ECONOMIC ANNALS-XXI
ISSN 1728-6239 (Online)
ISSN 1728-6220 (Print)
https://doi.org/10.21003/ea
http://www.soskin.info/ea/

Volume 189 Issue (5-1(1))'2021

Citation information: Dziatkovskii, A., & Hryneuski, U. (2021). The possibilities of ensuring the security of the software product in the conditions of unauthorized access. *Economic Annals-XXI, 189*(5-6(1)), 90-100. doi: https://doi.org/10.21003/ea.V189-09

UDC: 004.056.5

Anton Dziatkovskii
Co-Founder,
Platinum Software Development Company
67-170 Punane Str., Lasnamae Distr., Tallin, 13619, Estonia
founder@platinum.fund
ORCID ID: https://orcid.org/0000-0001-7408-3054

Uladzimir Hryneuski
Chief Content Marketing,
Platinum Software Development Company
67-170 Punane Str., Lasnamae Distr., Tallin, 13619, Estonia
vsvoboden@gmail.com
ORCID ID: https://orcid.org/0000-0002-8506-5131

The possibilities of ensuring the security of the software product in the conditions of unauthorized access

Abstract
Ensuring the security of a software product in the conditions of large companies, taking into account confidential financial and corporate data, is quite an urgent topic in 2021-2023. Over the past year, the number of leaks of confidential information reached a historic peak, together with cyber attacks, and amounts to 114 identified cases. In modern conditions, software security testing is aimed at identifying security errors and design flaws at all stages of the software development lifecycle. At the same time, at the design stage, this type of work should be provided in order to facilitate the implementation of these characteristics in the final version of the security-related system.
Research has shown that there is a wide range of opportunities for developing and using security testing software. These options may differ in implementation technologies, cost and other tactical and technical indicators, characteristics of individual elements, and so on. The main task of developing a software security testing method is to develop, improve and select models, methods and tools that belong to a subset and provide maximum software security indicators.
Our approach allows us to prevent any penetration into the information system, while maintaining 100% security of confidential files and the system as a whole. The threat prevention model works with the help of proactive technology, and if you calculate the economic effect of these measures, it can be different, depending on the value of the enterprise's information itself, and can also be calculated in millions of US dollars. The reliability of the results of mathematical modelling of technologies for creating and implementing «penetration testing» tools is evaluated. The experimental results showed that for all the studied data types, the confidence probability that the value of the statistical value «does not deviate» from the mathematical expectation by more than 1 is 0.94.
Keywords: Security; Software Product; Unauthorized Access; Economic Security; Information; GERT; Cyberattack
JEL Classification: D85; E17; M15
Acknowledgements and Funding: The authors received no direct funding for this research.
Contribution: The author contributed equally to this work.
DOI: https://doi.org/10.21003/ea.V189-09

Дзятковський А.
співзасновник,
компанія з розробки програмного забезпечення Platinum, Таллінн, Естонія
Гриневський В.
керівник, відділ контент-маркетингу,
компанія з розробки програмного забезпечення Platinum, Таллінн, Естонія
Можливості забезпечення безпеки програмного продукту в умовах несанкціонованого доступу
Анотація
Забезпечення безпеки програмного продукту в умовах великих компаній з урахуванням конфіденційних фінансових і корпоративних даних в 2021–2023 роках є актуальною темою. За минулий

рік кількість витоків конфіденційної інформації досягла історичного піку спільно з кібератаками й становить 114 виявлених випадків. У сучасних умовах тестування безпеки програмного забезпечення спрямовано на виявлення помилок безпеки та конструктивних недоліків на всіх етапах життєвого циклу розробки програмного забезпечення. У той же час на етапі проектування цей вид робіт повинен бути передбачений для того, щоб полегшити реалізацію потрібних характеристик в остаточній версії системи, пов'язаної з безпекою.
Дослідження показали, що існує широкий спектр можливостей для розробки та використання програмного забезпечення для тестування безпеки. Ці варіанти можуть відрізнятися технологіями реалізації, вартістю та іншими тактико-технічними показниками, характеристиками окремих елементів і так далі. Основним завданням розробки методу тестування безпеки програмного забезпечення є розробка, вдосконалення та вибір моделей, методів й інструментів, які забезпечують максимальні показники безпеки програмного забезпечення.
Наш підхід дозволяє нам запобігати будь-якому проникненню в інформаційну систему, зберігаючи при цьому 100% безпеку конфіденційних файлів і системи в цілому. Модель запобігання загрозам працює за допомогою проактивної технології, і якщо розрахувати економічний ефект від цих заходів, він може бути різним, залежно від цінності самої інформації підприємства, а також може бути розрахований в мільйонах доларів США. Нами оцінюється достовірність результатів математичного моделювання технологій створення та реалізації інструментів «тестування на проникнення». Результати експериментів показали, що для всіх вивчених типів даних довірча ймовірність того, що значення статистичної величини «не відхиляється» від математичного очікування більш ніж на 1, становить 0,94.
Ключові слова: безпека; програмний продукт; несанкціонований доступ; економічна безпека; інформація.

