Синтезований комплекс алгоритмів обфускації програмних модулів, що відрізняється від відомих урахуванням варіативності типів даних. Це дозволило описати дані процеси на верхньому стратегічному рівні формалізації. Досліджено можливості використання GERT-моделей з метою застосування різних варіантів законів розподілу і їх параметрів при переході між станами. Розроблено уніфіковану GERT-модель процесу обфускації програмних модулів. Дана модель відрізняється від відомих реалізацією парадигми використання математичного апарату Гамма розподілу в якості ключового на всіх етапах моделювання процесу обфускації. Це дозволило досягти уніфікації моделі в умовах модифікації GERT мережі. Розраховані математичне очікування і дисперсії часу виконання випадкової величини часу обфускації і деобфускації програмних модулів. Результати дослідження показали, що для розробленої математичної моделі додавання додаткового процесу обфускації призводить до збільшення дисперсії часу виконання на 12 %, а при видаленні з системи – зменшується на 13 %. Математичне очікування часу виконання змінюється в геометричній прогресії – так, при видаленні вузла відбувається зменшення математичного очікування на 9 %, а при збільшенні на 1 вузол – збільшення математичного очікування на 26 %. Це показує незначність змін досліджуваних показників в умовах модифікації моделі і підтверджує гіпотезу про уніфікацію моделі в умовах використання математичного апарату Гамма розподілу як основного. Дані результати дають розробнику можливість спрогнозувати поведінку системи захисту програмних модулів з точки зору часу виконання. Це дозволяє зменшити час на прийняття рішення про доцільність використання процесу обфускації

Ключові слова: GERT модель, обфускація програмних модулів, програмний код, гамма розподіл, java

UDC 044.422
DOI: 10.15587/1729-4061.2020.206232

DEVELOPMENT OF UNIFIED MATHEMATICAL MODEL OF PROGRAMMING MODULES OBFUSCATION PROCESS BASED ON GRAPHIC EVALUATION AND REVIEW METHOD

S. Semenov
Doctor of Technical Sciences, Professor*
E-mail: s_semenov@ukr.net
V. Davydov
PhD*
E-mail: vyacheslav.v.davydov@gmail.com
O. Lipchanska
PhD*
E-mail: lipchoks@gmail.com
M. Lipchanskyi
PhD, Associate Professor*
E-mail: ctpu@ukr.net
*Department of Computer Engineering and Programming
National Technical University
"Kharkiv Polytechnic Institute"
Kyrpychova str., 2, Kharkiv, Ukraine, 61002

Received date 13.05.2020
Accepted date 17.06.2020
Published date 30.06.2020

1. Introduction

The widespread use of computer systems increases the value and role of software. This determines the role of malicious cyber attacks aimed at discrediting software products and reducing the security of computer systems in general.

In accordance with the requirements of laws and regulations [1, 2], software (computer programs) is one of the objects requiring protection at all stages of development and operation. At the same time, software obfuscation at the development stage can become one of the key processes for providing security, privacy, authentication and integrity services. This is primarily due to the feasibility of these algorithms and procedures of obfuscation at the very early stages of code development and implementation, as well as their advanced capabilities while providing a variety of security services in general.

One of the methods of protecting the code of a software product is the obfuscation process [3], which provides the services of information security and privacy. Thus, obfuscation is an important component of providing practical security services.

The software obfuscation process consists of subprocesses that may or may not be used depending on the business process, total runtime, and level of protection provided. Creating models for each particular case is a costly process that requires unification.

In this regard, the urgent task is to develop an approach based on the unification of the mathematical formalization of the software protection process to assess the probabilistic

characteristics of the runtime of the programming modules obfuscation procedures.

2. Literature review and problem statement

An increasing number of attacks aimed at reducing the security of computer systems in some cases is associated with the emerging gaps between the development of theory and the practice of applying theoretical results, as well as imperfect mathematical models that do not provide increased requirements for practitioners.

The review of the literature [4–12] showed that for a number of special cases, the final results of the study taking into account possible limitations are presented. So, in [4–6], comprehensive results are obtained for the case when a random process determining transitions from one state to another is formalized by the exponential distribution law. This allows simplifying the solution of problems, but in advance introduces an error in the descriptive part of the model formalizing non-Markov systems.

