Spyware Detection Technique Based on Reinforcement Learning

Sergii Lysenko[1[0000-0001-7243-8747]], Kira Bobrovnikova[1[0000-0002-1046-893X]], Peter Popov[2[0000-0002-3434-5272]], Viacheslav Kharchenko[3[0000-0001-5352-077X]], Dmytro Medzatyi [1[0000-0002-1879-2945]]

[1]Khmelnitsky National University, Khmelnitsky, Ukraine
[2]City University, London, Great Britain
[3]National Aerospace University "Kharkiv Aviation Institute", Kharkiv, Ukraine
sirogyk@ukr.net
bobrovnikova.kira@gmail.com
p.t.popov@city.ac.uk
v.kharchenko@csn.khai.edu
medza@ukr.net

Abstract. Analysis of the antivirus technologies, showed that they are not able to detect new spyware with high efficiency, which significantly reduces the reliability and efficiency of its identification. Techniques based on heuristic analysis have a high rate of false positives. The paper presents a new technique for the spyware detection method in computer systems that provides a principle of proactivity and is based on mechanisms machine learning with the reinforcementlearning. The suggested method of spyware detection is based on software behavior analysis in computer systems. The suggested method involves the computer systems monitoring concerning the software, operates with the behavior.

Keywords: Spyware, Malware, Cyberattack, API, Machine Learning, Reinforcement Learning, Network, Cybersecurity, Computer system, Host, Detection

1 Introduction

Today spyware is one of the most common threats on the Internet to businesses and individual users, since it can steal sensitive information and harm the network [1-3]. Spyware is a type of malware that gathers and relays personal information it to advertisers, data firms, or external users without the knowledge and consent of the data owners. There are four main types of spyware: adware, trojan, tracking cookies, system monitors [1]. They use tracking functions to send various private information, such as a list of visited websites, user's contact email addresses or keystrokes on a keyboard, screenshots, online activities on computers or mobile devices. Meanwhile, data obtained by spyware may contain PIN codes, security codes, credit card numbers, etc. Also, spyware can activate cameras and microphones to watch and listen to

users undetected [1-4]. Some types of spyware can use unauthorized analysis of the state of security systems, scan ports and vulnerabilities [1].

Some types of spyware are able to install other additional malware or removing certain programs and modify the parameters of the operating systems. In addition, this kind of malware can redirect browser activity, which entails visiting websites blindly with the risk of virus infection.

Some types of viruses can bring spyware along for the ride as they spread [3]. Also, spyware can be included with free versions of the program in order to study the requirements and interests of users and a revenue obtaining from software sales [1-3].

The main problem is that there is no clear border between benign and malicious spyware applications when the received information is used to the detriment.

2 Related works

Today solutions to the problems of security are widely presented in the literature. One of the ways to prevent and defend computer system from the spyware spread is to construct resilient systems [4], use honeynets [5], implement IDS and security case assessment [6]. The paper [7] presents a review of different spyware detection approaches and tools. Also, a behavioral-based machine learning technique which used high-level architecture for monitoring and filtering of outgoing packets was proposed.

The work [8] is devoted to investigation concerning the undocumented API calls and middleware libraries, which are used by the malware creator to steal the user information remotely by injecting into the process and how hide them from the anti-malware protector. The experimental results of the proposed work showed that the antimalware protector need to take more attention on API call hooking at network level injection by X-cross languages.

In the paper [9] a cyber kill chain based taxonomy of banking trojans features was proposed. This threat intelligence is based on the taxonomy providing of a stage-by-stage operational understanding of a cyber-attack. It presented the beneficial to security practitioners and the design of the evolutionary computational intelligence on trojans detection and mitigation strategy.

The paper [10] provides a technique for detection the stealth and obfuscated spyware and ransomware, including keyloggers, screen recorders, and blockers. The proposed method is based on a dynamic behavioral analysis through deep and transparent hooking of kernel-level routines. This paper also presents the anti-spyware application to track spyware footprints in order to detect and force terminate running processes, eliminate executable files, and restrict network communications.

The works [11-14] devoted to keyloggers spyware detecting. In [11] a new detection technique that is able to detect the present keyloggers in PC using Support Vector Machine learning algorithm was presented. In the paper [12] a logging and testing technique to detect the active keylogger attack was proposed. In the work [13] a strategy based on the detection manner techniques for userspace keyloggers by matching I/O of all processes with some simulated activity of the user was proposed.

