CSIT Conference 2023, Yerevan, Armenia, September 25 - 30

Polymorphic Malware Analysis Model

Robert Hakobyan
National Polytechnic University of Armenia
Yerevan, Armenia
e-mail: rob.hakobyan@polytechnic.am

Timur Jamgharyan
National Polytechnic University of Armenia
Yerevan, Armenia
e-mail: t.jamgharyan@yandex.ru

Abstract **— The paper presents the results of research on the use of Kohonen neural network in the analysis of polymorphic malware. The assessment of the quality of training was carried out by the fuzzy logic method. The datasets for training the neural network are based on the source code of the polymorphic malware *abc, cheeba, december_3, stasi, otario, dm, v-sign, tequila, flip.* Simulation of the Kohonen neural network operation at different iterations and visualization of the results was carried out.**

Keywords**—Kohonen neural network, fuzzy logic, polymorphic malware, intrusion detection system, context triggered piecewise hashing, InetSIM, REMnux.**

I. Introduction

When developing machine learning (ML) intrusion detection systems (IDS), one of the tasks is to model the behavior of various types of malware in various network infrastructure (NI) operating environments. Malware developers use a variety of techniques and tactics to hide the source code of malware, as well as apply ML techniques to develop it. The use of various artificial neural networks and the ML paradigm itself allows malware developers, using ML methods, to generate new versions of this software, the neutralization of which is difficult by known methods. Malware is able to adapt both to a specific operating environment and to a set of protocols used in a given NI.

In studies [1-4], various approaches to the application of IDS in NI are considered. In particular, in [3], a consensus-based IDS was considered, where a solution was proposed that could bypass the restriction for centrally controlled IDS, since initially attackers attack the IDS itself (this problem is relevant for hosted IDS with centralized control). In [5] the ability to apply the support vector machine algorithm was considered, improving the granularity of malware detection at the time of the study.

To detect malware, methods of static and/or dynamic (behavioral) analysis, program code or memory analysis [6]. An important component in building a security architecture for an NI is the IDS. However, the existing IDS are mostly deterministic. As a rule, detection of the fact of an attack itself is often difficult, since deterministic IDS are not able to detect attacks that go beyond the rules and/or signatures of attack detection.

The creation of IDS using ML is one of the important tasks facing NI security researchers. The task of integrating into IDS with ML a software component capable of detecting polymorphic malware is one of the most relevant.

Network security researchers offer various solutions for integrating ML IDS with deterministic IDS [7-9]. But there is a task, of preliminary assessment of network traffic for the presence of a critical number of malware code segments, beyond which the SID goes beyond the boundaries of reliable operation. Another task is the problem of efficient use of hardware resources, since ML IDS with several neural networks in its composition requires a huge hardware resource, although the activation of all neural networks is not always necessary. Also, the research task of creating an intermediate IDS component with ML becomes relevant, which activates all or specified neural networks from the composition of IDS with ML, when the quantitative value of malware is exceeded.

This research presents the results of a study on the use of Kohonen's artificial neural network to detect polymorphic malware using the *fuzzy logic* method. The use of the Kohonen artificial neural network is due to the fact that polymorphic malware is able to independently change its source code, increasing the number of possible and probable versions, and the Kohonen neural network is one of the effective tools for reducing and assessing the dimensionality of changes in malicious polymorphic malware code variations.

The use of the *fuzzy logic* method as an assessment of the quality of neural network training is that because it is rather difficult to set boundary between polymorphic malware and non-malware software, or malware non-polymorphic software, but an artificial neural network trained on datasets previously structured using *fuzzy logic*, allows solving this problem, within the given constraints. Also, the use of *fuzzy logic* makes it possible to improve the quality of neural network training, since, unlike evaluation by binary methods (in particular, using the Matthews correlation [10]), the *fuzzy logic* method allows you to determine the boundary of polymorphic malware change, which allows you to quickly reconfigure the IDS.