More complex models based on the principles of decomposition of complex algorithms and private-level architectures are presented in [7–12]. However, the problem of obtaining final relations for calculating probability-time characteristics for cases when transitions between states are described by more complex than exponential distribution laws, but lacking signs of "markovianity" is studied insufficiently.

It should be noted that in addition to substantiating the mathematical apparatus for solving the problem, the choice of means of mathematical formalization is important.

Analysis of approaches to network stochastic modeling showed their great diversity (based on Petri nets [13], finite discrete automata [14], PERT networks [15], etc.). However, the main drawback of this modeling approach is the limited practical application due to the lack of predictability properties.

In [16, 17], GERT (Graphical Evaluation and Review Technique) models of complex technical systems and processes are presented. However, the introduction of the assumption of an exponential distribution law as the main law characterizing the process of transition from state to state significantly reduces their theoretical and practical value.

In [17], a situation is proposed where the semi-Markov process is the main iterative process of obfuscation. However, it is shown that the problem of finding the distribution law for semi-Markov models of large dimension is solved with an error of about 15 %, and the final distribution is discrete. It is shown that these models are not suitable for solving problems that require accurate knowledge of the distribution function or density.

The review of the literature [13–17] showed that existing modeling methods have both advantages and disadvantages. It should be borne in mind that the obfuscation process by these models was not described.

Thus, the review showed that a number of mathematical models of complex algorithms and processes of software protection are formalized in terms of graph theory. It is often assumed that the functioning of the system as a whole can be described by one distribution law. In this case, possible options for applying the distribution laws and their parameters during the transition from state to state are not taken into account. The solution of this contradiction is possible by the mathematical formalization of processes using GERT structures. At the same time, finding the probability distribution of transitions from state to state in the process of software protection, as well as the final result in the form of a distribution law with the found parameters are of theoretical and practical interest.

It should be noted that the study of complex GERT networks is difficult due to the high computational requirements of stochastic modeling approaches. At the same time, the problem of developing simplified unified GERT models has not been studied enough. Thus, there is a need to develop a unified model to formalize the programming modules obfuscation process in order to eliminate this drawback.

3. The aim and objectives of the study

The aim of the study is to develop a unified GERT model of programming modules obfuscation. This will make it possible to achieve the unification of the model in conditions of modifying the GERT network.

To achieve the aim, the following objectives were set:

– synthesis of a set of algorithms of the programming modules obfuscation/deobfuscation process;

– development of a GERT model of the programming modules obfuscation process based on algorithms;

– study of a unified GERT model with a modified number of nodes.

4. Synthesis of a set of algorithms of the programming modules obfuscation/deobfuscation process

The decompiled bytecode of existing software products using programming modules obfuscation processes for their protection is investigated. On the basis of the studies, algorithms of programming modules obfuscation are developed (Fig. 1).

In accordance with the presented algorithms, the process of the source code obfuscation can be described by the following steps.

Step 1. Initial state. There is raw source code in a high-level language written using a virtual machine (Java, CLI, etc.). Due to the fact that the compiled code has a number of problems with modification, the process of modification and obfuscation of the code takes place based on the "raw" (source) code. Due to the fact that this work uses a combined approach of two independent obfuscation methods, the first step can be either paragraph 2, or paragraph 4, or paragraph 5.

Step 2. The source code runs through the obfuscation method, based on the modification of string literals. The modified source code will not make sense without the reverse algorithm of Step 3.

Step 3. Due to the fact that the algorithm of string literals conversion is symmetric, the inverse conversion function is added to the source code. The inverse conversion algorithm of paragraph 2 is an integral complement, however, its order of addition does not matter.

Step 4. The source code runs through the obfuscation method, based on the "untangling" of structures containing Boolean operations (performs operations opposite to simplification). This method has no dependencies and can be executed at any step/not executed depending on the combining rule.

Step 5. The source code runs through the obfuscation method based on obfuscation of identifier names. This method has no dependencies and can be executed at any step/not executed depending on the combining rule. This method has a number of options, which allows obfuscating: local variables, global variables, functions, classes.

