X-ware: a proof of concept malware utilizing artificial intelligence

Thanh Cong Truong[1], Jan Plucar[2], Quoc Bao Diep[2], Ivan Zelinka[2]
[1]Department of Computer Networks and Data Communications, Faculty of Information Technology, University of Finance-Marketing, Ho Chi Minh City, Vietnam
[2]Department of Computer Science, Faculty of Electrical Engineering and Computer Science, VSB-Technical University of Ostrava, Ostrava, Czech Republic

Article Info

Article history:

Received Mar 8, 2021
Revised Sep 9, 2021
Accepted Oct 2, 2021

Keywords:

Complex network
Cybersecurity
Malware
Neural network
Swarm intelligence
Swarm virus

ABSTRACT

Recent years have witnessed a dramatic growth in utilizing computational intelligence techniques for various domains. Coherently, malicious actors are expected to utilize these techniques against current security solutions. Despite the importance of these new potential threats, there remains a paucity of evidence on leveraging these research literature techniques. This article investigates the possibility of combining artificial neural networks and swarm intelligence to generate a new type of malware. We successfully created a proof of concept malware named X-ware, which we tested against the Windows-based systems. Developing this proof of concept may allow us to identify this potential threat's characteristics for developing mitigation methods in the future. Furthermore, a method for recording the virus's behavior and propagation throughout a file system is presented. The proposed virus prototype acts as a swarm system with a neural network-integrated for operations. The virus's behavioral data is recorded and shown under a complex network format to describe the behavior and communication of the swarm. This paper has demonstrated that malware strengthened with computational intelligence is a credible threat. We envisage that our study can be utilized to assist current and future security researchers to help in implementing more effective countermeasures.

Corresponding Author:

Thanh Cong Truong
Department of Computer Networks and Data Communications, Faculty of Information Technology, University of Finance-Marketing
Ho Chi Minh City, Vietnam
Email: ttcong@ufm.edu.vn

1. INTRODUCTION

Malware, or more precisely, malicious software, stands for software or program which infiltrates and damages computer systems with no prior consent to the system owner. Over the years, malware is indicated as a security threat by cyber security standards [1]. Malware includes viruses, Trojan horses, worms, exploits, botnets, and retroviruses [2]. Among those listed, the computer virus termed by Cohen [3], which is the most popular. Despite being an unharmful software that causes minor disturbances to computer users in the early days, the virus has been developed to reach a particular financial goal or even becomes a virtual weapon utilized in cyber war.

Today, creating malware or anti-malware has been a billion-dollar industry. The never-ending war between malware and anti-malware is where cyber attackers implement new techniques to overcome malware detection as the defenders strive for effective measures to counter the attacks. The anti-malware authors have implemented various types of heuristics to recognize new and unknown malware. On the other hand, their opponents constantly find different methods for attacking [4], [5].

The malware spreading has been a topic for numerous researches. Studies of its dynamic and behaviors when placed in real-world networks are among the main trends. For example, virus infection on computers was investigated by authors in [6] implementing epidemiologically compartmental models to identify potential contagious sources. Concurrently, another group [7], introduced a moderate epidemiological model inspired by the fractional epidemiological model for descriptions of computer viruses with an arbitrary order derivative and a non-singular kernel. Besides, authors in [8] proposed a novel virus heterogeneous propagation model and a tool to optimize its propagation. Lately, the combination of malware and machine learning (ML) and deep learning (DL) has been raised as a new research trend among scientists. Many researches focused on utilizing ML or DL techniques as in [9]–[16] to evade anti-malware system.

Recently, we have witnessed a trend towards using bio-inspired techniques like swarm intelligence (SI) in various complex issues in general and cyber security in particular [17]. Frameworks for malware evolution based on evolutionary computation for malware were proposed in [18]. Furthermore, the researchers in [19] exploited the evolutionary algorithm (EA) to generate malware automatically. Kudo *et al*. [20], proposed botnets with adopted ML techniques for prediction of system vulnerabilities and malware self-governed evolution. Nevertheless, a search of the literature revealed few studies which address the combination of SI and other intelligence techniques in malware. Hence, we aim to explore the possibility of fusing SI and ML to develop a proof of concept malware, allowing us to identify the characteristics of this potential threat for developing mitigation methods in the future. Accordingly, we experiment with a malware prototype in a secured virtual environment with real-time observations, data recording, visualization, and some fundamental analysis.