II. Terms and Definition

A. Basic concepts

• Polymorphic malware - is malware characterized by the following behavior: encryption, self-propagation, and modification of one or more components of the source code. It is designed to avoid detection by being able to create modified copies of itself [11-12].

https://doi.org/10.51408/csit2023_17

• VVV -Volume, Velocity, Variety, a set of characteristics for Big Data [13].
• Fuzzy logic is a form of multivalued logic in which the true values of variables can be any real numbers from 0 to 1 inclusive [14].
• Context-triggered piecewise hashing (CTPH) - method, calculation of piecewise hashes from input data [15].
• Software-defined networking (SDN) - is a data transfer network in which the network management level is separated from data transfer devices and implemented in software [16]

B. Neural networks

• Kohonen neural networks (Fig.1) - is an unsupervised ML algorithm the main element of which is the Kohonen layer. This network is useful when the data is scattered over many dimensions, and it is required to obtain them in a given number [17].

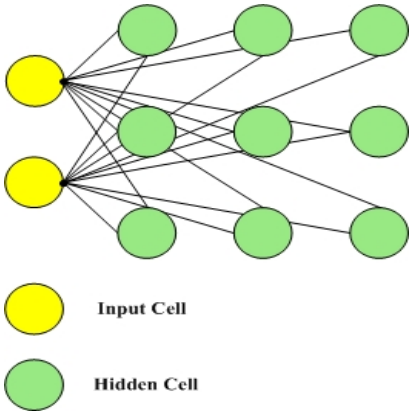

Fig. 1. Kohonen neural network

III. Model Description

The Hyper-V role is installed and an SDN is deployed in a virtual environment based on the Windows Server 2016 Standard operating system [18], in which:
• Kali Linux OS with pre-installed software InetSIM, for modeling various servers and protocols [19],
• Parrot OS which integrates the Metasploit framework that allows you to download malicious polymorphic software [20]. The malware used was *abc, cheeba, december_3, stasi, otario, dm, v-sign, tequila, flip* obtained from sources [21-24]. The datasets introduced into the operating environments under study are subjected to a noise denoising procedure [25].
• REMnux OS [26] in which the Kohonen neural network is installed in the *Clion* development environment. Kohonen neural network is trained on the values of CTPH obtained from datasets from malware *abc, cheeba, december_3, stasi, otario, dm, v-sign, tequila, flip.* The size of the CTPH file is 20,40,80 bytes. To increase the reliability of the results, the SDN using Hyper-V is displayed in a separate vlan (virtual local network, vlan). Quantitative and qualitative results were compared with the results obtained using the *ssdeep* software [27] and the methods proposed in [28-29], as well as with the *virustotal* service. The research scheme in SDN is presented in Fig. 2.

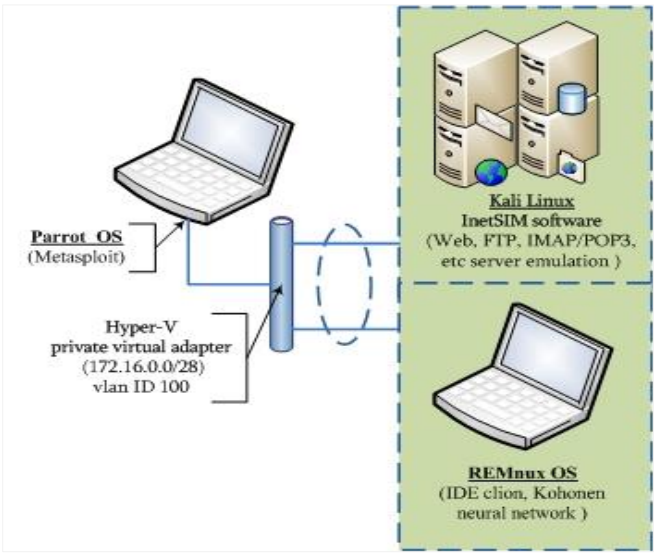

Fig. 2. The research scheme in SDN

Stage 1.
The trained Kohonen neural network is disabled. Search and detection of malware is carried out by the built-in REMnux OS tools. The decision on whether the software belongs to a malware type is made by analyzing the results obtained using the REMnux OS.