Дзятковский А.
соучредитель,
компания по разработке программного обеспечения Platinum, Таллинн, Эстония
Гриневский В.
руководитель, отдел контент-маркетинга,
компания по разработке программного обеспечения Platinum, Таллинн, Эстония
Возможности обеспечения безопасности программного продукта в условиях несанкционированного доступа
Аннотация
Обеспечение безопасности программного продукта в условиях больших компаний с учетом конфиденциальных финансовых и корпоративных данных в 2021–2023 годах является актуальной темой. За прошлый год количество утечек конфиденциальной информации достигло исторического пика совместно с кибератаками и составляет 114 выявленных случаев. В современных условиях тестирование безопасности программного обеспечения направлено на выявление ошибок безопасности и конструктивных недостатков на всех этапах жизненного цикла разработки программного обеспечения. В то же время на этапе проектирования этот вид работ должен быть предусмотрен для того, чтобы облегчить реализацию нужных характеристик в окончательной версии системы, связанной с безопасностью.
Исследования показали, что существует широкий спектр возможностей для разработки и использования программного обеспечения для тестирования безопасности. Эти варианты могут отличаться технологиями реализации, стоимостью и другими тактико-техническими показателями, характеристиками отдельных элементов и так далее. Основной задачей разработки метода тестирования безопасности программного обеспечения является разработка, совершенствование и выбор моделей, методов и инструментов, которые обеспечивают максимальные показатели безопасности программного обеспечения.
Наш подход позволяет нам предотвращать любое проникновение в информационную систему, сохраняя при этом 100% безопасность конфиденциальных файлов и системы в целом. Модель предотвращения угроз работает с помощью проактивной технологии, и если рассчитать экономический эффект от этих мер, он может быть разным, в зависимости от ценности самой информации предприятия, а также может быть рассчитан в миллионах долларов США. В статье оценивается достоверность результатов математического моделирования технологий создания и реализации инструментов «тестирования на проникновение». Результаты экспериментов показали, что для всех изученных типов данных доверительная вероятность того, что значение статистической величины «не отклоняется» от математического ожидания более чем на 1, составляет 0,94.
Ключевые слова: безопасность; программный продукт; несанкционированный доступ; экономическая безопасность; информация.

1. Introduction
At the coding and testing stages, it is necessary to conduct an in-depth analysis of the code base and identify shortcomings and errors made during code design and development. At the

same time, you need to combine methods of automated source code analysis and methods of manual code verification.

It should be noted that all the listed stages for obtaining a finished software product allow us to evaluate its quality indicators only in a probabilistic way at the macro level of considering the structure of the software package. Therefore, there is an urgent need to organize a special stage in the creation of software, which is necessary to confirm that the quality indicators of a real software product meet the requirements set for it. As part of the creation of modern information technologies, solving software testing problems and obtaining documentary confirmation of the necessary program quality indicators is combined within the certification process.

In general, software security testing can be divided into two main components: software vulnerability detection; penetration testing; and operation testing. At the same time, among the main methods of software security testing, we highlight the following:
1) analysis of CCD architecture and design;
2) building a threat model;
3) search for vulnerabilities in the source code;
4) penetration test;
5) risk-based testing;
6) fuzzi-testing;
7) check the deployment process for compliance with the requirements.

2. Brief Literature Review

Analysis of the literature (Harrison et al., 1976) has shown that today there are many approaches to mathematical modelling of the software testing process in general and software security testing in particular. They are based on the principles of Information Security (IEEE, 2018), queuing (Kanner et al., 2012), neural networks (Matveychikov, 2014), fuzzy logic (Rakitskiy, 2011), chaotic dynamics (Sandhu, 2000), graph models and combinatorial calculation methods (Shcheglov, 2014), etc. in addition, tools for solving optimization problems formalized on the basis of these models have been developed. These are, first of all, analytical methods, mathematical programming methods, heuristic methods, etc. (Dileep, 2020). Let's analyze the most commonly used software security testing models in practice, which are adapted to varying degrees to meet the requirements for formalizing software development management processes (Maglaras et al., 2018).

Another important method of testing software security is the penetration test. In a general sense, «penetration testing» is a product or service for an authorized attempt to circumvent information system security tools. The test result is a report that can / should contain a list of detected vulnerabilities, attack vectors used, results achieved, and recommendations for fixing. This approach is described in OSSTM (Open Source Security Testing Methodology) (Ferrag et al., 2018). It includes: testing the perimeter, the human factor, all types of communications and interactions, operations on data in any form, etc.