The paper [14] presents a survey of keylogger and screenlogger attacks by the covering basic concepts related to bank information systems and explaining their functioning, as it presents and discusses an extensive set of plausible countermeasures.

The papers [15, 16] aim to detect spyware threat in mobile Android environment. The approach proposed in [15] based on model checking technique and involvement temporal logic formulae to identify spyware behaviors.

In the work [16] Android spyware detection approach based on a feature transformation was proposed. This approach transmutes a known malware features into another domain of features by means using of new feature transformations types. This allows distinguish three types of malware: rootkit, spyware, and banking trojans from other malware types and benign applications.

The articles [17-19] present the approaches for malware detection based on the software obfuscation techniques and its behavior analysis.

In [20] an approach concerning the modelling of the cyber-attacks' effects on the software reliability is presented. In [21] sustainability issue related to software and IT reliability, safety and security are discussed.

Nevertheless, the mentioned above approaches have common drawbacks: they don't take into account a set of features, that may assign benign and malicious spyware clearly.

3 Spyware Detection Technique based on Reinforcement Learning

A new spyware detection technique based on reinforcement learning is proposed. It is a proactive approach for the malware detection, and allows detecting all types of spyware. The technique is based on mechanisms machine learning and is able to detect new unknown spyware.

The suggested method of the spyware identification uses software behavior analysis in the computer systems.

The main steps of the proposed approach are presented below:

1. Spyware sample construction.
2. Usage of the reinforcement learning algorithm, the rewards evaluation.
3. Computer systems monitoring concerning the software behavior.
4. Features selection that may indicate the presence of spyware in the computer systems.
5. Evaluation of the reward for research object.
6. Comparison of the obtained rewards with the rewards values of the known spyware.

Let us consider the steps of the method in more detail.

3.1 Computer systems monitoring concerning the software behavior

In order to conduct the monitoring stage, we are to investigate the spyware functioning in the computer system. Thus, let us present it as a tuple:

$$N = \langle Q_1, S_1, H_1, G_1, I_1, B_1, W_1 M_1, E_1 \rangle, \quad (1)$$

where $Q_1 = \{q_j\}$ – a set of malware's actions that tells an attacker about its presence on the network; $S_1 = \{s_j\}$ – a set of actions to penetrate the operation system of the computer system via copying its body into the system directory; $H_1 = \{h_j\}$ – a set of actions that register the spyware's body into the system registry in order to automatically start after the operating system restarting; $G_1 = \{g_j\}$ – a set of actions for the messages sending, g_e – via e-mail; g_m – via messengers, $g_e, g_m \in G_1$; I_1 – an operating memory of the computer system; $B_1 = \{b_j\}$ – a set of actions when an attacker begins to tap a specified TCP port; $W_1 = \{w_j\}$ – a set of actions by which spyware accesses the command line of the victim's computer (only works when receiving the message); $M_1 = \{m_j\}_{j=1}^{4}$ – a set of spyware lifecycle stages; $E_1 = \{e_j\}$ – a set of spyware functions, which is determined by spyware lifecycle stages. Let us present the function of the infection, $m_1^1 \Rightarrow Y \overset{f_1^1}{\rightarrow} \{t_{in} | t_{in} \in T\}$, where Y – a set of malicious spyware actions, T - a set of the computer systems in the network, t_{in} – an infected computer system with the spyware. Let u present the function of copying to the operating system directory and register itself in the system registry $m_2^1 \Rightarrow S_1 \overset{f_2^1}{\rightarrow} \{h | h \in H_1\}$; a command to re-give the collected data an attacker to the attacker to listen to the specified TCP/IP port, $m_3^1 \Rightarrow B_1 \overset{f_3^1}{\rightarrow} I_2'$; an access function to a command line of the victim computer $m_4^1 \Rightarrow W_1 \overset{f_4^1}{\rightarrow} \{g | g \in G_1\}$.

3.2 Usage of the Reinforcement Learning

The proposed technique is based on the usage of the reinforcement learning. It is a new approach in the machine learning. The main idea of it is learn how a research object's states are being changed under the specified action in the situation when there is a single reward feature. During the learning stage aims the achievement of the target via usage of a reinforcement factor finding the most optimal object's action to be performed in each its state. The action is chosen anyway. And after that, the agent is receiving a new state or a reward. Iterating this process, the specified agent is being learned that the best action to be obtained is to the expect maximum value as in a form of rewards [22].