The approach described in this paper is an improvement of the swarm virus introduced in [21]–[23]. Thus, we propose fusing SI, an artificial neural network (ANN), and a classical computer virus to form a new kind of malware. This software can simulate a biological swarm system's behavior, which usually does not have a dominant central communication point. Thus, a virus prototype, namely X-ware, with controllable and limited contagion, was created. Furthermore, we discuss the possible mitigation approaches based on this idea and its use in complex computer systems as the artificial intelligence (AI) supervisor (deus ex machina) for solving non-malware problems. Briefly, the significant contributions of this paper are as follows: i) we present a novel malware proof of concept, which is showed that swarm intelligence, ANN could be embedded in a virus; ii) we conduct experimentations and demonstrate how to capture and visualize the virus's behavior and analyze the swarm malware activities using complex network analysis, iii) finally, we discuss possible mitigation and remediation ideas for future anti-malware technologies.

This manuscript's remaining proceeds as follows: section 2 presents the method for developing the X-ware with its structure and functionalities. After that, section 3 contains experimental results and discussion. Finally, section 4 concludes the paper.

2. RESEARCH METHOD

This section describes the method of developing a prototype of the hypothesis X-ware. We also show how to capture and visualize such malware's behavior when it walks through the operating system. The swarm virus prototype, designed here, mimics a swarm system behavior and follows the main idea of a swarm algorithm. Malware has evolved drastically since its early days, being an unharmful annoyer. Various techniques have been adopted to virus development, such as oligomorphism, metamorphism, polymorphism, encryption, armouring (armoured viruses) and obfuscation to bypass anti-malware. Many modern viruses such as Mirai, Lokibot and AZORult are controlled with the command and control (C&C) infrastructure method. Nevertheless, C&C has a weak spot of being immobilized when its control center is demolished.

Acting as a malware researcher, to eliminate the weak spot of the botnet structure being its C&C center, the author group utilized swarm-based intelligence and ANN to a traditional virus to produce a new malware named X-ware. Technically, this virus consists of instances (many individuals) forming a swarm (population) that propagates in the computer file system and computer network. The swarm individuals communicate via specific communication channels (when shifting from host to host) and among themselves. The global data is stored inside each virus so that the swarm can be expected to perform decentralized behavior. Whenever there is a change in the swarm (e.g., the virus moves to another host), the information will be updated to every individual in the flock. Furthermore, when a member of the swarm is eliminated (removed by a user or deleted by antivirus software), another one will be regenerated to ensure that the population's number is constant. In addition, the ANN embedded on the prototype can be used as an intelligent center that keeps payload, triggers conditions to execute the payload on the aimed target. On the other hand, the virus can simulate the working mechanism of an ANN to enhance robustness. Figure 1 illustrates principle of underlying the proposed idea.