Another modern approach to mathematical modelling is modelling using neural networks (Al-Shaer, 2016). To a large extent, this is due to the specifics of the functioning of computer systems, which are human-machine systems. Many scientists are engaged in neuroinfomatics and research of neural networks in various fields (Kimani et al., 2019). With the help of artificial neural networks, you can process, analyze and generalize information similar to the work of the human brain. Neutron networks are used in many industries (economics, medicine, communications, security (including computer antivirus security, information input and processing). But, as with the biological approach, neural network models have some limitations in their application. These models are most often used in decision-making (support) systems, pattern recognition, optimization, and so on.

3. Methods and Tasks of the Research

Research has shown that there are a wide range of options for developing and using security testing software. These options may differ in implementation technologies, cost and other tactical and technical indicators, characteristics of individual elements, and so on.

The main task of developing a software security testing method is to develop, improve, and select models, methods, and tools that belong to a subset and provide maximum software security indicators.

When developing a set of mathematical models of technologies for generating and implementing «penetration test» tools, it is necessary to identify the following specific scientific tasks:
- development and research of GERT model for the initial generation of cyberattack code for unauthorized access to peer-to-peer computer system resources and GERT model for the processes of active analysis of the resource management system and implementation in the computer system;
- setting and solving a mathematical modelling problem;
- assessment of the effectiveness and adequacy of the developed set of mathematical models.

When developing a method for selecting an algorithm from binary code for software security analysis, there is a need to evaluate and choose methodological approaches to the development and construction of organizational structures, develop a general scheme for selecting an algorithm from binary code for software security analysis, develop a model for selecting a set of attractors with common features, synthesize related attractors, develop a model for detecting the presence of dynamic code in part of the attractor, develop an optimization model for selecting an algorithm from binary code, which are components in the software security testing process.

4. Results

4.1. GERT - Model of Initial Generation of Cyberattack Code for Unauthorized Access to Computer System Resources

When simulating a node cyberattack, a large number of tasks arise that can be solved using models of this kind. They can be used both independently of each other and in combined systems (Maglaras, 2014). The user can be provided with several new types of GERT models (homogeneous networks of large dimensions, heterogeneous networks, networks with aging applications, random GERT networks, etc.).

We will analyze and develop a GERT model of the initial stage of the considered criminal influence - the initial generation of the node cyberattack code to the resources of the computer system of the temporary network (Figure 1).

Development of the GERT network at the initial code generation stage.

The significance of each parameter is determined by the pairwise comparison method. Determining significance coefficients involves:
- determining the significance level of a parameter by assigning different ranks;
- checking the suitability of expert assessments for further use;
- determining the evaluation of the PairWise priority of parameters;
- processing the results and determining the significance coefficient.

The results of the expert ranking are shown in Table 1.

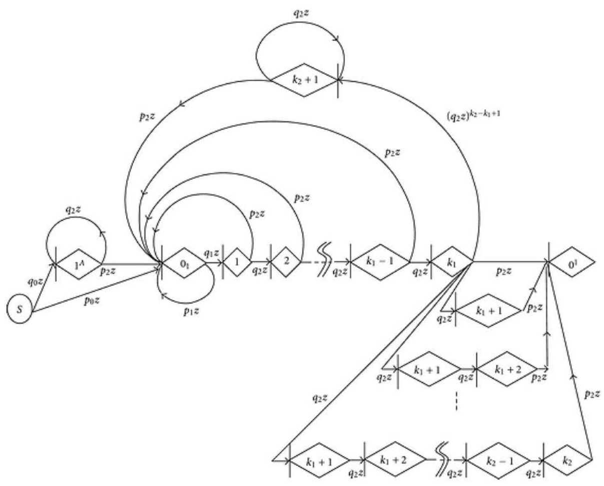

Figure 1:
GERT - network of the node cyberattack code generation algorithm
Source: Authors' own research

Table 1:
Expert ranking of results

Parameter designation	Parameter name	Units of measurement	Rank of the parameter according to the expert's assessment							Sum of ranks R_i	Rejection Δ_i	Δ_i^2
			1	2	3	4	5	6	7			
k1	Performance of the programming language	Op / Ms	2	2	1	2	2	1	2	12	-2	4
k2	Amount of memory to save data	MB	1	1	2	1	1	2	1	9	-5	25
k3	Potential amount of program code	Number of code terms	3	3	3	3	3	3	3	21	7	49
	Together		6	6	6	6	6	6	6	42	0	78

Source: Authors' own research

To check the degree of reliability of expert assessments, we will define the following parameters:
a) the sum of the ranks of each of the parameters and the total sum of the ranks:

$$R_i = \sum_{j=1}^{N} r_{ij} R_{ij} = \frac{Nn(n+1)}{2} = 42 \ , \tag{1}$$

where:
N - number of experts;
n - number of parameters;
R_{ij} - rank;

b) average sum of ranks:

$$T = \frac{1}{n} R_{ij} = 14 \ ; \tag{2}$$

c) deviation of the sum of ranks of each parameter from the average sum of ranks:

$$\Delta_i = R_i - T \ , \tag{3}$$

the sum of deviations for all parameters must be 0;

d) total sum of squares of deviation:

$$S = \sum_{i=1}^{N} \Delta_i^2 = 78 \ , \tag{4}$$

where:
Δ_i^2 - rejection;
N - number of experts.

