Proof of Stake And Proof of Work Approach for Malware Detection Technologies

Dmytro Denysiuk[a], Tomas Sochor[b] and Mariia Kapustian[a]

[a] *Khmelnytskyi National University, Institutska str., 11, Khmelnytskyi, 29016, Ukraine*
[b] *Prigo University, V. Nezvala 801/1, 736 01 Havířov, Czech Republic*

Abstract

Malware detection is critical given the rapid spread of malware on the Internet as it functions as an early warning system for computer security against malware and cyberattack. This keeps hackers away from the computer and prevents information from being compromised. Existing antivirus software and hardware are unable to effectively detect new or modified old classes of viruses, and are prone to a large number of false positives. Therefore, the problem of malware detection requires an immediate solution to ensure the safe use of the network. Thus, there is a need to develop new methods of analyzing potentially dangerous code in order to detect malicious software.

To solve this problem, a Proof of stake and Proof of work approach for malware detection technologies based on the use of Blockchain technology was developed. A mechanism has been implemented to remove features that may indicate that a potentially dangerous code belongs to a certain class of malware, as well as a mechanism that analyzes potentially dangerous code, carried out in parallel by different network participants using Proof of work. By using the concept of Proof of work, the developed method provides accelerated analysis of potentially dangerous codes. The use of the concept of Proof of stake provides an opportunity to increase the accuracy of malware detection by validating the results of the participant's analysis, taking into account the coefficient of efficiency of the participant's computing resources by the method of soft voting. In the key of using blockchain technology, validation provides an opportunity to prevent the use of analysis results from a potentially compromised participant. The use of the concept of Proof of stake provides an opportunity to increase the accuracy of malware detection by validating the results of the participant's analysis, taking into account the coefficient of efficiency of the participant's computing resources by the method of soft voting. In the key of using blockchain technology, validation provides an opportunity to prevent the use of analysis results from a potentially compromised participant. The use of the concept of Proof of stake provides an opportunity to increase the accuracy of malware detection by validating the results of the participant's analysis, taking into account the coefficient of efficiency of the participant's computing resources by the method of soft voting. In the key of using blockchain technology, validation provides an opportunity to prevent the use of analysis results from a potentially compromised participant.

The application of the developed approach makes it possible to detect malicious software of different classes with an accuracy of 98.81- 99.33%.

Keywords

malware, malware detection, cybersecurity, Blockchain, Proof of stake, Proof of work

1. Introduction

With the development of the Internet, there is a need to develop systems to protect personal data from attacks by malicious software. According to a study by Avast [1] the number of cyberattacks on

IntelITSIS'2022: 3rd International Workshop on Intelligent Information Technologies and Systems of Information Security, March 23–25, 2022, Khmelnytskyi, Ukraine
EMAIL: denysiuk@khmnu.edu.ua (D. Denysiuk); tomas.sochor@prigo.cz (T.Sochor); kapustian.mariia@gmail.com (M.Kapustian)
ORCID: 0000-0002-7345-8341 (D. Denysiuk); 0000-0002-1704-1883 (T.Sochor); 0000-0001-9200-1622(M.Kapustian)

business increased from 11.25% in 2020 to 13.9% in 2021. The World Economic Forum called cyberattacks the fifth most dangerous. According to a report by Microsoft Defender [2], 77% of cyberattacks were carried out on large and small businesses. As most of the country's budgets are filled with small and medium-sized businesses, these sections of the economy need to develop innovative methods to prevent cyberattacks by malware, as well as new methods to quickly detect malware to prevent further attacks.

Despite the large number of known methods of protecting information systems, they are under attack from well-known and new families of malwares, such as crypto miners, spyware, backdoors, banking trojans and droppers. These malware has a wide range of capabilities aimed at compromising information systems. Therefore, in order to ensure the integrity, confidentiality and accessibility of information in information systems, the problem of detecting malicious software in these systems is important.

2. Related works

Today, various approaches to the detection of malware are widely represented in scientific sources. [3] presents an API detection approach that classifies malware based on user feedback. However, in the case of sensitive resources that require a significant portion of permits, the approach may increase the number of false alarms. In [4], an approach is proposed that uses a combination of permissions and intentions, supplemented by several stages of classifiers, to detect APS. Decision tables, a multilayer perceptron, and decision trees are combined using three schemes: determining the mean of the probabilities, the product of the probabilities, and the majority of votes.

In [5], a method for detecting malware based on the analysis of system call logs is proposed. The results of the experiments showed high detection accuracy, but the authors did not take into account the ability of some applications to identify sandbox-type environments.

[6] proposed a system for detecting malware, which uses a deep convolutional neural network (CNN). The classification of malware is based on static analysis of the raw code sequence from the disassembled program.