Step 6. As a result of Steps 2–5, the source code is ready for the compilation process. At the same time, to eliminate side effects, it is necessary to make sure that the compiler does not perform premature code optimization. Also, to reduce the readability of the code and complicate the debugging process, debugging information must be disabled. So, for example, for languages based on the Java virtual machine, compiling with the "-g:none" attribute disables debugging information. And the "-Xint" compiler option disables Just-In-Time and Ahead-Of-Time compilations, leading to code optimization by the compiler.

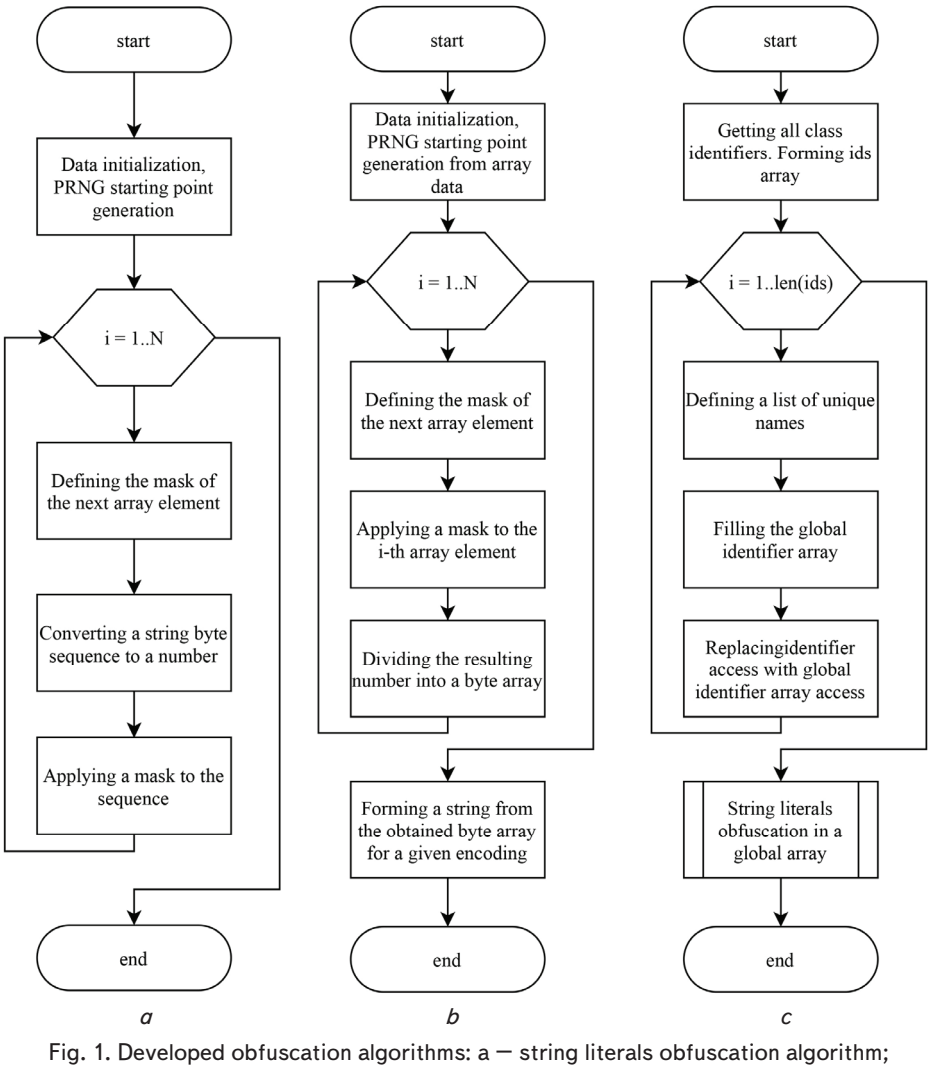

Fig. 1. Developed obfuscation algorithms: a – string literals obfuscation algorithm; *b* – string literals deobfuscation algorithm; *c* – identifier names obfuscation algorithm

In accordance with the above, a general algorithm of programming module obfuscation is developed (Fig. 2), as well as a GERT network of programming modules obfuscation and deobfuscation processes (Fig. 2, 3).