Let us calculate the consistency coefficient:

$$W = \frac{12S}{N^2(n^3-n)} = \frac{12 \cdot 78}{7^2(3^3-3)} = 0.79 > W_k = 0.67 \ . \tag{5}$$

The ranking can be considered reliable, since the found consistency coefficient exceeds the standard one, which is equal to 0.67.

Using the ranking results, we will perform a pairwise comparison of all parameters and enter the results in Table 2.

Table 2:
Parameters of experts' evaluation

Parameters	Experts							Final evaluation	Numeric value
	1	2	3	4	5	6	7		
k1 & k2	>	>	<	>	>	<	>	>	1.5
k1 & k3	<	<	<	<	<	<	<	<	0.5
k2 & k3	<	<	<	<	<	<	<	<	0.5

Source: Authors' own research

Numeric value that determines the degree of preference i parameter over j, a_{ij} is determined by the formula:

$$a_{ij} = \begin{cases} 1{,}5 \; when \; Xi > Xj; \\ 1.0 \; when \; Xi = \; Xj; \\ 0.5 \; when \; Xi < \; Xi. \end{cases} \tag{6}$$

From the obtained numerical preference estimates, we make a matrix $A = ||a_{ij}||$.

Relative estimates are calculated several times until the following values differ slightly from the previous ones (less than 2%).

Thus, a mathematical GERT model of the process of generating the code of an unauthorized access cyberattack has been developed. The proposed mathematical model differs from the known ones by taking into account the main stages of code generation for Windows or Linux operating systems during the mathematical formalization of the GERT network with the ability to search for modern solutions on the Internet. The model can be used to study the main stages of generating the node cyberattack code in order to develop practical recommendations for countering the node process to the resources of a computer system of a temporary network, as well as during the development of new methods, algorithms and methods for managing computer systems.

4.2. Model for Detecting the Presence of Dynamic Code in a Part Attractor

The presence of dynamically changing code in the program can be caused by various reasons. Here are several possibilities, from the most common ones that are not related to deliberate opposition to analysis, to targeted ones that create difficulties, especially for static analysis (Aloul, 2012).

Dynamic editor links loading and unloading dynamic libraries. The address range of the newly loaded library may overlap with the address range that belonged to the already uploaded ones. Thus, at different points in time, the same memory addresses will contain different code.

Use of «jumps» and deferred binding. When you first go to an address, a dynamically linked control function can be passed to a subroutine that performs deferred binding. After the binding is completed, the subroutine will correct the «springboard» so that the function is called directly on subsequent calls, and transfers control to it for the first time.

The presence in the program of mechanisms for decrypting, decompressing, and dynamically generating code that is overwritten or already unnecessary and discarded fragments of the program, or if such mechanisms work with the program in parts, they use one rewritable buffer for the next decrypted, unpacked, or generated part of the program.

Polymorphic nature of the program or part of it. This case differs from the previous one in that often the next version of the program code is built on the basis of the previous one, which further complicates the analysis. Most often, this mechanism is embedded in malicious code, especially viruses, in order to interfere with signature analysis in antivirus software (Ferrag et al., 2020).

The method proposed in this paper does not distinguish between the causes of dynamic code changes. All features are handled in the same way, which, on the one hand, has the advantages of method universality, but, on the other hand, ignores the additional «hints» about application behaviour that are contained in these reasons. At the same time, the evolution graph (Figure 2) gives a certain idea of the nature of dynamic code in this program.