In [7] the authors propose a system based on static analysis, which operates in four stages. It first builds a call graph for each application, then retrieves the API call sequences using all the unique nodes, and then assigns each call to a specific class, packet, or family. The third stage involves modeling the behavior of each application by constructing Markov chains from sequences of API calls, with the transition probabilities used as a feature vector, provide an opportunity to classify the application as benign or malicious software.

In [8], a framework was developed that uses triage to rank applications based on their potential risk. The approach combines a probabilistic model for predicting the existence of information flows with an indicator of how significant the flow is in benign and malicious applications. The results of experiments show that the approach is able to accurately predict the availability of information flows and provides significant savings.

In [9], a method for detecting malware using both static and dynamic analysis is proposed. In particular, the method uses traditional features (such as permissions and API calls) in order to increase the efficiency of malware detection based on static analysis. Also, in order to bring the features obtained from call graphs of different sizes to the same dimension, the proposed approach used the methods of feature selection and clustering.

In [10], a study was conducted aimed at the possibility of using features based on content and relationships to identify malware; modeling different types of entities (such as file, archive, machine, API, DLL) and rich semantic relationships between these entity types (ie file-archive, file-machine, file-file, API-DLL, file-API).

Based on the study, a structured heterogeneous information network (HIN) was built and a metagraph-based approach was presented, which provides an opportunity to show the relationship and affinity between files. In order to be able to measure the affinity of files on the constructed HIN in order to detect RRP, it is necessary to use effective methods of studying hidden representations for HIN. To solve this problem, a new model of metagraph2vec built into HIN is proposed on the basis of the constructed schemes of metagraphs.

In [11] the possibility of using evolutionary calculation methods is investigated both for the development of new variants of malware, which successfully avoids SPP protection systems based on static analysis, and for automatic development of better solutions for protection against malware.

In [12], a new method of detecting malware based on the analysis of information flows in order to identify existing patterns of behavior and related flows that have common computational pathways. Such complex flows, with their structure, regularities and relations, accurately capture the complex behavior demonstrated by both malware and good-quality programs. The analysis of N-gram API calls available in these complex streams is used to identify unique and common patterns of behavior.

The approach proposed in [13] uses discriminatory adversarial competition (DAN) with deep training to classify applications as malicious or benign according to three sets of features: raw operation codes, permissions and API calls. The proposed approach provides an opportunity to detect malicious programs that use obfuscation methods to evade detection.

A review of the literature has shown that the problem of detecting malware is extremely relevant. Known methods of detecting malware have a high level of efficiency, but also demonstrate a high rate of false positives. A common weakness of the above approaches is the need for large amounts of computing resources and the fact that they are not able to respond adaptively to known and unknown attacks carried out by malware. Also, the approaches considered have some common shortcomings, which are to ignore the packaged malware and the inability to protect the device from the threats of zero-day and malicious programs that can modify themselves.

2.1. Blockchain in cybersecurity

One of the promising technologies used in cybersecurity is Blockchain technology. Blockchain provides the ability to effectively counter cyber attacks, in particular, provides reliable protection of data from compromise, theft or destruction [14].

Blockchain is a technology of information exchange and storage that works and develops without centralized control. This technology uses peer-to-peer networks (P2P), which allows you to create replicated and distributed registers.

These registers are protected by cryptography by linking blocks to each other. The hacking reliability of Blockchain technology is ensured through the formation of data blocks, using a number of complex calculations and encryption of information (Figure 1).

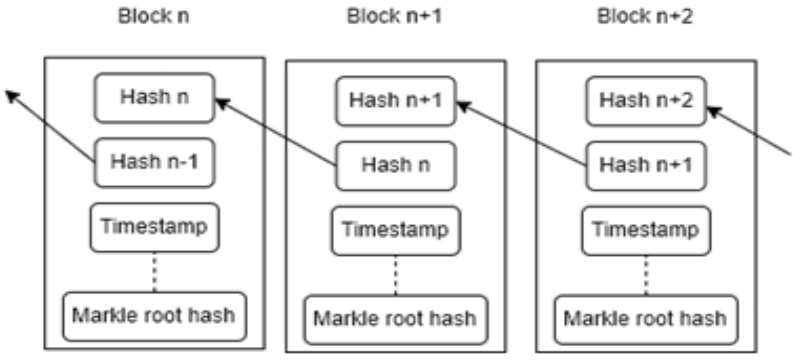

Figure 1: Blockchain structure

Each block is "connected" to the previous one in sequence and is recorded unchanged throughout the network. Cryptographic trust and security technology applies a digital fingerprint or unique identifier to each transaction. Thus, trust, accountability, transparency and security are built into each blockchain. As a result, different partners can access and share the data contained in the blocks. This concept is known as third party trust by consensus (Figure 2).