In Fig. 3 and the corresponding Table 1, based on the developed algorithms of Fig. 1, transitions between states are formulated that characterize:

– (1, 2): processing the source code by modifying (encoding) string literals, embedding a function in the source code that decodes the modified string literals into the initial state "on the fly";

– (2, 3): after successfully encoding string literals, we perform the process of Boolean operations obfuscation, verification of the conversion success of Boolean operations;

– (3, 4): after successful obfuscation of Boolean operations, we perform the process of identifier names obfuscation, verification of the conversion success of identifier names;

– (4, 5): after successful obfuscation of identifier names, we go to the final state, ready for compilation;

– (1, 3): depending on the business scenario, we skip the source code processing by encoding string literals and perform the process of Boolean operations obfuscation;

– (1, 4): depending on the business scenario, we skip the source code processing by encoding string literals and obfuscation of Boolean operations. We perform the process of identifier names obfuscation;

– (2, 4): depending on the business scenario, we skip the source code processing by encoding Boolean operations obfuscation. We perform the process of identifier names obfuscation;

– (3, 5): depending on the business scenario, we skip the source code processing by obfuscation of identifier names;

– (2, 5): depending on the business scenario, we skip the source code processing by obfuscation of Boolean operations and identifier names;

– (2, 1): an error occurred in the process of encoding by modifying string literals and adding a decoding function. We return to the initial state in order to repeat the obfuscation attempt;

– (3, 1): an error occurred in the process of verification of the conversion success of Boolean operations. We return to the initial state (since the reason for the error is unknown – the problem of Boolean operations obfuscation or the error is caused by the modification of string literals in the previous steps) in order to repeat the obfuscation attempt;

– (4, 1): an error occurred in the process of verification of the conversion success of identifier names. We return to the initial state (since the reason for the error is unknown – the problem of identifier names obfuscation, the problem of Boolean operations obfuscation or the error is caused by the modification of string literals in the previous steps) in order to repeat the obfuscation attempt.