Stage 2.
The trained Kohonen neural network is enabled. The decision whether the software belongs to a malware or non-malware type is based on *fuzzy logic*. The membership function of the Gaussian type is described by (1) [30].

$$mf\left(x\right)=\exp\left[-\left(\frac{x-c}{\sigma}\right)^2\right] \qquad (1)$$

where:
mf-membership function,
x-degree of membership in a fuzzy set,
c-center of fuzzy set,
σ - steepness of the membership function.
The block diagram of the integration of the Kohonen neural network with the software that performs the comparison using the fuzzy logic method is shown in Fig. 3.

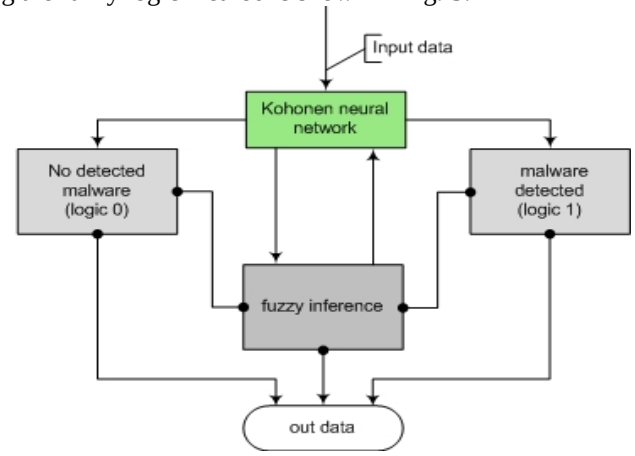

Fig. 3. Kohonen neural network with integrated module of *fuzzy logic*

In active services deployed using InetSIM, datasets of polymorphic malware with different value of the CTPH file size are gradually introduced. The input of malware datasets was carried out using the Metasploit framework. The researches was conducted in 3 learning epochs, with 11 iterations.

IV. Research results

Figures. 3, 4 and 5 show the visualization of the results of the detection of polymorphic malware *abc, cheeba, december_3, stasi, otario, dm, v-sign, tequila, flip* with a CTPH file size of 20,40, 80 bytes.

Tables 1, 2, 3 present the numerical values of the detected segments of polymorphic malware at a CTPH, file size of 20,40, 80 bytes per 50 MB of network traffic.

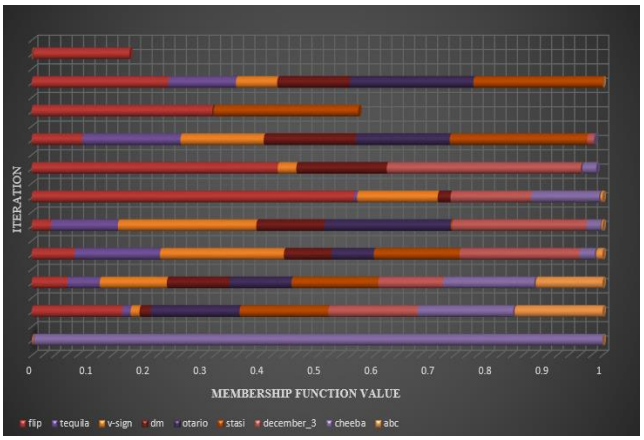

Fig. 4. Visualization of the detected polymorphic malware at 1 training epoch (11 iterations, CTPH value 20 bytes)