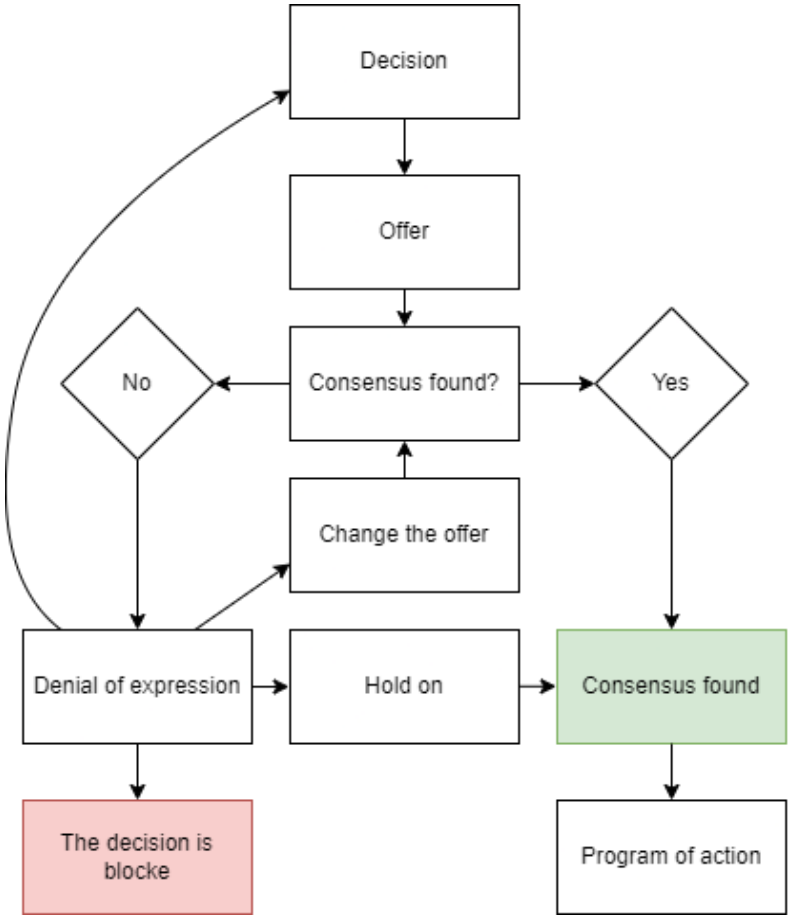

Figure 2: Method of reaching consensus

Cryptographic hash functions are used to effectively protect Blockchain blocks [15]. A hash function H is a function that takes data of any size and converts it to data of fixed size.

The following characteristics are important for determining the stability of a cryptographic hash: (1) resistance to collisions - it is difficult to find two sets of data d_1 and d_2 such that as a result of hashing their original data coincided $H(d_1) = H(d_2)$; (2) resistance to finding the first prototype - for this source it is difficult to find the input data and that the condition is met $H(d_1) = d_2$; (3) resistance to finding the second prototype - for a given input d_1 and output $d_2 = H(d_1)$ it is difficult to find a second input d_3, such that $H(d_3) = d_2$.

The most popular cryptographic hash function used in the blockchain is SHA-2, in particular its variant SHA-256 [16]. The SHA-256 algorithm is a unidirectional function for creating 256-bit fixed-length digital prints. This algorithm is limited to an input length of up to 2^{64} bits.

The complexity of compromising data blocks is due to the decentralization of the system, in which copies of block sequences are owned by all network members. All participants keep an encrypted record of each transaction using a robust decentralized and scalable recording mechanism that cannot be compromised.

An important advantage of the blockchain is the absence of the need for additional overhead or intermediaries. The use of a single decentralized reliable source reduces the cost of secure business transactions between partners. However, partners do not necessarily have to trust each other.

Another feature of blockchain technology is that to compromise the sequence of blocks requires more than 51% of the computing power involved in the formation of new blocks. Thus, this unique technology benefits any number of partners who need real-time shared secure access to transactions. Blockchain technology does not have a single storage location, so there is no central point of vulnerability. This approach improves the security and availability of data for each network participant.