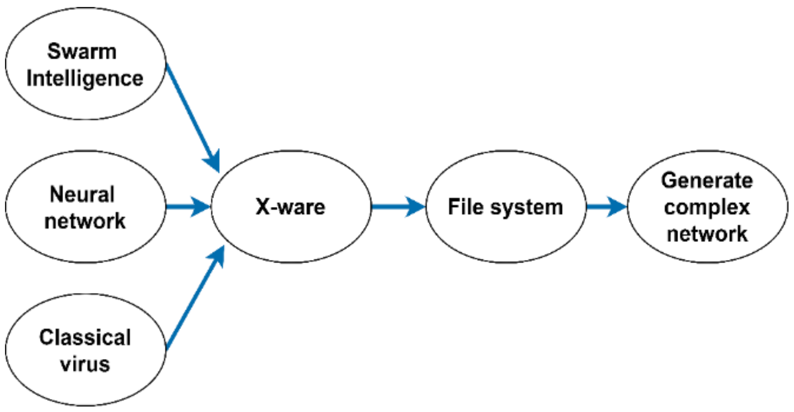

Figure 1. The main idea of the X-ware and its visualization

2.1. Incorporate ANN and the virus

This part discusses how to incorporate ANN and the proposed virus. In this study, the concept of a multi-layer perceptron (MLP) network has been utilized to enhance the malware. More specifically, we adopted MLP in two scenarios: i) each of the virus in the swarm will contain an MLP and ii) several individuals in the population will simulate the behavior of the MLP to perform their intended actions.

2.1.1. Each virus comprises a whole ANN

In this implementation, each individual in the swarm contained an MLP. The MLP implementation, in this scenario, is composed of three layers: input layer, hidden layer and output layer. The file's size was used as input to the MLP network's input layer, which contains a total of two neurons. The hidden layer consists of two neurons for network training, and an output layer consists of one neuron as it produces the result of whether to perform the malware tasks or not. In terms of activation function, there are a variety of methods that could be used for training. In this paper, for the purpose of functionality demonstration, the logistic sigmoid function is utilized. The back-propagation algorithm [24] is used to train the MLP in the proposed approach. The dataset used for training is collected from system files. After training the model, the optimized network weights are integrated into the virus. When the virus executes, these weights are used on the embedded MLP to make the computation. Then, the MLP makes the trigger conditions to perform the execution, or it can be responsible for other activities such as locating the target and executing payload on the right object. Viruses are trained to perform system searches for finding a suitable target. The ANN then generates signals for conducting a task. For the herein experiments, the task is to display a message. Additionally, the file size is utilized for target identification. Other attributes, such as system-level features, can be utilized. Subsequently, it is impossible to reverse the ANN to work out the target specifications. Malware can use an ANN model, which is a black box, instead of a traditional *if-then* command line to camouflage its trigger condition. This seems to be a much more effective camouflage technique than obfuscation. Technique wise, this complicates the deciphering jobs of anti-malware by hiding the target categories or triggering conditions.

2.1.2. Virus act as a node of an ANN

This kind of implementation uses several viruses in the swarm to act as input layers, hidden layers and output layers of an MLP. In this kind of approach, only some individuals in the swarm (black colored) act as nodes in the MLP network. More precisely, two individuals are utilized as input nodes to receive signals, three for hidden nodes and one for the output node. The MLP is also trained to obtain the optimized weights in the same manner as mentioned above. The difference is the weight allocated to individuals that act as nodes in the MLP network. In other words, each node comprises its weights. When the virus executes, the swarm simulates the working mechanism of an MLP network. More specifically, the signal values propagated from the inputs virus, through the connection to the hidden virus, and then onward through more connection to the output virus. Following this strategy, the virus's payload could be distributed, and it is extremely hard to reverse engineer the virus (reverse engineers must capture the whole swarm).

2.2. Visualizing X-ware behavior

The previous literature [21] pointed out that the virus's movement in the swarm and the communication inside the swarm likely follow the complex network topology so that any swarm intelligence based techniques, which permit search over the graph, could be adopted to build the complex network. The principal idea is such that a vertex represents each individual, and edges between vertices should reflect dynamics in population, i.e., interactions between individuals. For instance, self-organizing migrating algorithm (SOMA) [25] could be used to form a complex structure. The SOMA algorithm consists of a leader drawing the entire population in each migration loop. Hence, the population of activated leaders shall be recorded like vertex, and the interactions between Leaders and individuals shall be recorded like edges.

3. RESULTS AND DISCUSSION

3.1. Results

In the herein experiments, a swarm consisting of five individuals was created, which means there are five virus instances. In each experiment, an individual jumped from file to file (infected a file) twenty times in total. The fascinating behavior of the swarm was recorded and visualized. Furthermore, the system's virus behavior had been analyzed for different network attributes such as degree centrality, closeness centrality, betweenness centrality, and eigenvector centrality. The obtained statistical data is given in Table 1. Figure 2 presents the visualization of the networks in term of degree, betweenness, closeness, and eigenvector centrality. As can be seen from the graphs, there are multiple nodes that are increasing (distinguished by their size), emphasizing their prominence in the population.