Primarily, in each iteration the agent is to be changed its current states s, s∈S and is to select a specified action a, a∈A. During the state changes the actor also is able to receive some reward signal with value r, r∈R. In these iterations, in order to get a useful experience in regards to states, agent, the transition and reward values, the agent demands best action and the system evaluation corresponding to the learning procedure [22, 23].

An effective approach for the reinforcement learning usage is the temporal difference technique [24]. It doesn't involve the environment model of learning and is able to evaluate the additive calculation with high efficiency.

This approach operates with the agent which is everting to evaluate the value function on the base of getting the step-by-step reward and the defined reward for the further object's state which it obtains.

In this point of view, the most useful and simple method is the Q-learning approach [24, 25]. It employs the background information received from object's state in the environment, and renews such information about its state in a special Q table. Such table presents the set of pairs - the state *S* and action *a*, and contains *Q(S, a*) for each pair. In addition, it involves information concerning each object's state changes from S_t to S_{t+1}, and the reward value r_{t+1}, renews the Q table according to formula:

$$Q(S_t, a_t) = Q(S_t, a_t) + \alpha[r_{t+1} + \gamma \max_a Q(S_{t+1}, a_t) - Q(S_t, a_t)]. \quad (2)$$

It means, that whether the agent is selecting the specified action *a* being in the specified state S and further is changing it to state S′, it is obtaining the maximum Q value of next state.

The employment of the reinforcement learning enables the spyware detection via the usage of the feedback of the reward obtaining, which is able to present data in a form, that enables a high efficiency identification [24].

Spyware detection technique based on reinforcement learning operates with m specified research object's states – the set of variants of the API functions, that perform malicious activity. Thus, the set of research object's states can be presented as $S=\{Si, | i = 1, m\}$.

In this case, object's states present the properties of the Markov chains, so it contains the state's information corresponding to the past and the present, and it is important for the training procedure.

In order to perform spyware detection, the technique has to identify n actions – the changes concerning executable.

In the approach we deal with the different categories API calls used for the spyware execution in the computer system:

1. Registry (registry manipulation for spyware installation, activation, stealth functioning etc).
2. Network management (managing the network related commands).
3. Memory management (managing the system memory for its hiding, temporary presence in the system etc).
4. File i/o (file management with stolen information).
5. Socket (socket related commands).
6. Processes and threads (operations with registers, threads for executing main spyware functioning).
7. Dynamically linked libraries (dll manipulations).

After the agent has chosen the action it had obtained an answer from the environment - reward or punishment. It is a of the environment's feedback corresponded with the mentioned system action.

Reinforcement learning deals with quality function. It is the presentation of the optimal action's state value, correlated with other actions' states. The quality function value dependents on both the state's and action's values, e.g. Q(s, a) demonstrates the $Q(s_1,a_2)$ the optimal action's state value of action a_2 in the state s1.

Based on the quality function it's chosen the optimal action's state value concerning the current state, which is used for the learning process.

Employing the reinforcement learning as the approach to the spyware detection, let us define the correspondence between its notions (environment, action, reward and agent) and spyware detection items.

Let's assume the environment item as the representation of the system monitor, able to extract spyware's features, which are to be analyzed.

The result of the analysis is the conclusion about the spyware's presence or absence in the computer system, and it is used for the reward definition.

The reward approach in the reinforcement learning is able to categorize objects via supervised learning. It answers to agent a question in what way the learning process may move on concerning object's actions.

The mechanism of rewarding has to present the changes that the environment has executed subsequently the agent implements the action in the current state. Also. It demonstrates the achievement process to the target – maximum value of the detection efficiency.

After that, the pair state/reward are sent to the agent. The agent obtains the set of features that may indicate the spyware's presence and the reward and uses them for the system learning via Q-learning algorithm [26].

Algorithm 1 operates with the parameters α, which defines the learning speed. In practice, its optimal value is 0.1, is usually recommended for the learning rate, and we have used this value as well, and γ – the discount factor, which is used for the divergence prevention with the value 0.9. The aim off learning is to achieve the optimal action's quality value $Q_{max}(s',a')$ for the new system's states.

The situation, when the agent has achieved needed values, that is object is the spyware on benign. While these values haven't achieved, the algorithm is repeated, otherwise it is terminated.

