3. Project “Linklobal” [Electronic resource]. Access mode: https://linkglobal.io
4. Project of the company “AST SpaceMobile” [Electronic resource]. Access mode: https://ast-science.com

Yu. Kravchenko, N. Kozakova, T. Pryshchepa

MODERN SCIENTIFIC PROBLEMS OF CYBER SECURITY

The results of war in cyberspace are decisive for the general development of combat operations in modern confrontations. A cyber attack and cyber intrusion can cause massive damage or significant disruption to critical information infrastructure at any level. This applies primarily to cyberattacks on energy, transport, and military infrastructure facilities, during which facilities for supply management, logistics management, etc. are disabled.

We have been observing such attacks throughout the entire period of confrontation with the Russian Federation, starting in 2014. Everyone remembers how at the end of 2015 there were cyber attacks on infrastructure facilities of the energy sector in the Carpathian region and in the Lviv region, as a result of which entire regions remained without electricity and it took some time to restore the network. On the night of the large-scale military invasion of the Russian Federation on the territory of Ukraine on February 24, 2022, a massive attack was carried out on Ukrainian networks, but thanks to cyber protection measures taken in advance, its consequences were minimized. Recently, there have been prolonged attacks on banking institutions, which did cause damage, but due to the rapid response of cyber defense systems, these losses were not as significant as they could have been. Attacks were also observed on the websites of state institutions and central authorities, as a result of which attackers placed provocative advertisements on them or disabled them with the help of so-called DDoS attacks.

The main means of a cyberattack are related to the use of malicious code and attempts to intrude using system vulnerabilities. Malicious code most often enters the system due to violations of cyber hygiene by users – switching to dangerous sites, opening attachments in suspicious e-mails, and therefore the degree of success of the intrusion is determined by the quality of the protection system.

In cyber security, five key activities can be distinguished for resisting an intrusion:

1) attack detection, i.e. recording the fact of abnormal network operation;

2) identification of the attack, which consists in the precise classification of its type;

3) defense against an attack, covering a sequence of steps aimed at countering an intrusion;

4) response in case of identification of the attacker;

5) elimination of the consequences of the attack and recovery of information using data recovery and preservation algorithms.

Each of these activities is a separate direction in engineering and applied sciences. In addition, in the process of preparing for possible attacks, two key questions should be answered:

1) how resistant is the system to intrusions and capable of countering cyber attacks?

2) what is the quality of the system in relation to the presence of vulnerabilities in software or hardware?

If the first question concerns the effectiveness of the intrusion detection system in the process of interaction with attacks, the second concerns the quality of testing and verification of the system at the stage of its creation.

Modern cyber security technologies actively use both engineering methods and the latest advances in science, such as artificial intelligence methods and formal algebraic methods. In particular, an algebraic approach is being actively implemented, which opens up opportunities for the most accurate analysis of software and hardware system models in order to determine resistance to intrusions and the absence of vulnerable behavior. Another approach is the use of deep learning neural networks built into the Intrusion Detection System (IDS).

Modern intrusion detection systems.

An intrusion can be defined as any type of unauthorized activity that causes damage to an information system. Modern intrusion detection systems provide a significant improvement in protection features compared to previous cyber security tools such as network firewall, virtual private network and notification encryption. Such systems perform two main functions. First, the system detects unwanted behavior as an anomaly, even though it may not be a true intrusion (false detection). Secondly, the system collects data, analyzes actions in network protocols and

compares them with so-called signatures containing data on possible attacks. According to these two functions, two main types of detection systems are distinguished, although in fact there are many varieties that combine these two functions.

Signatures of known attacks exist in the database of the detection system in various specifications, for example, in the form of rules over protocol parameters in the form of if-then-else. If a rule contained in the signature database that qualifies as a certain type of intrusion is met for protocol parameters, then an alarm is triggered.

There are certain difficulties in detecting intrusions that appear over a period of time and contain signs of intrusion in different packets of protocol traffic. To overcome these complications, some systems use the representation of signatures of finite automata, in particular as implemented in the developments of the University of Michigan. At National Taiwan University, the systems use language string templates or semantic conditions. However, invasions stretched over time are not always detected.

It is believed that the comparison with signatures for such intrusion detection systems is a fairly effective method that gives good results in detecting already known attacks, but in the case of zero-day attacks, that is, previously unknown, they are powerless.

Systems based on anomaly detection can detect previously unknown intrusions such as deviations from the normal behavior of network activity. Several varieties are distinguished among these systems. Statistics-based detection systems model the distribution of events for normal behavior, then detect low-probability events and flag them as potential intrusions.

Knowledge-based systems use facts about the normal operation of a network protocol and classify any deviation as an intrusion. The disadvantage of this method is that it is very difficult to gather all the facts about the normal operation of the system, even with the use of formalizing the operation of the protocol with the help of formal structures. More modern systems based on anomaly detection often use machine learning. They demonstrate better accuracy on both known attacks and zero-day intrusions. In addition, such systems, if trained on the right data, can classify known attacks, although other problems arise. Machine learning is the process of extracting knowledge from large amounts of data in order to recognize or predict behavior. Knowledge is formed in the form of a classification model provided by a certain generation algorithm. Clustering algorithms, generation of neural networks, genetic

algorithms, decision trees and the k-nearest neighbors method are used to build models of network behavior classification.

Today, neural networks are the main model in intrusion detection systems. Detection of vulnerabilities in software and hardware systems.

Software vulnerabilities are the main target of attacks that can damage the operation and reputation of millions of systems worldwide, as well as lead to huge financial losses. Therefore, identifying vulnerabilities in both software and hardware is one of the main tasks of cyber security.