Table 1. Swarm virus network centralities

	Min	Median	Max
Degree centrality	0.095	0.057	0.485
Closeness centrality	0	0.178	1
Betweenness centrality	0	0.02	0.162
Eigenvector centrality	0	0.004	1

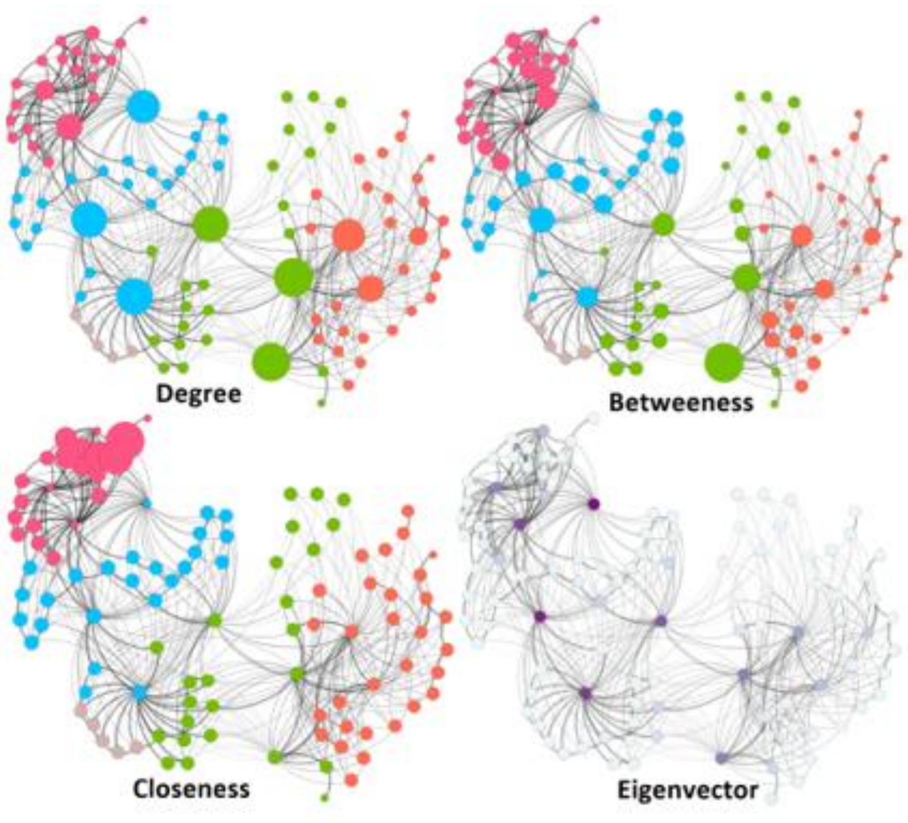

Figure 2. Centralities of the X-ware network, capturing its movement through host system

Figure 3 depict the histograms of observed network attributes. The empirical result shows that the more important a node is, the higher centrality it has. Subsequently, if a node has a higher centrality, then it has more probability of being visited. In other words, the important files have a higher centrality, which means they have a higher chance to be infected by the malware prototype. In contrast, less important files have lower centrality values and a lower chance of being visited by the virus.