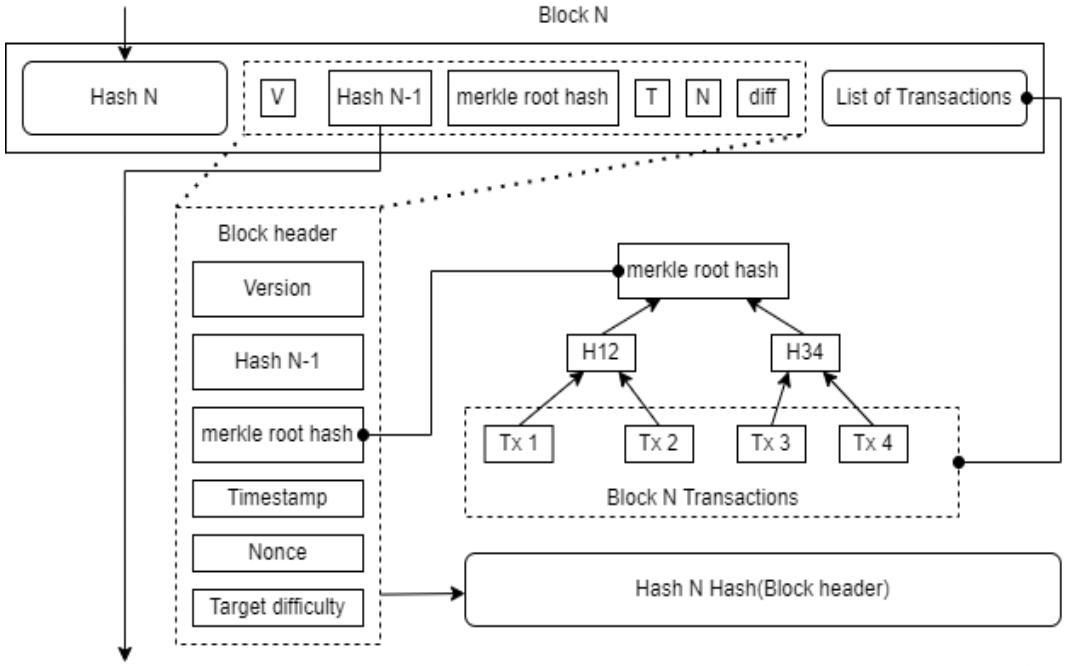

Figure 3: Blockchain data structure with block format

2.2. Proof of work and Proof of stake concepts

An integral part of Blockchain is the concept of Proof of work [17]. Proof of work is a blockchain consensus algorithm that is used to confirm transactions and create new blocks. With Proof of work, participants compete with each other to complete online transactions for a reward. Participants send each other digital tokens, after which the participants' transactions are collected in blocks and recorded in a distributed register (blockchain). The work of network members is based on the calculation of complex mathematical functions and the ability to easily prove that the solution is obtained. The Proof of work mechanism provides the possibility of cryptographic confirmation, in which one network member provides evidence of the task to the verifying party. For its part, the verifying party can easily verify the result of the work, and reward the executing party. This technology also has a different smart contract [18].

With the increase in network capacity, there was a problem in increasing the speed of transactions in it. As the speed of validation decreased as the number of transactions in the blockchain network increased, there was a need to develop new validation methods. Thus, there are prerequisites for the development of the concept of Proof of stake. The concept of the Proof of stake algorithm is to reduce the amount of computational work required to verify blocks and transactions. Proof of stake has changed the way blocks are checked. It is the use of own assets in the system. Participants in the system provide their own assets as collateral, and thus become validators of the system. After receiving the validator status, the network members check the blocks for authenticity. When the unit is tested with a sufficient number of validators, it enters the network,

Figure 4 shows an example of a comparison of Proof of work and Proof of stake algorithms, in which the hash rate of network user capacity is used to validate blocks with the Proof of work algorithm. Depending on the capacity of the participant, he receives the appropriate percentage of the reward.

The Proof of stake algorithm uses the information about the corresponding share of invested resources in the network to distribute the reward, thus using the Proof of stake algorithm reduces the need for increased network capacity.

3. Proof of stake and Proof of work approach for malware detection technologies

The paper presents an adaptive approach to the search for malicious software in networks, which is based on the methods of Proof of work and Proof of stake. In addition, the developed approach uses a variety of machine learning methods to identify potentially dangerous code fragments and assign the code to a particular class of malware or good quality software.