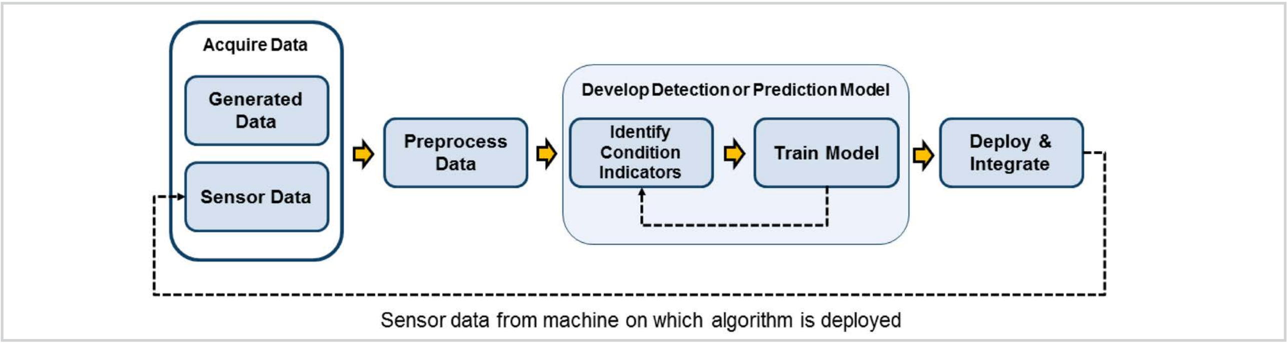
Acquire Data
Generated Data
Sensor Data
Preprocess Data
Develop Detection or Prediction Model
Identify Condition Indicators
Train Model
Deploy & Integrate
Sensor data from machine on which algorithm is deployed

Figure 2:
Model for detecting the presence of the dynamic code
Source: Authors' own research

Regardless of the content included in an episode of dynamic program code changes, it follows one of the following two scenarios.

The program running on this system writes values to memory. This memory can belong to either this program or any other (to do this, the operating system must support the ability to connect the execution thread of one process to the address space of another). The written values either change the previously executed code, or form code that will be executed in the future. One way or another, it is written to addresses that also appear in the attractor as executable.

One or more pages of physical memory are changed as a result of performing a DMA transaction by a computer system device (most often a hard disk controller). At the same time, the attractor does not observe the fact of direct recording, since such transactions are initiated by the device itself, and they do not correspond to any instructions. The recorded addresses, as in the first scenario, were either used earlier or will be used as executables in the future.

In some questionable situations, the constructed algorithm may require additional clarifications. Such situations are possible in programs equipped with certain types of attachment protection, as well as in the operating system code. Therefore, in the future, there is a need for practical adaptation of the developed algorithms to possible casuistic deviations in programs.

Hence, a set of algorithms for isolating a set of attractors with common features and synthesizing information about the system under study is developed, which is the first stage of the method for isolating an algorithm from binary code using additional attractors (Figure 3). A distinctive feature of the development of this stage is the ability to build a graph for an arbitrary attractor, without limiting the static nature of the code. This will make it possible to significantly expand the range of program codes under study, including codes that show signs of dynamic change.

4.3. Development of a Simulation Model of the Generation and Implementation Process «Penetration Test» Tools

One of the key stages of the dissertation research is the synthesis of the obtained results into a single methodological system for the practical application of technologies for testing the security of peer-to-peer computer system software and the development of a simulation model of this system. At the same time, the goal should be to solve the following partial tasks:

- checking the adequacy of the developed models and the method of isolating the algorithm from binary code for software security analysis;
- analysis of the reliability of the results obtained when solving the set optimization tasks;
- reasonable choice of indicators, coefficients and parameters of system functioning;
- development of scientific and practical recommendations on the use of models and a method for isolating an algorithm from binary code for software security analysis.