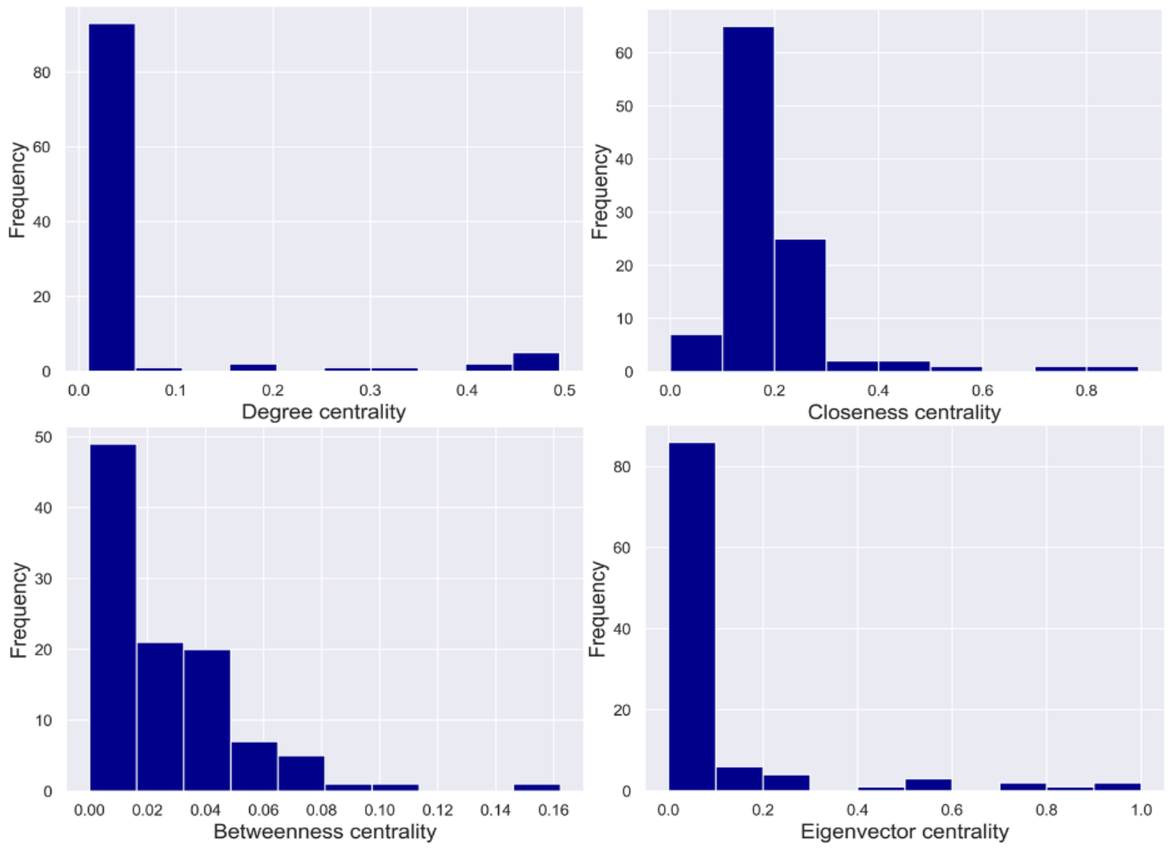

Figure 3. Histogram of the network centralities

As shown in the Figure 3, the closeness centrality is mainly distributed over the entire network. This conveys that the majority of the nodes are contributed evenly in the network. In this graph, the bigger nodes have higher betweenness centrality. Incidentally, the node which has the highest betweenness centrality also has the best fitness. One crucial aspect to be taken into account is that once the node for the best fitness changes, the betweenness centrality of the system alters as well. Furthermore, in the Figure 3, the colour coding is utilised to distinguish the centrality. Darker nodes are more central, and lighter nodes are less central. As shown in the Figure 3, some individuals are more influent than others in the swarm.

3.2. Discussion

Our empirical experiments show that the files with higher fitness also have higher centrality metrics, which means these files are more important than the others. They are distinguishable from others. In contrast, less critical files have lower centrality estimates. This means a random X-ware instance, which is moving through the system files or network, has a higher probability of infecting a file with the higher fitness files.

Furthermore, Figure 3 indicates that there is a powerful inverse correlation between centrality measure values (for betweenness, degree, and eigenvector) and their frequencies, which lead to its being modelled as a power-law distribution. Additionally, these figures show that the major vertices have low centrality values, while just minor vertices have high centrality values. On the other hand, the distribution of closeness centrality follows the normal curve. The relationship between degree centrality and its frequency probability matches the curve, which may indicate that the network has a scale-free character.