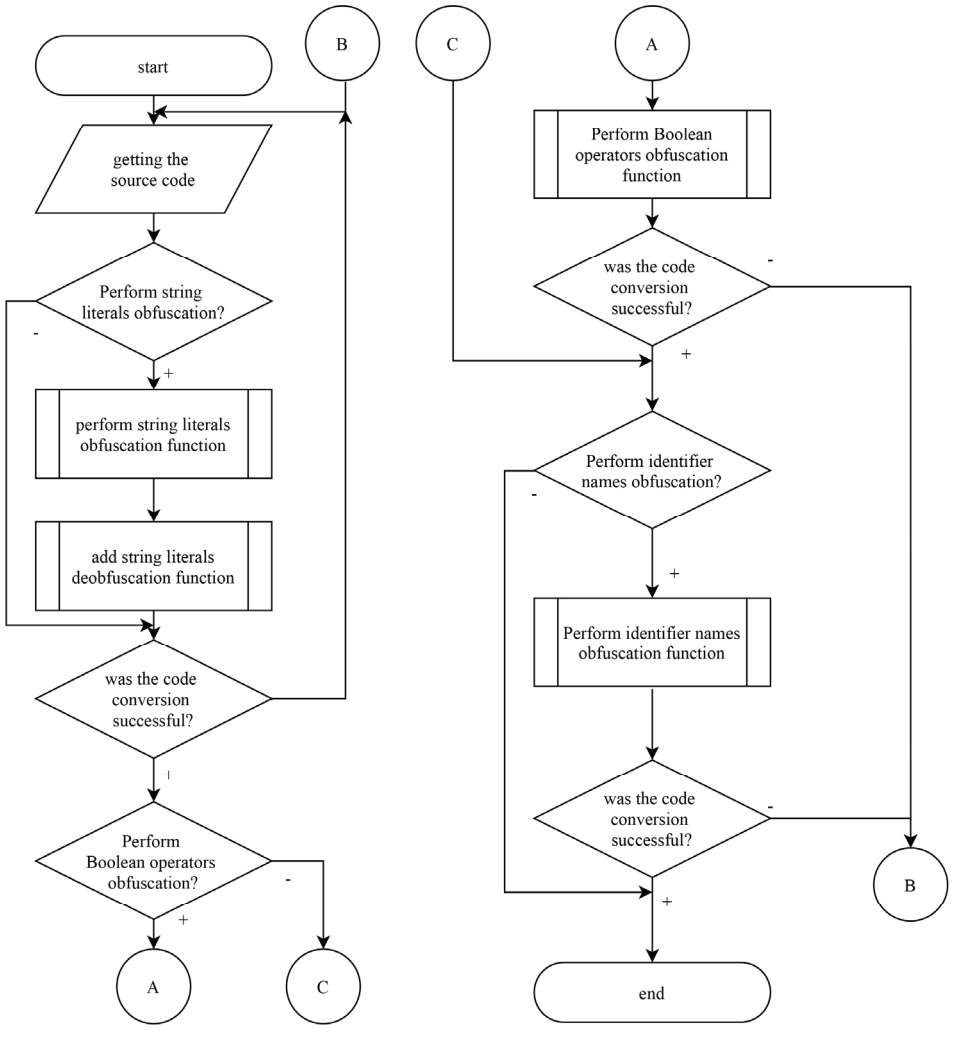

Fig. 2. Developed algorithm of programming module obfuscation

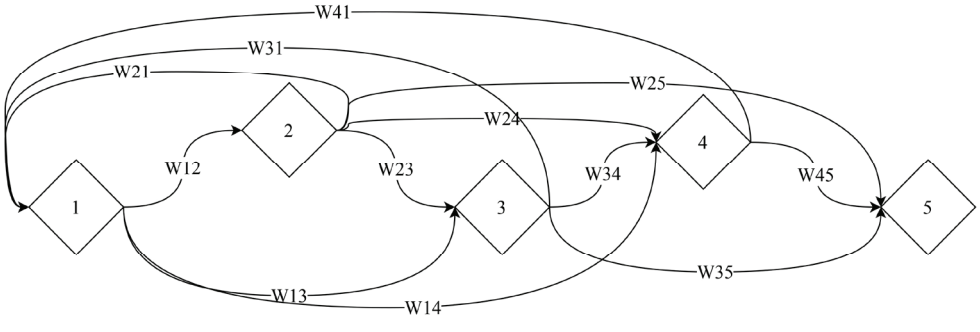

Fig. 3. Developed GERT network of programming modules obfuscation and deobfuscation processes

Thus, a set of algorithms of programming modules obfuscation and deobfuscation is synthesized, which made it possible to comprehensively describe these processes at the upper strategic level of formalization.

5. Development of a GERT model of the programming modules obfuscation process based on algorithms

The studies showed [18, 19] that the general algorithm of programming modules obfuscation has a number of specific iterations that greatly complicate the overall process of its mathematical formalization. Therefore, it seems appropriate to divide this process into a number of subprocesses. For mathematical modeling of obfuscation and deobfuscation processes, network stochastic models are the most flexible and useful. A special case of the stochastic model is a GERT network.

This is largely due to the availability of a mathematical apparatus for finding a continuous probability density function of the transition time of the GERT network. However, this is only possible provided that the set of distributions that can characterize individual arcs of the model includes known distributions. These are: discrete, binomial, Poisson, geometric, negative binomial, uniform, exponential, gamma and normal.

In addition, it is possible to find and use continuous arbitrary distributions. It is shown in [17] that the probability density function of the transition time of the GERT network is determined by the following expression:

$$\phi(x)= =\frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-isx}W_E(s)\mathrm{d}s, \quad (1)$$

where $W_E(s)$ is the equivalent transfer function of the GERT network, s is a real variable.