The Q-learning algorithm as a training tool involves the steps,

1. Initialization of the Q(A, s) value to zero for all states and actions.
2. Obtaining of the system's current state.
3. Choosing procedure for the object's action on the base of algorithm.
4. The object action performance and gathering of the information via monitoring and obtaining the rewards concerning this object's action.
5. The obtaining of the new system state (S') caused by the object's action performance.
6. The evaluation of the values Q (A, s) using formula:

$$Q(S_t, a_t)=Q(S_t,a_t)+\alpha[r_{(t+1)}+\gamma \max_a (Q(S_{(t+1)},a\_t)-Q(S_t,a_t))], \quad (3)$$

where α=[0,1] is the learning rate.

The scheme of the spyware detection process using the reinforcement learning functioning is presented in Fig.1.

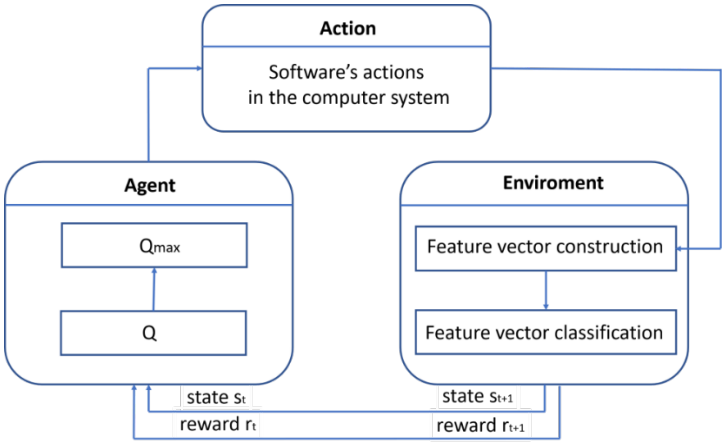

Fig. 1. The scheme of the spyware detection process using the reinforcement learning

4 Experiments

In order to assess the efficiency of the proposed technique a number of experiments were carried out. For the purpose of the classifier training, the API call sequence of spyware samples and benign application from dataset [27, 28] were used.

The training dataset included 16350 malicious samples and 14571 benign samples. All malicious samples were divided into 6 spyware classes: adware, keyloggers, infostealers, red shell spyware, tracking cookies, rootkits.

In order to determine effectiveness of the proposed spyware detection technique 17231 spyware samples from evaluation dataset [29-32] were employed. Also, the test data contained 16298 samples of benign applications.

The experimental results were estimated using standard detection metrics [33-40]: sensitivity (SN); specificity (SP); overall accuracy (E); True Positives (TP); True Negatives (TN); False Positives (FP); False Negatives (FN).

$$SN = TP/(TP + FN),\ SP = TN/(TN + FP),\ E = (TP + TN)/(TP + TN + FP + FN) \quad (4)$$

Distribution of the malicious samples that involved the experiments by spyware classes and results of experiments are presented in the Table 1.

The experimental results show that usage of the reinforcement learning demonstrated ability to detect spyware with the efficiency in the range of 95,63 to 99,05% with the false positives in the range of 0,02 to 0,28%.

Table 1. Test result of experiments for different spyware types

Spyware types	Number of spyware samples	Evaluation set				Results		
		Malicious samples		Benign samples		Sensitivity, %	Specificity, %	Overall accuracy, %
		TP	FN	TN	FP			
adware	2419	2954	58	2872	7	98,07	99,76	98,90
keyloggers	2693	2852	62	2639	4	97,87	99,85	98,81
infostealers	2874	2937	41	2597	12	98,62	99,54	99,05
red shell spyware	2941	3002	39	2853	36	98,72	98,75	98,74
tracking cookies	2745	2787	24	2739	47	99,15	98,31	98,73
rootkits	2678	2699	211	2598	31	92,75	98,82	95,63
Total number	16350	17231	435	16298	137	-	-	-

5 Conclusion

The work presents the technique for the spyware detection method in computer systems that provides a principle of proactivity and is based on mechanisms of machine learning with the reinforcement. The suggested method of spyware detection uses software behavior analysis in computer systems. The suggested method involves the computer systems monitoring concerning the software, operates with the behavior.

Experimental research demonstrated a promising result of the detection ability of the proposed approach, while the false positives rate is low.

The further work may be devoted to the development of the techniques that involve other machine learning algorithms and spyware features analysis.