The essential factor is that files with higher fitness should be distinct. In our experiments show that most important files also have higher centrality. In figures, the nodes that represent important files are bigger than the other. In contrast, less important should be smaller. Graphs\ref{fig: centralities} is showing the dependency between importances and centralities. In these graphs, the most important file is the biggest node in the figure. So it is the most influence node in the network as well. Furthermore, in the figure, the darker nodes indicate that they are more influential than the others. These results suggest that if we can identify the most centrality node in the network, then the virus's spreading rate could be decreased and potentially eradicated.

The prototype version of X-ware is developed and observed in a controlled environment so that its behavior and data can be easily obtained. Nevertheless, in reality, identifying the self-replicating swarm structures is a difficult task. As our experiments verify, the X-ware has two significant features i) it is moving over the hosts while keeping the constant of the population and ii) the communication among them. These features shall act as the critical criteria for identifying the to-be-created likewise prototypes. Hence, we suggest that the protection systems should not destroy them instantly but observe them and analyze their activities data as a whole in order to discover the activities of some subset of such malware that can be expected from the X-ware (i.e., movement, communication, trigger). Additionally, complex network visualization and analysis can help a lot. By applying the network analysis, we can discover the critical nodes, which play an essential role in the swarm network and thus can take corresponding actions based on the analysis. X-ware's ideas are fully applicable to designing future anti-malware solutions. For instance, we can create adaptive, autonomous AI agents that collaborate to achieve common tasks. Instead of getting guidance from a single, centralized AI model, agents will be smart and robust enough to communicate with each other and work together to achieve common goals.

Agents will learn how to protect systems depend on what they inspect from their networks and local hosts. Furthermore, their strength is further enhanced by observations and behaviors learned across different industries and majors. Generally speaking, we will have a swarm of rapid response local AIs that accommodate their environment while collaborating with each other, instead of one big AI system delivering decisions. This will improve organizations' IT performance by saving resources and helping them avoid sharing confidential, potentially sensitive information through the cloud or other means.

4. CONCLUSION

In this paper, we presented X-ware, a proof of concept malware using ANN and SI, to study its features to form the anti-malware solution in the future. This research discussed the method to develop an X-ware prototype in which the ANN acts as an intelligent center that keeps payload, triggers conditions with no destructive payload and controllable contagion. Furthermore, there were practical experiments to visualize, measure, and analyze the X-ware behavior under the form of a complex network. Our research pointed out that its movement and internal communication follow the complex network topology. In addition, information was shared, updated, and used by all swarm members directly in the communication progress. The results yielded from this work offer a better understanding of the behavior of a possible new generation of malware in order to protect future computer technology. As this work has shown, the X-ware prototype is a swarm one in which all individual viruses can communicate amongst themselves as a warm in nature would do. This leads to another possibility of adopting the idea to other kinds of malware like worms or Trojans so that their activities can be more distributed and robust. Furthermore, the environment then is not only on the PCs but also on networks. In future work, we will focus on proposing it as the autonomous anti-malware technology (i.e. Deux ex Machina) in complex and large systems.

ACKNOWLEDGEMENTS

The following grants are acknowledged for the financial support provided for this research: Grant of SGS No. SP2021/72, VSB-Technical University of Ostrava.