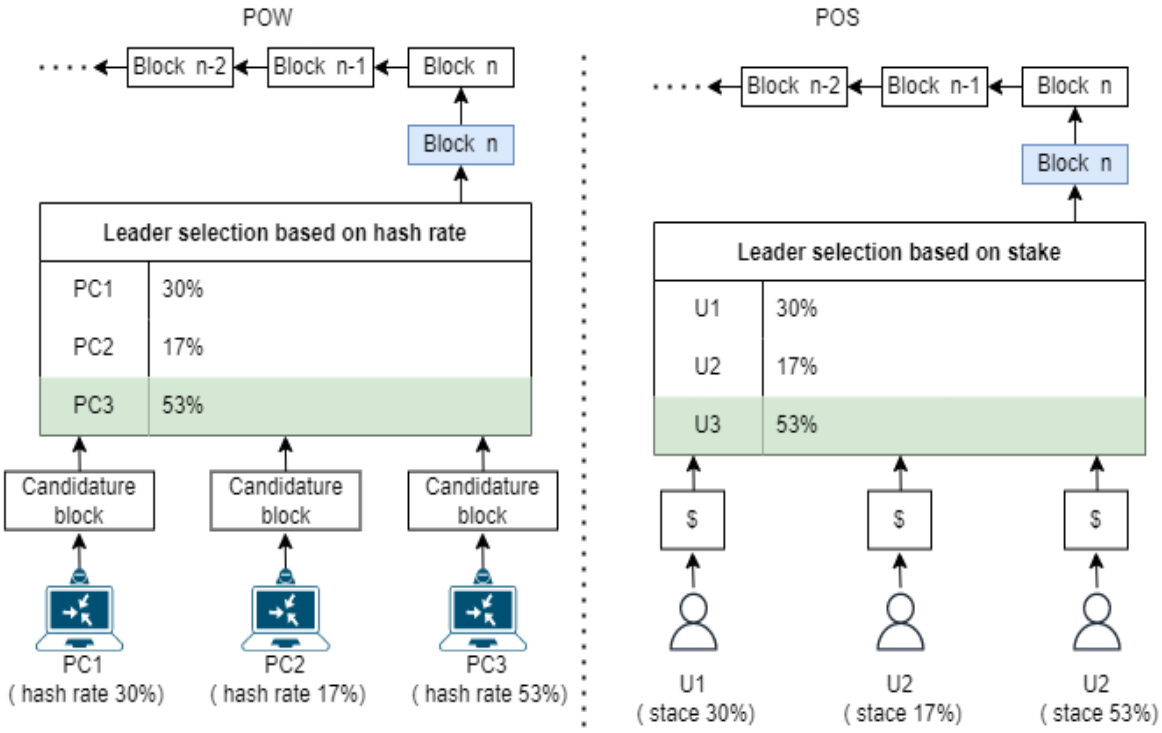

Figure 4: Comparison of POW and POS algorithms

The main idea of the proposed approach is to increase the efficiency of malware detection by parallelizing the analysis of potentially dangerous code fragments on different network participants and preventing the use of analysis results from potentially compromised participants.

Let us denote the set of network participants as $P = \{p_i\}_{i=1}^{X}$, where X is the total number of participants participating in the analysis of potentially dangerous code fragments

Suppose that the proposed approach to the detection of malware uses a certain set of methods M based on machine learning.

To increase the effectiveness of the approach, each network member p operates to analyze potentially dangerous code by a subset of methods $M_o \in M, M_o = \{m_i\}_{i=1}^{N}$, where N is the total number of methods in the subset M_o.

The subset M_o is generated randomly, while the uniqueness of the generated M_o is provided by calculating the checksum e, where $e \notin E$, and where E - the set of checksums calculated for other participants.

For each method from the set M, a method can be defined to represent potentially dangerous code in a form suitable for analysis by this machine learning method by extracting the corresponding features (n-grams, control flow graphs, feature vectors, operating code sequences, etc.).

Representation of a potentially dangerous code snippet in a form suitable for analysis is provided by a set of participants $S = \{s_i\}_{i=1}^{Q}$, where Q is the total number of participants involved in the process of removing features from a potentially dangerous code snippet.

Based on the analysis of a potentially dangerous code fragment, a data block b is generated, which contains the results of the analysis by each method from the set M_o. The data block b can be represented as a tuple:

$$b = \langle c, M_o, L, e \rangle, \quad (1)$$

where c - potentially dangerous code, $L = \{l_i \cup l_{i+1} \cup \dots l_k\} l_k$ - results of analysis of potentially dangerous code fragment for previous participants, where l_k- results of analysis of potentially dangerous code fragment by current participant.

The checksum e can be represented as follows:

$$e = \langle h, N, f \rangle, \quad (2)$$

where h - the result of calculating the hash sum, $\varphi: (c, M_o) \longrightarrow h$, φ - the function of calculating the hash sum; f - additional fields containing the current version of the implementation of the proposed approach.

Figure 5 shows the structure of the data block b.