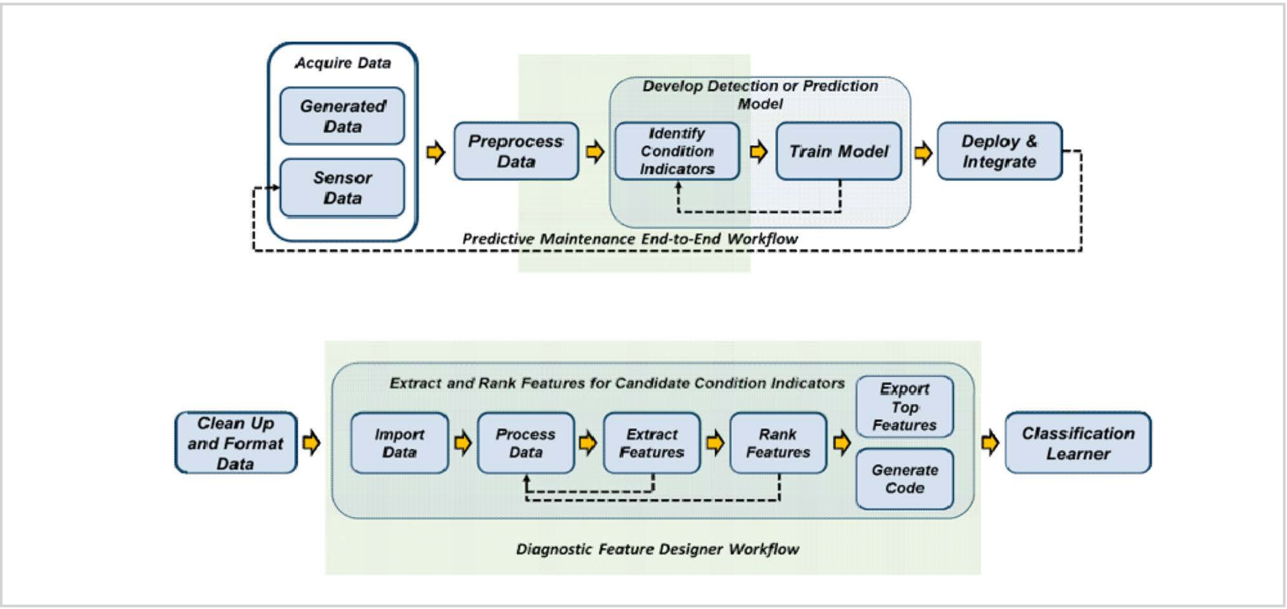

Figure 3:
Ensemble data for the dynamic code analyzing
Source: Authors' own research

To substantiate the reliability of the obtained results of mathematical modelling and evaluate the effectiveness of the method of isolating the algorithm from binary code for software security analysis, simulation modelling was performed. The MathCAD - 14 symbolic mathematics environment and specialized programs that perform the functions of both security tools and tools for generating cyber attacks of various formats (Ixchariot 9, WireShark) are used as simulation modelling tools (Kimani et al., 2019).

When evaluating the effectiveness of software security testing, it is necessary to develop criteria for selecting warnings for analysis. In the dissertation work, an indicator of true warnings was selected to assess the effectiveness of the proposed method and the level of software security. On a large volume of source code, as a rule, a large number of warnings are detected, and viewing all issued warnings and judging for each of them whether it is true or false is an excessively time-consuming task (up to several thousand warnings of the same type can be issued on a complete set of test software packages). The paper uses a pseudo-random sample of a fixed number of warnings of each type (or a group of similar types) that is stable between different analysis runs (Figure 4).

The use of the proposed method for the studied practical examples also made it possible to increase the level of software security by up to 3%.

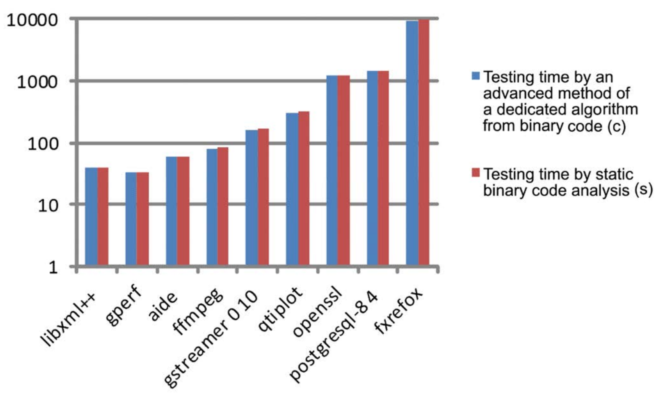

Figure 4:
Comparative histogram of software security testing time
Source: Authors' own research

4.5. Economic Consequences of Information Threats in Companies

The purpose of ensuring the economic security of the enterprise should be a system for countering potential and real threats, the development of preventive measures to eliminate or minimize which should ensure the business entity's successful functioning in unstable conditions of the external and internal environment. At the same time, the security of the enterprise should be ensured in such main areas as economic, scientific and technical, information, personnel, social, environmental, physical security, etc (Dewa et al., 2016).

The breakthrough of Information Technologies at the end of the XX - beginning of the XXI century caused significant systemic transformations in the world, which made it possible to form and develop fundamentally new and integral global substances - the information space and the information society. The uncontrolled spread and unlimited use by the world's leading countries of the information space as an arena of action in the process of modern information warfare has gradually led to the vulnerability of the information sphere of these countries to the impact of internal and external cyber interventions and threats of a deliberate, accidental, natural or artificial nature (Sutherland, 2020).

At the same time, the dependence of the overall level of economic security of the state and enterprises on its information component is becoming increasingly obvious.

the global sphere is incidents in cyberspace that include cybercrime or unauthorized interference in databases, as well as technical failures in information and communication systems.

So, according to the study, the share of cyber incidents increased by 11% compared to 2019, for the first time this risk moved from the fifth position to the third. Two years ago, in the first study, only 5% of respondents considered cyber incidents as a risk (Table 3).

Unfortunately, the management of domestic enterprises does not pay enough attention to the protection of information and communication systems (Table 4). The main reasons are the lack of awareness of the possible consequences of cyber attacks and significant investments in creating a system for protecting information resources.

Since 2011, Kaspersky Lab, together with the international analytical company B2B International, has been conducting annual global surveys of Information Technology Specialists of small, medium and large companies around the world.