References

1. Norton. What is spyware? And how to remove it. Available online: https://us.norton.com/internetsecurity-how-to-catch-spyware-before-it-snags-you.html (accessed on March 20, 2020).
2. Eset. Spyware. Available online: https://help.eset.com/glossary/en-US/spyware.html (accessed on March 20, 2020).
3. Avast. Spyware: Detection, Prevention, and Removal. Available online: https://www.avast.com/c-spyware (accessed on March 20, 2020).
4. Drozd, O., Kharchenko, V., Rucinski, A., Kochanski, T., Garbos, R., Maevsky, D. Development of Models in Resilient Computing, Proc. of 10th IEEE International Conference on Dependable Systems, Services and Technologies, pp. 2-7 (2019).
5. Sokol, P., Zuzčák, M., Sochor, T. Definition of attack in the context of low-level interaction server honeypots. Lecture Notes in Electrical Engineering 330, 499-504 (2015).
6. Potii, O., Illiashenko, O., Komin, D. Advanced Security Assurance Case Based on ISO/IEC 15408. In: Zamojski W., Mazurkiewicz J., Sugier J., Walkowiak T., Kacprzyk J.

(eds) Theory and Engineering of Complex Systems and Dependability. DepCoS-RELCOMEX. Advances in Intelligent Systems and Computing, Springer, Vol. 365, pp. 391-401 (2015).
7. Afzulpurkar, A., Alshemaili, M., Samara, K. Outgoing Data Filtration for Detecting Spyware on Personal Computers. In: International Conference on Emerging Internetworking, Data & Web Technologies. Springer, Cham, pp. 355-362 (2019).
8. Prabhavathy, M., Maheswari, S. U. Remote Network Injection Attack Using X-Cross API Calls. In: International Conference on Emerging Current Trends in Computing and Expert Technology. Springer, Cham, pp. 1399-1404 (2019).
9. Kiwia, D., Dehghantanha, A., Choo, K. K. R., Slaughter, J. A cyber kill chain based taxonomy of banking Trojans for evolutionary computational intelligence. Journal of computational science, Vol. 27, pp. 394-409 (2018).
10. Javaheri, D., Hosseinzadeh, M., Rahmani, A. M. Detection and Elimination of Spyware and Ransomware by Intercepting Kernel-Level System Routines. IEEE Access, 2018, 6: 78321-78332 (2018).
11. Pillai, D., Siddavatam, I. A modified framework to detect keyloggers using machine learning algorithm. International Journal of Information Technology, 11.4: 707-712 (2019).
12. Mallikarajunan, K. N., Preethi, S. R., Selvalakshmi, S., Nithish, N. Detection of Spyware in Software Using Virtual Environment. In: 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI). IEEE, pp. 1138-1142 (2019).
13. Wajahat, A., Imran, A., Latif, J., Nazir, A., Bilal, A. A Novel Approach of Unprivileged Keylogger Detection. In: 2019 2nd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). IEEE, 2019. pp. 1-6 (2019).
14. Sbai, H., Goldsmith, M., Meftali, S., Happa, J. A survey of keylogger and screenlogger attacks in the banking sector and countermeasures to them. In: International Symposium on Cyberspace Safety and Security. Springer, Cham, pp. 18-32 (2018).
15. Fasano, F., Martinelli, F., Mercaldo, F., Nardone, V., Santone, A. Spyware Detection using Temporal Logic. In: 5th International Conference on Information Systems Security and Privacy, ICISSP 2019. SciTePress, pp. 690-699 (2019).
16. Han, Q., Subrahmanian, V. S., Xiong, Y. Android Malware Detection via (Somewhat) Robust Irreversible Feature Transformations. IEEE Transactions on Information Forensics and Security (2020).
17. Savenko, O., Lysenko, S., Nicheporuk, A., Savenko, B. Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search. CEUR-WS, ISSN: 1613–0073, Vol. 1844, pp. 555-569 (2017).
18. Savenko, O., Lysenko, S., Nicheporuk, A., Savenko, B. Approach for the Unknown Metamorphic Virus Detection. The 9-th IEEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications: Proceedings, vol 1, pp. 453-458 (2017).
19. Lysenko, S., Savenko, O., Bobrovnikova, K., Kryshchuk, A. Self-adaptive system for the corporate area network resilience in the presence of botnet cyberattacks. Communications in computer and information science, pp. 385-401 (2018).
20. Popov, P. Models of Reliability of Fault-Tolerant Software Under Cyber-Attacks. IEEE 28th International Symposium on Software Reliability Engineering, pp. 228-239 (2017).
21. Kharchenko, V., Kondratenko, Y., Kacprzyk, J. (eds). Concepts of Green IT Engineering: Taxonomy, Principles and Implementation. Green IT Engineering: Concepts, Models, Complex Systems Architectures. Studies in Systems, Decision and Control, Springer, Cham, Vol. 74, pp. 3-19 (2017).