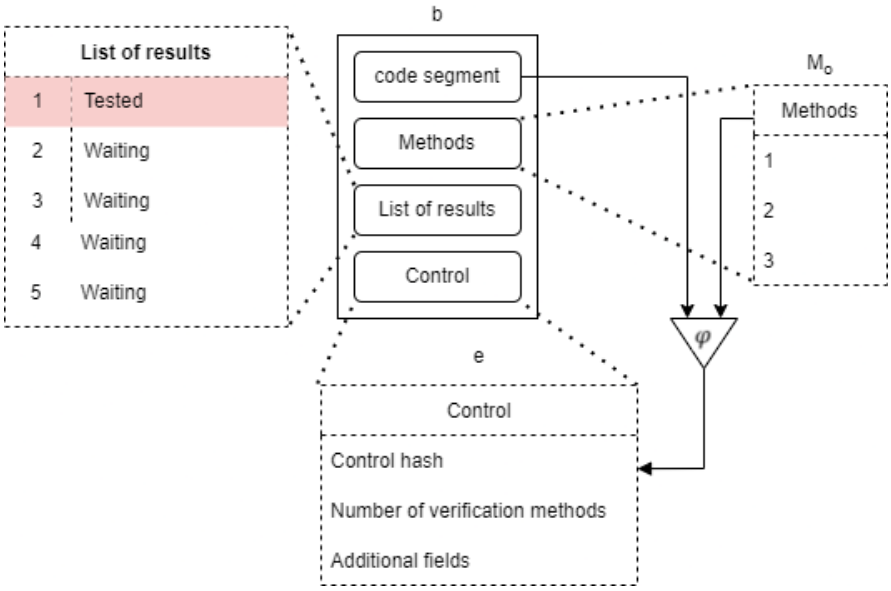

Figure 5: Data block b

Figure 6 shows a simplified diagram of the algorithm for forming blocks.

When performing the analysis of a potentially dangerous fragment of code for its belonging to a certain class of malware, the participant receives the coefficient k of efficiency of the computing resource. This ratio depends on the speed of the analysis, which is related to the power of the participant's resources .

The set of validators $J = \{j_i\}_{i=1}^{D}$, where D is the total number of validators used, guarantees the reliability of the obtained results of the analysis of a potentially dangerous fragment of code.

To this end, each validator j analyzes the list of results L obtained from the participant p.

To implement the resulting conclusion, the method of soft voting (soft voting) is used [19], where the weights of the participants are their coefficient k of the computing resource efficiency.

When a set of blocks to a set of validators J is found, as a result of validation we obtain a conclusion regarding the belonging of a potentially dangerous code to one of the classes of malware or to the class of good quality software.

In the key of using blockchain technology, validation provides an opportunity to prevent the use of analysis results from a potentially compromised participant.

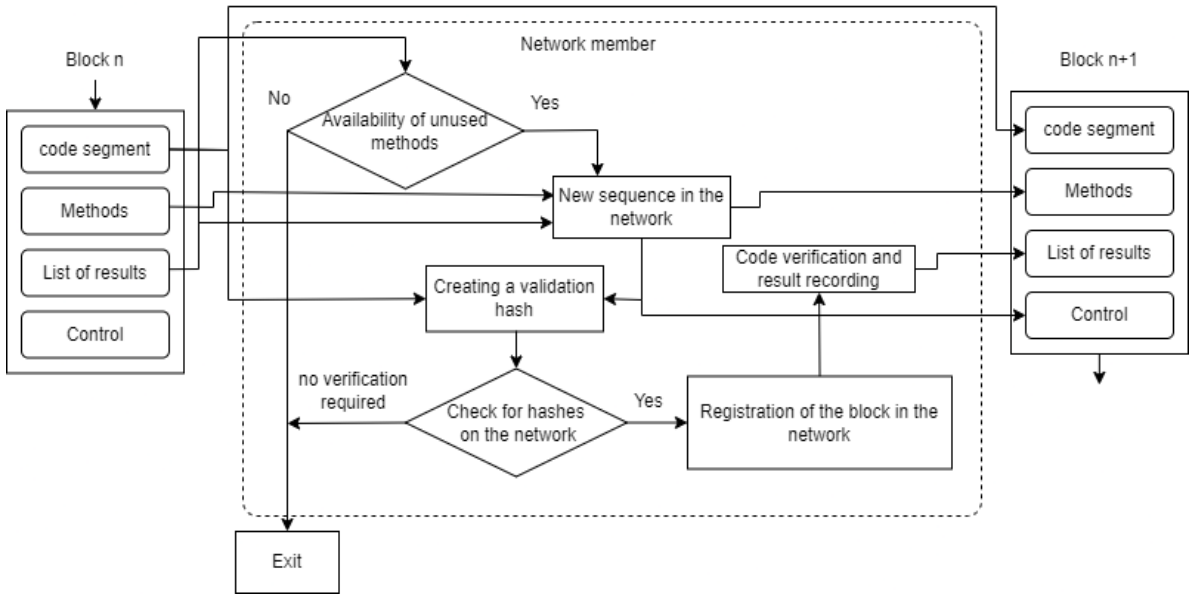

Figure 6: Algorithm for forming blocks *b*

Figure 7 shows a generalized diagram of the validation process.