REFERENCES

[1] T. Kumar, S. Sharma, R. Dhaundiyal, and P. Jain, "Invesitigation of Malware and forensic tools on internet," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 5, pp. 3179-3186, Oct. 2018, doi: 10.11591/ijece.v8i5.pp3179-3186.
[2] P. Szor, *The art of computer virus research and defense: Art comp virus res defense _p1*. Pearson Education, 2005.
[3] F. Cohen, "Computer viruses," *Computers & Security*, vol. 6, no. 1, pp. 22–35, Feb. 1987, doi: 10.1016/0167-4048(87)90122-2.
[4] C. T. Thanh and I. Zelinka, "A survey on artificial intelligence in malware as next-generation threats," *MENDEL*, vol. 25, no. 2, pp. 27–34, Dec. 2019, doi: 10.13164/mendel.2019.2.027.
[5] T. C. Truong, Q. B. Diep, and I. Zelinka, "Artificial intelligence in the cyber domain: offense and defense," *Symmetry*, vol. 12, no. 3, p. 410, Mar. 2020, doi: 10.3390/sym12030410.
[6] W. Pan and Z. Jin, "Edge-based modeling of computer virus contagion on a tripartite graph," *Applied Mathematics and*

Computation, vol. 320, pp. 282–291, Mar. 2018, doi: 10.1016/j.amc.2017.09.044.
[7] J. Singh, D. Kumar, Z. Hammouch, and A. Atangana, "A fractional epidemiological model for computer viruses pertaining to a new fractional derivative," *Applied Mathematics and Computation*, vol. 316, pp. 504–515, Jan. 2018, doi: 10.1016/j.amc.2017.08.048.
[8] X. Zhang and C. Gan, "Global attractivity and optimal dynamic countermeasure of a virus propagation model in complex networks," *Physica A: Statistical Mechanics and its Applications*, vol. 490, pp. 1004–1018, Jan. 2018, doi: 10.1016/j.physa.2017.08.085.
[9] K. Grosse, N. Papernot, P. Manoharan, M. Backes, and P. McDaniel, "Adversarial perturbations against deep neural networks for malware classification," *Cryptography and Security*, Jun. 2016.
[10] W. Xu, Y. Qi, and D. Evans, "Automatically evading classifiers: a case study on PDF malware classifiers," in *Proceedings 2016 Network and Distributed System Security Symposium*, 2016, doi: 10.14722/ndss.2016.23115.
[11] H. S. Anderson, J. Woodbridge, and B. Filar, "DeepDGA," in *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*, 2016, pp. 13–21, doi: 10.1145/2996758.2996767.
[12] H. S. Anderson, A. Kharkar, B. Filar, and P. Roth, "Evading machine learning malware detection," *Black Hat*, pp. 1–6, 2017.
[13] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on GAN," *Machine Learning*, Feb. 2017.
[14] H. S. Anderson, A. Kharkar, B. Filar, D. Evans, and P. Roth, "Learning to evade static PE machine learning malware models via reinforcement learning," *Cryptography and Security*, Jan. 2018.
[15] A. Al-Dujaili, A. Huang, E. Hemberg, and U.-M. OReilly, "Adversarial deep learning for robust detection of binary encoded malware," in *2018 IEEE Security and Privacy Workshops (SPW)*, 2018, pp. 76–82, doi: 10.1109/SPW.2018.00020.
[16] X. Li and Q. Li, "An IRL-based malware adversarial generation method to evade anti-malware engines," *Computers & Security*, vol. 104, p. 102118, May 2021, doi: 10.1016/j.cose.2020.102118.
[17] J. Del Ser *et al.*, "Bio-inspired computation: Where we stand and what's next," *Swarm and Evolutionary Computation*, vol. 48, pp. 220–250, Aug. 2019, doi: 10.1016/j.swevo.2019.04.008.
[18] G. Meng *et al.*, "Mystique: evolving android malware for auditing anti-malware tools," in *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, 2016, pp. 365–376, doi: 10.1145/2897845.2897856.
[19] A. Cani, M. Gaudesi, E. Sanchez, G. Squillero, and A. Tonda, "Towards automated malware creation," in *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, 2014, pp. 157–160, doi: 10.1145/2554850.2555157.
[20] T. Kudo, T. Kimura, Y. Inoue, H. Aman, and K. Hirata, "Stochastic modeling of self-evolving botnets with vulnerability discovery," *Computer Communications*, vol. 124, pp. 101–110, Jun. 2018, doi: 10.1016/j.comcom.2018.04.010.
[21] I. Zelinka, S. Das, L. Sikora, and R. Šenkeřík, "Swarm virus - Next-generation virus and antivirus paradigm?," *Swarm and Evolutionary Computation*, vol. 43, pp. 207–224, Dec. 2018, doi: 10.1016/j.swevo.2018.05.003.
[22] T. C. Truong, I. Zelinka, and R. Senkerik, "Neural swarm virus," in *Swarm, Evolutionary, and Memetic Computing and Fuzzy and Neural Computing*, Springer International Publishing, 2020, pp. 122–134.
[23] T. C. Truong, Q. B. Diep, I. Zelinka, and T. T. Dao, "X-Swarm: The upcoming swarm worm," *MENDEL*, vol. 26, no. 1, pp. 7–14, May 2020, doi: 10.13164/mendel.2020.1.007.
[24] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986, doi: 10.1038/323533a0.
[25] I. Zelinka, "SOMA-self-organizing migrating algorithm," in *New Optimization Techniques in Engineering*, Springer Berlin Heidelberg, 2004, pp. 167–217.