Table 1

Number of detected segments of polymorphic malware with a CTPH value of 20 bytes

mf value	*abc*	*cheeba*	*december*	*stasi*	*otario*	*dm*	*v-sign*	*tequila*	*flip*
0	1.25	1.23	1.12	0.25	1	2.92	1.15	2.84	2.48
0.1	0.12	1.13	2.25	1.12	0.01	0.14	2.25	0.68	1.23
0.2	0.13	2.32	3.25	2.32	0.25	0.26	1.29	0.38	0.75
0.3	0.14	2.17	1.25	1.14	0.04	1.14	2.21	0.08	1.32
0.4	1.23	2.15	1.12	2.12	0	0.24	2.15	0.48	2.25
0.5	1.24	3.02	2.25	0.24	0	0.24	3.14	2.25	2.35
0.6	1.35	2.25	3.15	2.25	0.25	2.36	0.15	0.38	1.04
0.7	1.25	3.16	0.36	0.26	0.26	0.16	0.96	0.44	0.54
0.8	1.36	2.36	0.22	0.34	0.01	0.14	0.12	0.68	0.12
0.9	1.16	2.12	1.14	2.14	2.05	1.05	1.15	0.12	1.12
1	2.35	3.25	3.04	1.16	1.06	1.12	2.18	0.65	1.64

As can be seen from Fig. 4 and Table 1, certain polymorphic malware is classified binary (*otario*), but the rest can only be declared to belong to the type of malware with a certain probability (*abc, cheeba, december,stasi, dm, v-sign, tequila, flip*). Accordingly, the task arises to test this malware in several epochs and iterations of the neural network in order to more accurately determine the level of significance.

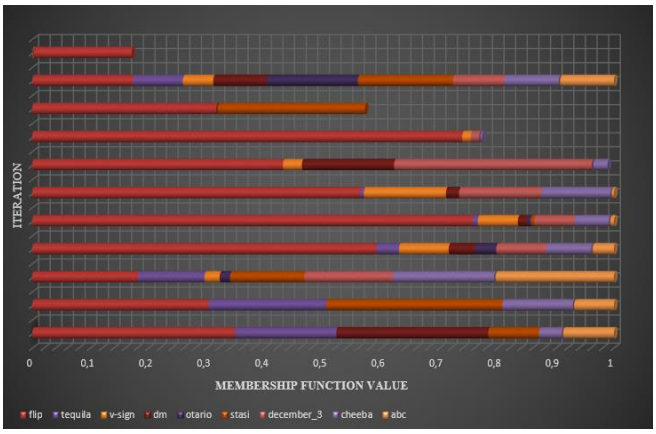

Fig. 5. Visualization of the detected polymorphic malware at 1 training epoch (11 iterations, CTPH value 40 bytes)

Reducing the step size of CTPH makes it possible to increase the selectivity of the output results based on fuzzy logic. The neural network, learning and retraining on data sets formed on the basis of CTPH, with an increase in training epochs, produces both binary results and results based on *fuzzy logic* (not binary).

Table 2

Number of detected segments of polymorphic malware with a CTPH value of 40 bytes

mf value	*abc*	*cheeba*	*december*	*stasi*	*otario*	*dm*	*v-sign*	*tequila*	*flip*
0	0.25	2.11	0	1.12	0.15	0.15	0	0.15	2.15
0.1	0.27	1.35	0.01	1.35	0.64	0.26	0	0	2.18
0.2	0.37	1.24	0.23	1.23	0.97	0.01	0	1.23	1.78
0.3	0.16	1.25	0.11	2.12	1.16	0.05	0	0	1.65
0.4	0.65	1.16	1.12	2.54	1.64	0	0	1.25	1.45
0.5	1.35	2.18	0.15	1.38	1.85	0	0.65	1.38	2.15
0.6	0.25	0	0.35	1.68	1.95	0.16	0.85	0	0
0.7	1.21	0.03	0.23	1.12	2.65	0.65	1.16	0	0
0.8	0.15	0.13	0.74	1.35	2.15	1.1	1.46	0.05	0
0.9	0.98	0	0.85	1.64	2.64	0.95	0.98	0	0
1	1.75	0	1.03	1.28	2.15	1.18	0.65	0	0