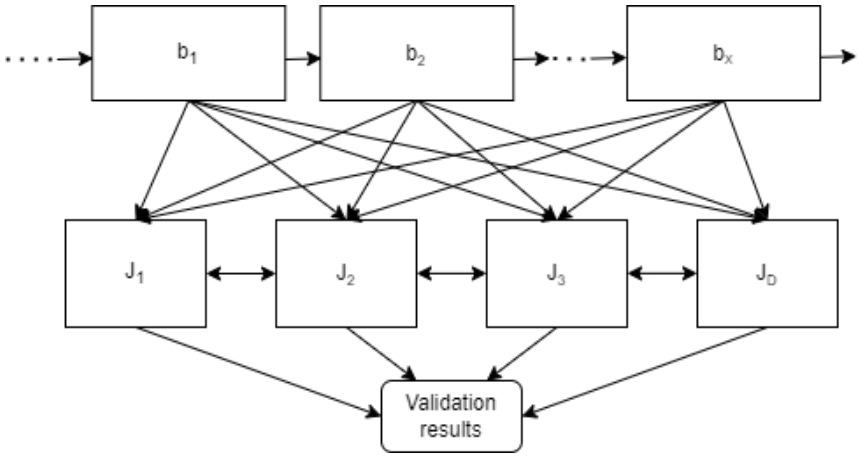

Figure 7: Validation of the result

4. Experiments

A network of 70 computer systems was used to conduct the experiments.

Computer systems were divided into groups according to their functional purpose (a set of participants *S* which remove signs that may indicate that the software belongs to a certain class of malware; a set of participants *P* which analyze potentially dangerous code to determine its affiliation to a certain class of malware or good quality software, a set of validators *J* that validate the results of the analysis of the malware obtained from multiple participants).

The following machine-based methods were chosen as methods for analyzing potentially dangerous code fragments [20-22]: K-Nearest Neighbor, Random Forest, Support Vector Machine, Semi-Supervised Fuzzy C-Means clustering, K-Means, Rotation Forest, Decision Trees [23, 24]

The public data set [26] was used as a database of malware, from which 2781 samples of malware of different classes were used. Samples of good quality software were taken from the Microsoft store [27] in the amount of 2720 units.

The results of the experiments are presented in Table 1.

Experimental results showed that the highest accuracy of malware detection was achieved with the detection of Adware-class malware (99.33%), and the lowest accuracy (98.81%) when detecting a Rootkit class.

Table1

The results of the experiments, TP - True positive, TN - True negative, FN - False positive, FP - False negative

Experiment number	malware class	TP	TN	FN	FP	Overall accuracy,%
1	Adware	371	366	2	3	99.33
2	Trojan	398	395	2	4	99.25
3	Worm	313	300	3	3	99.03
4	Backdoor	357	355	5	2	99.03
5	Dropper	307	301	3	3	99.02
6	Downloader	346	331	2	5	98.98
7	Polymorphic virus	326	320	6	1	98.93
8	Rootkit	333	330	7	1	98.81

5. Conclusions

The paper presents the Proof of stake and the Proof of work approach for malware detection technologies. The method is based on the use of Blockchain technology to increase the efficiency of malware detection by parallelizing the analysis of potentially dangerous code fragments on different network participants and preventing the use of analysis results from potentially compromised participants by involving Proof of work.

Parallelization is achieved through the distribution of participants by functional purpose (participants who remove signs that may indicate that the software belongs to a certain class of malware; participants who analyze potentially dangerous code to determine its belonging to a certain class of malware or good quality software, using methods based on machine learning; validators who validate the results of the malware analysis obtained from the participants).

Validation of the results of the analysis of potentially dangerous code fragments by the participants is carried out with the involvement of the Proof of stake algorithm. The soft voting method is used to determine the final result of the analysis based on the results obtained from the participants. The results of experiments showed high efficiency of detection of malware of different classes using the proposed approach (98.81-99.33%). Further research will focus on finding the most effective ways to divide participants into groups by functional purpose and the most effective network topology.

6. References

[1] A quick look at Avast's latest Global Risk Report for SMBsURL: https://blog.avast.com/global-risk-report-2021-snapshot-avast.

[2] Microsoft Digital Defense Report OCTOBER 2021 URL:https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RWMFIi?id=101738.

[3] Amro, B. Personal Mobile Malware Guard PMMG: a mobile malware detection technique based on user's preferences / B. Amro. - International Journal of Computer Science and Network Security, 2018. - Vol. 18, no. 1. - pp. 18–24.

[4] Idrees, F. Pindroid: a novel android malware detection system using ensemble learning methods / F. Idrees, M. Rajarajan, M. Conti, T. Chen, Y. Rahulamathavan. - Computers & Security, 2017. - Vol. 68. - pp. 36–46.

[5] Chaba, S. Malware Detection Approach for Android systems Using System Call Logs / S. Chaba, R. Kumar, R. Pant, M. Dave. - arXiv preprint arXiv: 1709.0880, 2017.

[6] McLaughlin, N. Deep android malware detection / N. McLaughlin, J. Martinez del Rincon, B. Kang. - Proc. of the Seventh ACM on Conference on Data and Application Security and Privacy, 2017. - pp. 301–308.