BIOGRAPHIES OF AUTHORS

Thanh Cong Truong currently working at the University of Finance-Marketing, Ho Chi Minh city, Vietnam. He obtained his MSc. in Computer Science at the University of Information Technology, Vietnam National University Ho Chi Minh City, and Ph.D. at the Faculty of Electrical Engineering and Computer Science, VŠB-Technical University of Ostrava (VŠB-TU). His primary research relates to Artificial intelligence, swarm intelligence algorithms, and cybersecurity. He has published numerous research articles in peer-reviewed journals and international conferences. He can be contacted at email: ttcong@ufm.edu.vn.

Quoc Bao Diep is currently an informatics Ph.D. student at the Faculty of Electrical Engineering and Computer Science, Technical University of Ostrava (VŠB-TU). He received the bachelor's and master's degree in Mechatronics Engineering from Can Tho University and the Ho Chi Minh City University of Technology in Vietnam, respectively. His forte studies include robotics control, swarm intelligence-based algorithm, and Matlab 3D simulation. He can be contacted at email: diepquocbao@gmail.com.

Jan Plucar has been employed as researcher at Technical University of Ostrava (VSB-TU), Faculty of Electrical Engineering and Computer Science since 2015. He graduated both master and post gradual level studies at VSB-TU. During his career, he first focused on bio-inspired calculations, but soon switched to computer security. In this area, he helped to establish a new study program at VSB-TUO with a focus on computer security. He guarantees or teaches three subjects in the given field, namely Computer Attack and Defense, Computer Viruses and Forensic Analysis. During his career, he has participated in a number of national and international projects. Among the interesting ones are, for example, FP7/218086-INDECT 7th Framework Program EU or TF01000091-Security of Mobile Devices and Communication. He can be contacted at email: jan.plucar@vsb.cz.

Ivan Zelinka is currently working at the Technical University of Ostrava (VSB-TU), Faculty of Electrical Engineering and Computer Science. He graduated consequently at Technical University in Brno (1995–MSc.), UTB in Zlin (2001–PhD) and again at Technical University in Brno (2004–assoc. prof.) and VSB-TU (2010-professor). Currently, he is a professor at the Department of Computer Science and in total, he has been the supervisor of more than 50 MSc. and 25 Bc. diploma thesis. Ivan Zelinka is also supervisor of doctoral students including students from the abroad. Ivan Zelinka is a member of British Computer Society, Editor in chief of Springer book series: Emergence, Complexity and Computation (http://www.springer.com/series/10624), Editorial board of Saint Petersburg State University Studies in Mathematics, a few international program committees of various conferences and international journals. He is the author of journal articles as well as of books in Czech and English language and one of three founders of TC IEEE on big data http://ieeesmc.org/about-smcs/history/2014-archives/44-about-smcs/history/2014/technical-committees/204-big-data-computing/ . He is also head of research group NAVY http://navy.cs.vsb.cz. He can be contacted at email: ivan.zelinka@vsb.cz.