	+	+		+	+

6 CONCLUSIONS

In this paper we present the neuronet architecture of immune detectors of artificial immune system for malware detection. The neural structure allows detectors to detect different malicious software. The feature of artificial immune system with neural architecture consists in capability of novel malicious code detection. Applications of the neural networks approach for immune detectors generation allow creating the powerful detectors. Applying of the AIS for malicious code detection will expand the potentialities of existing antivirus software and will increase level of computer systems security.

Reference

Haykin, S. *Neural Networks: A Comprehensive Foundation.* Prentice-Hall, 1999.

Janeway, C.A. "How the Immune System Recognizers Invaders." *Scientific American. Vol. 269, no. 3,* 1993: pp. 72–79.

Kohonen, T. " Self-organized Formation of Topologically Correct Feature Maps." *Biological Cybernetics*, 1982: pp. 59–69.

L.N. de Castro, J.I. Timmis,. "Artificial Immune Systems: A New Computational Intelligence Approach." *Springer-Verlag*, 2002.

S. Hofmeyr, S. Forrest. "Architecture for an artificial immune system." *EvolutionaryComputation, vol. 8, no. 4*, 2000: pp. 443–473.