22. Morales, E.F., Zaragoza, J.H. An introduction to reinforcement learning', in Sucar, L.E., Morales, E.F. and Hoey, J. (Eds.): Chapter in Decision Theory Models for Applications in Artificial Intelligence: Concepts and Solutions, Vol. 19, No. 4, pp.639-668 (2012).
23. Jagielski, M., Oprea, A., Biggio, B., Liu, C., Nita-Rotaru, C., Li, B. Manipulating machine learning: Poisoning attacks and countermeasures for regression learning. In 2018 IEEE Symposium on Security and Privacy (SP), IEEE, pp. 19-35 (2018).
24. Buffet, O., Dutech, A., Charpillet, F. Shaping multi-agent systems with gradient reinforcement learning. Autonomous Agents and Multi-Agent Systems, Vol. 15, No. 2, pp.197-220 (2007).
25. Gomes, E. R., Kowalczyk, R. Dynamic analysis of multiagent Q-learning with e-greedy exploration. In Proc. Int. Conf. Mach. Learn., pp. 369-376 (2009).
26. Matiisen, T. Demystifying deep reinforcement learning. Computational Neuroscience LAB, 19 (2015).
27. VirusShare. Available online: https://virusshare.com/ (accessed on March 20, 2020).
28. Nexginrc. Datasets. Available online: http://nexginrc.org/Datasets/Default.aspx (accessed on March 20, 2020).
29. Kaggle. Microsoft Malware Classification Challenge (BIG 2015). Classify malware into families based on file content and characteristics. Available online: https://www.kaggle.com/c/malware-classification (accessed on March 20, 2020).
30. Ronen, R., Radu, M., Feuerstein, C., Yom-Tov, E., Ahmadi, M. Microsoft malware classification challenge. arXiv preprint arXiv:1802.10135 (2018).
31. GitHub. Malware dataset for security researchers, data scientists. Public malware dataset generated by Cuckoo Sandbox based on Windows OS API calls analysis for cyber security researchers. Available online: https://github.com/ocatak/malware_api_class (accessed on March 20, 2020).
32. Catak, F. O., Yazı, A. F. A Benchmark API Call Dataset for Windows PE Malware Classification. arXiv preprint arXiv:1905.01999 (2019).
33. Pomorova, O., Savenko, O., Lysenko, S., Kryshchuk, A., Bobrovnikova, K. A technique for the botnet detection based on DNS-traffic analysis. Communications in Computer and Information Science, Vol. 522, pp. 127-138 (2015).
34. Grażyna Suchacka, Jacek Iwański. Identifying legitimate Web users and bots with different traffic profiles - an Information Bottleneck approach. Knowledge-Based Systems, Vol. 197, 105875 (2020).
35. Komar M., Sachenko A., Golovko V., Dorosh V. Compression of Network Traffic Parameters for Detecting Cyber Attacks Based on Deep Learning. Proceedings of IEEE 9-th International Conference on Dependable Systems Services and Technologies, pp. 44-47 (2020).
36. Obeidat, I., & AlZubi, M. Developing a faster pattern matching algorithms for intrusion detection system. International Journal of Computing, 18(3), 278-284, (2019).
37. Canadian Institute for Cybersecurity. Malware dataset, https://www.unb.ca/cic/datasets/botnet.html (accessed January 10, 2020).
38. API Monitor. Availabe: http://www.rohitab.com/apimonitor (accessed January 10, 2020).
39. Lysenko, S., Bobrovnikova, K., Savenko, O., Kryshchuk, A. BotGRABBER: SVM-based self-adaptive system for the network resilience against the botnets' cyberattacks. Communications in computer and information science, pp. 127-143 (2019).
40. VX Heavens Computer virus collection. Availabe: http://vx.netlux.org (accessed January 10, 2020).