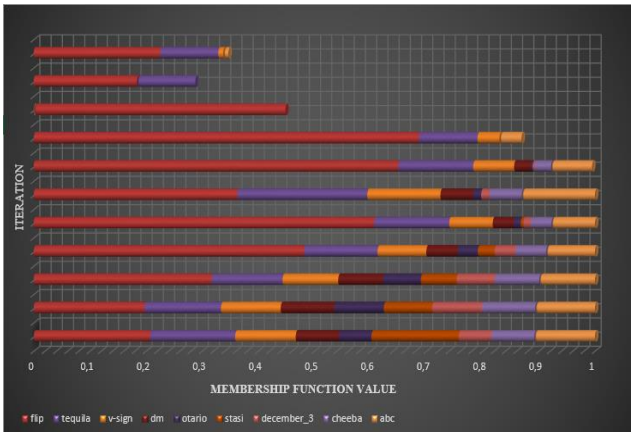

Fig. 6. Visualization of the detected polymorphic malware at 1 training epoch (11 iterations, CTPH value 80 bytes)

Table 3

Number of detected segments of polymorphic malware with a CTPH value of 80 bytes

mf value	*abc*	*cheeba*	*december*	*stasi*	*otario*	*dm*	*v-sign*	*tequila*	*flip*
0	0	0	0.12	0	0	0	0	0	1.18
0.1	0	0	0.11	0.32	0	0	0	0	1.65
0.2	0	2.55	0	0.22	2.03	0	0	3.15	1.32
0.3	0.35	2.97	0	0.34	2.08	0	0.02	3.18	1.15
0.4	0	3.01	0.01	1.15	0	0	2.15	2.16	1.76
0.5	0	2.11	1.18	1.45	0.81	1.35	2.34	1.17	0
0.6	2.37	5.15	3.65	1.35	0.88	1.65	2.16	2.15	0
0.7	2.45	1.12	3.75	1.75	0.95	1.24	2.11	1.27	0
0.8	2.78	1.06	3.88	0.58	0.11	1.64	2.13	1.34	0.23
0.9	3.15	1.02	1.35	0.06	0.08	0.01	2.10	1.14	0.45
1	3.10	0.98	1.65	0	0	0	2.09	1.03	0.78

The application of the method makes it possible to carry out a preliminary assessment of the presence of polymorphic malware in network traffic and, accordingly, make a decision on the activation / non-activation of other neural networks for malware detection, which, in particular, allows for more efficient hardware management.

V. Implementation

The implementation of the program code was made in the Python programming language using the Tensor flow deep learning library and the Keras library for the rapid implementation of neural networks. The Google Colab service [31] was also used.

VI. Conclusions

The developed model allows a preliminary assessment of network traffic for the presence of polymorphic malware. Based on the results obtained, taking into account pre-formed threshold values for the presence of malware for a given amount of network traffic, it is possible to set an activation policy for other components of the ML IDS.

With a decrease in CTPH, an increase in the degree of detection of polymorphic malware is observed. With an increase in the step of CTPH, the number of errors of the 1st kind increases. The calculations were carried out on a Dell Power Edge T-330 server. Software source code and all research results are available at [32].

References

[1] A.Sayed, A.Aziz, A.Azar, A.Hassanien, S. Hanafy, (2014), "Negative Selection Approach Application in Network Intrusion Detection Systems", (2014), https://doi.org/10.48550/arXiv.1403.2716

[2] A.Madbouly, A.Gody, T.Barakat, "Relevant Feature Selection Model Using Data Mining for Intrusion Detection System", (2014), https://doi.org/10.14445/22315381/IJETT-V9P296

[3] M.Toulouse, B.Minh, P.Curts, «A consensus based network intrusion detection system», (2015), https://doi.org/10.48550/arXiv.1505.05288

[4] P.Sree, I.Babu, "Towards a Cellular Automata Based Network Intrusion Detection System with Power Level Metric in Wireless Adhoc Networks (IDFADNWCA)", (2014), https://doi.org/10.48550/arXiv.1401.4012

[5] R.Chen, K.Cheng, C.Hsieh, "Using Rough Set and Support Vector Machine for Network Intrusion System", (2010), https://doi.org/10.48550/arXiv.1004.0567

[6] K.A.Monappa. *Learning Malware Analysis.Explore the concepts, tools, and techniques to analyze and investigate Windows malware.* Packt>. Birmingham-Mumbai. 2018

[7] F.Zhong et al, "MalFox: Camouflaged adversarial malware example generation based on Conv-GANs against black—box detectors", (2020), https://arxiv.org/abs/2011.01509