[7] Mariconti, E. MaMaDroid: Detecting Android Malware by Building Markov Chains of Behavioral Model / E. Mariconti, L. Onwuzurike, P. Andriotis, E. De Cristofaro. - ACM Trans. Priv. Sec., 2019. Vol. 1, no. 1. - pp. 1–33.
[8] Mirzaei, O. Triflow: Triaging android applications using speculative information flows / O. Mirzaei, G. Suarez-Tangil, J. Tapiador, JMde Fuentes. - Proc. of the 2017 ACM on Asia Conference on Computer and Communications Security, 2017. - pp. 640-651.
[9] Y. Liu, K. Guo, X. Huang, Z. Zhou, and Y. Zhang. Detecting Android Malwares with High-Efficient Hybrid Analyzing Methods. Mobile Information Systems, 2018, pp. 1–12, doi: 10.1155 / 2018/1649703.
[10] Y. Fan, S. Hou, Y. Zhang, Y. Ye, and M. Abdulhayoglu. Gotcha - Sly Malware !: Scorpion A Metagraph2vec Based Malware Detection System. Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, 2018, pp. 253-262.
[11] S. Sen, E. Aydogan, and AI Aysan. Coevolution of Mobile Malware and Anti-Malware. IEEE Trans.Inform.Forensic Secur., 2018, vol. 13, no. 10, pp. 2563–2574, doi: 10.1109 / TIFS.2018.2824250.
[12] F. Shen, JD Vecchio, A. Mohaisen, SY Ko, and L. Ziarek, Android Malware Detection Using Complex-Flows. IEEE Trans. on Mobile Comput., 2019, vol. 18, no. 6, pp. 1231–1245, doi: 10.1109 / TMC.2018.2861405.
[13] S. Millar, N. McLaughlin, J. Martinez del Rincon, P. Miller, Z. Zhao. DANdroid: A multi-view discriminative adversarial network for obfuscated Android malware detection. Proceedings of the tenth ACM conference on data and application security and privacy, 2020, pp. 353-364.
[14] Ahmed, I., Darda, M., & Nath, S. (2021). Blockchain: A New Safeguard to Cybersecurity. Blockchain Technology: Applications and Challenges, 271-284.
[15] PHAM, Hoai Luan, et al. A High-Efficiency FPGA-Based Multimode SHA-2 Accelerator. IEEE Access, 2022, 10: 11830-11845.
[16] NANNIPIERI, Pietro, et al. SHA2 and SHA-3 accelerator design in a 7 nm technology within the European Processor Initiative. Microprocessors and Microsystems, 2021, 87: 103444.
[17] Nair, PR, & Dorai, DR (2021, February). Evaluation of performance and security of proof of work and proof of stake using blockchain. In 2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV) (pp. 279-283). IEEE.
[18] Hewa, T., Ylianttila, M., & Liyanage, M. (2021). Survey on blockchain based smart contracts: Applications, opportunities and challenges. Journal of Network and Computer Applications, 177, 102857.
[19] O'Reilly. Understanding different voting schemes.
[20] URL: https://www.oreilly.com/library/view/machine-learning-for/9781783980284/47c32d8b-7b01-4696-8043-3f8472e3a447.xhtml.
[21] S. Lysenko, O. Pomorova, O. Savenko, A. Kryshchuk and K. Bobrovnikova. DNS-based Anti-evasion Technique for Botnets Detection. Proceedings of the 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Warsaw (Poland), September 24–26, 2015, pp. 453–458.
[22] B. Savenko, S. Lysenko, K. Bobrovnikova, O. Savenko, & G. Markowsky. Detection DNS Tunneling Botnets. Proceedings of the 2021 IEEE 11th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Cracow, Poland, September 22-25, 2021, Vol. 1, pp. 64-69. IEEE.
[23] S. Lysenko, K. Bobrovnikova, R. Shchuka, & O. Savenko. A cyberattacks detection technique based on evolutionary algorithms. In 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies (DESSERT), 2020, pp. 127-132. IEEE.
[24] O. Pomorova, O. Savenko, S. Lysenko, A. Kryshchuk A. Nicheporuk. A Technique for detection of bots which are using polymorphic code. Communications in Computer and Information Science. – 2014. – Vol. 431. – PP.265-276, ISSN: 1865-0929
[25] Gao, Y., Hasegawa, H., Yamaguchi, Y., & Shimada, H. (2021, January). Malware Detection Using Gradient Boosting Decision Trees with Customized Log Loss Function. In 2021 International Conference on Information Networking (ICOIN) (pp. 273-278). IEEE.
[26] MalwareBazaar | Malware sample exchange. URL: https://bazaar.abuse.ch/
[27] Microsoft Store. URL: https://apps.microsoft.com/store/apps.