According to the results of the survey, the following trends became the main ones in the field of information security threats and countering them:

- large companies spend an additional 2.1 million on Incident Response and Prevention. rubles, and small ones - about 300 thousand rubles;
- most often, as a result of cybersecurity incidents, companies lose operational data about internal activities, personal data of customers and financial statements.

Also, significant changes occurred in the top three priority tasks in the field of Information Technology compared to last year: the main problem was the protection of confidential data (customer data, financial information, etc.) from targeted attacks that were not even included in this list last year. The second place (34%) was taken by the more general issue of data protection, which had previously been in the lead for three years. And the third place went to a task that was also not previously included in the list of priorities - 29% of respondents noted the need to ensure the smooth operation of critical systems (for example, through the use of DDoS protection tool).

Our approach allows us to prevent any penetration into the information system, while maintaining 100% security of confidential files and the system as a whole. The threat prevention model works with the help of proactive technology and if you calculate the economic effect of these measures, it can be different, depending on the value of the enterprise's information itself, and can also be calculated in millions of US dollars.

5. Conclusion

Analysis of the main security requirements for software tools of computer systems has shown that in the context of increasing demand for software products, as well as an increase in the

Table 3:
The most significant risks for European enterprises

№	Risks	Rating of 2019	Place in rating of 2018
1	Break in production and delivery	53%	2
2	Market changes (volatility, increased competition, stagnation)	52%	1
3	Cyber incidents (cybercrime, data leaks, IT failures)	40%	6
4	Changes in legislation and regulation (economic sanctions, protectionism)	39%	5
5	Macroeconomic changes (austerity, rising prices, inflation)	31%	New
6	Natural disasters (thunderstorms, floods, earthquakes)	31%	3
7	Loss of reputation or brand value	29%	7
8	Fire, explosion	22%	4
9	New technologies and innovations	19%	New
10	Political risks (war, terrorism, riots)	17%	8

Source: https://osha.europa.eu/en/publications/european-survey-enterprises-new-and-emerging-risks-managing-safety-and-health-work

Table 4:
Most significant risks for CIS enterprises

№	Risks	Rating 2019	Place in rating of 2018
1	Political risks (war, terrorism, riots)	67%	1
2	Embezzlement, fraud and corruption	41%	2
3	Terrorism	33%	3
4	Fire, explosion	28%	4
5	Break in production and delivery	24%	5
6	Natural disasters	16%	6
7	Macroeconomic changes (austerity, rising prices, inflation)	19%	7
8	Single social contribution of the market	12%	9
9	Cyber incidents (cyber attacks, data leaks, IT failures)	13%	8
10	Changes in legislation and regulation (protectionism)	15%	10

Source: CIS Audit/Assurance Program Helps Enterprises Navigate Risk, 2019 intensity of malicious actions, software security testing methods do not allow to provide the necessary level of Information Protection. The study of the main approaches of mathematical modelling of software security testing technologies allowed us to make a reasoned choice and formulate the scientific task of dissertation research.

A mathematical model of the initial generation of the code of a cyberattack of unauthorized access to Information Resources has been developed, which takes into account in the process of mathematical formalization of the GERT network, the main stages of code generation for Windows or Linux operating systems with the ability to search for solutions on the Internet, which made it possible to find the maximum values of the probability distribution density of the time of initial generation of the code of a cyberattack of unauthorized access to information resources.

The mathematical model of the processes of active analysis of the resource management system is improved and implemented in the security system, which differs from the known factors of completeness of test techniques and finality of the specification when implementing the algorithm in the security system. This made it possible to increase the effectiveness of software security testing.

Improved the method of separating the algorithm from binary code for software security testing, which differs from the known ones by the ability to take into account iterative scenarios in reverse engineering technologies, when new paths are added to the consideration and there are no restrictions on the nature of the analyzed code, which made it possible to increase the level of software security.

The use of GERT networks in mathematical modelling makes it possible to use the results obtained in analytical form (distribution functions and densities) for comparative analysis and research of various stages and stages of testing computer systems, which increases the efficiency of software security testing by up to 5%.

The synthesis of the method for selecting an algorithm from binary code for software security analysis made it possible to implement the appropriate software tools and carry out the proposed procedure for selecting an algorithm, including iteratively, with the replenishment of the set of paths considered during the analysis, as well as to increase the level of software security up to 3%.

The reliability of the results of mathematical modelling of technologies for generating and implementing «penetration test» tools is evaluated. The results of the experiments showed that for all the studied types of data, the confidence probability that the value of the statistical value «does not deviate» from the mathematical expectation by more than 1 is 0.94.