[8] Dominik Kus et al, "A false sense of security? Revisiting the state of machine learning-based industrial intrusion system", (2022), https://arxiv.org/abs/2205.09199

[9] K.Jallad, M.Aljnidi, M.Desoki, «Big data analysis and distributed deep learning for next-generation intrusion detection system optimization", (2022) // https://arxiv.org/abs/2209.13961

[10] V.V. Starovoitov, Yu.I. Golub, "Comparative study of quality estimation of binary classification". *Informatics*. (2020), 17(1):87-101. (In Russ.) https://doi.org/10.37661/1816-0301-2020-17-1-87-101

[11] Official website of Kaspersky anti-virus software. [Online]. Available: https://encyclopedia.kaspersky.ru/knowledge/malicious-programs/

[12] K.Jallad, M.Aljnidi, M.Desoki, "Big data analysis and distributed deep learning for next-generation intrusion detection system optimization", (2022) // https://arxiv.org/abs/2209.13961

[13] D. Laney, "3D Data Management:Controlling Data Volume, Velocity, and Variety", Application Delivery Strategies. Published by Meta Group Inc. 6 February 2001.

[14] L.A.Zadeh, Fuzzy sets, Department of Electrical Engineering and Electronics Research Laboratory, University of California, Berkeley, California, USA https://doi.org/10.1016/S0019-9958(65)90241-X

[15] Kornblum, Jesse. (2006). Kornblum, J.: Identifying Almost Identical Files using Context Triggered Piecewise Hashing. Digital Investigation 3(suppl.), 91-97. Digital Investigation. 3. 91-97. 10.1016/j.diin.2006.06.015.

[16] Benzekki, Kamal; El Fergougui, Abdeslam; Elbelrhiti Elalaoui, Abdelbaki, «Software-defined networking (SDN): A survey». *Security and Communication Networks*. vol. **9** (18),pp.5803–5833, 2016, https://doi.org/10.1002%2Fsec.1737

[17] T. Kohonen, "Self-organized formation of topologically correct feature maps". Biol. Cybern, 43, 59–69, 1982.

[18] Microsoft official site. Windows Server 2016 Operating System Download Page [Online].Available: https://www.microsoft.com/en-us/evalcenter/download-windows-server-2016

[19] Kali Linux OS official site. Kali Linux Operating System Download Page. [Online].Available: https://www.kali.org/

[20] Parrot OS official site. Parrot OS Operating System Download Page [Online].Available: https://www.parrotsec.org/

[21] Malware Bazaar Database. [Online]. Available https://bazaar.abuse.ch/browse/

[22] Malware database. [Online]. Available http://vxvault.net/ViriList.php

[23] Malware repository. [Online]. Available https://avcaesar.malware.lu/

[24] Viruses repository. [Online]. Available: https://virusshare.com/

[25] T.Jamgharyan, «Research of the data preparation algorithm for training generative-adversarial network», *Bulletin of High Technology* no. 19, pp. 40-50, 2022.

[26] REMnux OS official site. REMnux Operating System Download Page [Online].Available: https://remnux.org/

[27] ssdeep software project website. [Online].Available https://ssdeep-project.github.io/ssdeep/index.html

[28] K.A. Tyurin, "Fuzzy hashing in information security problems" // Obzor. NTSPTI. no. 1 (16), 2019. [Online].Available. https://cyberleninka.ru/article/n/nechyotkoe-heshirovanie-v-zadachah-informatsionnoy-bezopasnosti

[29] T. V. Jamgharyan, "Research of Obfuscated Malware with a Capsule Neural Network". *Mathematical Problems of Computer Science*, vol. *58*, pp. 67–83, 2022, https://doi.org/10.51408/1963-0094

[30] L.Rutkowski. "*Artificial neural networks. Theory and practice",* Hot line - Telecom, 2010.

[31] Official website of data analysis and machine learning Colaboratory. [Online].Available: https://colab.research.google.com

[32] Software source code and all research results, [Online]. Available: https://github.com/T-JN?tab=repositories