Vulnerability detection tools have long been used in software development systems, as well as as separate detection systems. Vulnerability detection is considered both at the level of source code in a high-level programming language and at the level of binary code. Software development systems offer detection of vulnerabilities based on erroneous code fragments, which makes it possible to build a so-called exploit. An exploit is a behavioral script and associated data that an attacker can use to perform an intrusion to destroy a system, compromise its identity, or seize control of a system.

The main disadvantages of using program code fragments are that this method does not guarantee the absence of other vulnerabilities, that is, a vulnerability may exist, for example, in the libraries used by the program, and analysis at the source code level will not detect it. In particular, this concerns the incorrect use of libraries. In addition, the detection of vulnerable code fragments can be false, meaning that the vulnerability found will never work when the programs are executed.

Another, more advanced means of representing vulnerabilities is their formal templates. At the same time, code modeling methods are used, although it is believed that these methods provide rather low coverage.

Both approaches use vulnerability detection systems in binary code. In this case, the problem of adequately presenting vulnerabilities at the level of binary code, which comes from the programming language in which the program is written, arises.

If vulnerabilities are identified, follow-up actions are important. If at the level of programming in high-level languages it is suggested to replace erroneous fragments with safer ones, then at the level of binary code the application of patching technology is considered. On the one hand, automatic correction can lead to unpredictable behavior of the program, but on the other hand, it can be correct if the correction is

equivalent, that is, it does not change the behavior of the program, which must be verified by formal methods.

Neural networks and intrusion detection.

Algebraic methods are used to test the vulnerability finding properties of a system that can be invaded. At the same time, the complexity of the calculations can be quite high and the verification will take a lot of time. However, when detecting attacks during system operation, time is critical, and the algebraic approach may prove ineffective.

To quickly detect an attack in a network environment, deep neuron networks are used, which are able to classify the behavior of the network protocol during an intrusion as abnormal. More advanced neural systems determine the type of attack according to its classification model. As mentioned above, neural networks are used in intrusion detection systems (IDS). Neural systems are built using machine learning or training on certain data sets that are collected in the process of observing the behavior of network protocols. The easiest and fastest way to detect attacks in real time is to identify anomalous behavior that does not correspond to normal protocol actions. However, false detections are possible, since deviations from normal behavior can occur not only due to intrusion into the system, but also due to improper use of resources, user errors, or programs operating in the environment. Therefore, preference is given to systems that are able to classify the causes of the anomaly. However, such systems will not be able to detect an attack for which no training set has been created.

Therefore, considerable attention is paid to data sets intended for training. Today, there are several open data sets containing the behavior of network protocols, but there are certain difficulties with their use. One of the main problems is that the amount of data is insufficient to accurately classify certain attacks. Data may be redundant or noisy. At the same time, there may be an imbalance in the distribution of model classes. Since behavior classification time is critical, deep learning neural networks with minimal number of layers were considered to reduce computation. Other types of networks were also used to improve efficiency: convolutional and recurrent neural networks.

Conclusions.

Therefore, the use of artificial intelligence methods such as machine learning and algebraic deductive methods are more effective in solving cybersecurity

problems than engineering solutions based on viral and specific behavioral signatures and quarantine "sandboxes".

REFERENCES

1. Khraisat A., Gonda I., Vamplew P., Kamruzzaman J. Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecur. 2019. 2: 20.
2. Lin C., Lin Y.-D., Lai Y.-C. A hybrid algorithm of backward hashing and automaton tracking for virus scanning. IEEE Trans. Comput. 2011. 60(4): 594–601.
3. Walkinshaw N., Taylor R., Derrick J. Inferring extended finite state machine models from software executions. Empiri-cal Software Engineering. 2016. 21(3): 811–853.
4. Shen Z., Chen S. A Survey of Automatic Software Vulnerability Detection, Program Repair, and Defect Prediction Techniques. Security and Communication Networks. 2020. 2020: 8858010.
5. Godefroid P., Levin M.Y., Molnar D. SAGE: Whitebox Fuzzing for Security Testing. Queue. 2012. 10(1): 20–27.
6. Kapitonova J., Letichevsky A. Algebraic programming in the APS system. In: ISSAC 90: Proc. Int. Symp. on Symbolic and Algebraic Computation. ACM, New York, 1990. P. 68–75.
7. Letichevsky A. Algebra of behavior transformations and its applications. In: Kudryavtsev V.B., Rosenberg I.G. (eds). Structural Theory of Automata, Semigroups, and Universal Algebra. NATO Science Series II. Mathematics, Physics and Chemistry. Vol. 207. Springer, 2005. P. 241–272.
8. Pulapaka H. Windows sandbox. Windows OS Platform Blog.
9. Letychevskyi O.O., Peschanenko V.S., Hryniuk Y.V. Fuzz Testing Technique and its Use in Cybersecurity Tasks. Cybernetics and Systems Analysis. 2022. 58(1): 157–163.

M. Lenskyi, H. Mykhalchuk, Yu. Honcharova

AN EXACT GPU-ACCELERATED ALGORITHM FOR THE SUBSET SUM PROBLEM

The Subset Sum Problem is a well-known NP-complete problem that asks whether there exists a subset of a given set of integers that sums up to a given target value. This problem has many applications in cryptography, combinatorics, and optimization. However, finding an exact solution for large instances of the problem is computationally challenging, as the number of possible subsets grows exponentially with the size of the input set.

An exact algorithm for solving the Subset Sum Problem with acceleration on GPU is proposed in this work. The algorithm is based on backtracking [1, p. 231], a general technique that explores the search space of possible solutions by recursively branching on each element of the input set. Pruning is utilized to restrict the exploration.