References

1. Aloul, F. A., Al-Ali, A. R., Al-Dalky, R., Al-Mardini, M., & El-Hajj, W., (2012). Smart grid security: Threats, vulnerabilities and solutions. *International Journal of Smart Grid and Clean Energy, 1*(1), 1-6. https://doi.org/10.12720/sgce.1.1.1-6
2. Al-Shaer, E., & Rahman, M. A. (2016). *Security and resiliency analytics for smart grids.* Advances in Information Security. https://doi.org/10.1007/978-3-319-32871-3
3. Dewa, Z., & Maglaras, L. A. (2016). Data mining and intrusion detection systems. *International Journal of Advanced Computer Science and Applications, 7*(1), 62-71. https://doi.org/10.14569/IJACSA.2016.070109
4. Dileep, G. (2020). A survey on smart grid technologies and applications. *Renewable Energy, 146,* 2589-2625. https://doi.org/10.1016/j.renene.2019.08.092
5. Ferrag, M. A., Maglaras, L. A., Janicke, H, Jiange, J., & Shu, L. (2018). A systematic review of data protection and privacy preservation schemes for smart grid communications. *Sustainable Cities and Society, 38,* 806-835. https://doi.org/10.1016/j.scs.2017.12.041
6. Ferrag, M. A., Maglaras, L., Moschoyiannis, S., & Janickeb, H. (2020). Deep learning for cyber security intrusion detection: Approaches, datasets, and comparative study. *Journal of Information Security and Applications, 50,* 102419. https://doi.org/10.1016/j.jisa.2019.102419
7. Harrison, M. A., Ruzzo, W. L., & Ullman, J. D. (1976). Protection in Operating Systems. *Communications of the ACM, 19*(8), 461-471. https://doi.org/10.1145/360303.360333
8. IEEE. (2018). *IEEE Std 1003.1-2017 (Revision of IEEE Std 1003.1-2008) and The Open Group Technical Standard Base Specifications, Issue 7.* IEEE and The Open Group. http://pubs.opengroup.org/onlinepubs/9699919799
9. Kanner, A. M., & Ukhlinov, L. M. (2012). Access control in GNU/Linux. *Information Security Questions, 3,* 35-38. https://www.okbsapr.ru/library/publications/kanner_2012_4 (in Russ.)
10. Kimani, K., Oduol, V., & Langat, K. (2019). Cyber security challenges for iot-based smart grid networks. *International Journal of Critical Infrastructure Protection, 25,* 36-49. https://doi.org/10.1016/j.ijcip.2019.01.001
11. Li, X., Liang, X., Lu, R., Shen, X., Lin, X., & Zhu, H., (2012). Securing smart grid: cyber attacks, countermeasures, and challenges. *IEEE Communications Magazine, 50*(8), 38-45. https://doi.org/10.1109/MCOM.2012.6257525
12. Maglaras, L. A., & Jiang, J. (2014). A real time OCSVM intrusion detection module with low overhead for SCADA systems. *International Journal of Advanced Research in Artificial Intelligence 3*(10), 45-53. https://doi.org/10.14569/IJARAI.2014.031006

13. Maglaras, L. A., & Jiang, J. (2014). Ocsvm model combined with k-means recursive clustering for intrusion detection in scada systems. *10th International conference on heterogeneous networking for quality, reliability, security and robustness* (pp. 133-134). https://doi.org/10.1109/QSHINE.2014.6928673
14. Maglaras, L. A., Kim, K. H., Janicke, H., Ferrag, M. A., Rallis, S., Fragkou, P., Maglaras, A., & Cruzg, T. J. (2018). Cyber security of critical infrastructures. *ICT Express, 4*(1), 42-45. https://doi.org/10.1016/j.icte.2018.02.001
15. Matveychikov, I. V. (2014). Overview of Dynamic Operating System's Kernel Hooking Methods (Study Case of Linux Kernel). *Bezopasnost Informatsionnykh Tekhnology, 4,* 75-82. https://bit.mephi.ru/index.php/bit/article/view/146 (in Russ.)
16. Rakitskiy, Y. S., & Belim, S. V. (2011). Model of Union Two Mandatory Security Policies. *Bezopasnost Informatsionnykh Tekhnology, 1,* 125-126 (in Russ.).
17. Sandhu, R., Ferraiolo, D. F., & Kuhn, R. (2000). The NIST Model for Role - based Access Control: towards a unified standard. *Proceedings of the fifth ACM Workshop on Role-based Access Control* (pp. 47-63). New York: ACM. https://doi.org/10.1145/344287.344301
18. Shcheglov, K. A., & Shcheglov, A. Yu. (2014). A Consistent Model of Mandatory Access Control. *Journal of Instrument Engineering, 57*(4), 12-15 (in Russ.).
19. Sutherland, B. R. (2020). Securing smart grids with machine learning. *Joule, 4*(3), 521-522. https://doi.org/10.1016/j.joule.2020.02.013

Received 24.03.2021
Received in revised form 9.04.2021
Accepted 22.04.2021
Available online 10.06.2021