From the topological equation [17] follows:

$$W_E(t)=\frac{\sum\limits_{\gamma_1=1}^{\Gamma_1}\prod\limits_{\delta_1=1}^{\Delta_1}\breve{W}^1_{\gamma_1\delta_1}(t)+\ldots+(-1)^m\sum\limits_{\gamma_m=1}^{\Gamma_m}\prod\limits_{\delta_m=1}^{\Delta_m}\breve{W}^m_{\gamma_m\delta_m}(t)}{1-\sum\limits_{a_1=1}^{A_1}\prod\limits_{\beta_1=1}^{B_1}\bar{W}^1_{a_1\beta_1}(t)+\ldots+(-1)^l\sum\limits_{a_l=1}^{A_l}\prod\limits_{\beta_l=1}^{B_l}\bar{W}^l_{a_l\beta_l}(t)}, \quad (2)$$

where $\prod\limits_{\delta_i=1}^{\Delta_i}\breve{W}^i_{\gamma_i\delta_i}$ – the product of the W functions of arcs of the γ_i-th loop of the i-th order, including the sink t, $1\le i\le m$;

$\prod\limits_{\beta_j=1}^{B_j}\bar{W}^j_{a_j\beta_j}$ – the product of the W functions of the arcs of the a_j-th loop of the j-th order, not including the sink t, $1\le i\le l$;

Γ_i – the number of loops of orders i, including the network sink;

A_i – the number of loops of orders i, not including the network sink.

In a number of practically important cases, distributions must be obtained in the form of mathematical expressions. Such problems include the study of algorithms of programming code obfuscation and deobfuscation. This problem is reduced to finding the random distribution density function of transition time formed on the basis of the developed GERT network. Note that it must be assumed that in the continuous probability density function of the transition time of the GERT network, $\phi(x)$ is determined by the expression (1).

The W function of the transition between states i, j is determined by the formula:

$$W_{ij}(x)=P_{ij}\int_{-\infty}^{\infty}e^{isx}\zeta_{ij}(x)\mathrm{d}x, \tag{3}$$

where $\zeta_{ij}(x)$ is the probability density of transition between states i, j; P_{ij} is the probability of transition from state i to state j.

In the study, we adopt the hypothesis that the use of gamma distribution during modeling as a key one when describing probabilistic transitions from state to state will make it possible to achieve unification of the model of the programming modules obfuscation process. The unification is that reducing or increasing the number of obfuscation operations will slightly change the modeling results. It is expected that a decrease in the number of nodes will slightly decrease the variance and expectation, and an increase – accordingly, increase. However, there are restrictions on changing the model – the structural architecture of the model (for example, the degree of connectivity of the nodes) should remain unchanged.

Thus, in the considered GERT network of programming modules obfuscation and deobfuscation processes, the probability density function of transitions are defined by the gamma distribution with variable coefficients k and θ:

$$\zeta(x)=\frac{x^{k-1}\cdot e^{\frac{-x}{\theta}}}{\theta^k\cdot\Gamma(k)}. \tag{4}$$

The resulting W function has the following form:

$$W_E(s)=\\ =\frac{\bar{W}_1+\bar{W}_2+\bar{W}_3+\bar{W}_{12}+\bar{W}_{13}+\bar{W}_{23}+\bar{W}_{123}}{1-\left(\bar{W}_{1f}+\bar{W}_{2f}+\bar{W}_{3f}+\bar{W}_{12f}+\bar{W}_{13f}+\bar{W}_{23f}+\bar{W}_{123f}\right)}, \tag{5}$$

where:

$\bar{W}_1$ – the product of W functions describing the successful execution of only the string literals obfuscation algorithm:

$$\bar{W}_1=W_{12}W_{25}, \tag{6}$$

$\bar{W}_{1f}$ – the product of W functions describing the unsuccessful execution of only the string literals obfuscation algorithm:

$$\bar{W}_{1f}=W_{12}W_{21}, \tag{7}$$

$\bar{W}_2$ – the product of W functions describing the successful execution of only the Boolean functions obfuscation algorithm:

$$\bar{W}_2=W_{13}W_{35}, \tag{8}$$

$\bar{W}_{2f}$ – the product of W functions describing the unsuccessful execution of only the Boolean functions obfuscation algorithm:

$$\bar{W}_{2f}=W_{13}W_{31}, \tag{9}$$

$\bar{W}_3$ – the product of W functions describing the successful execution of only the identifier names obfuscation algorithm:

$$\bar{W}_3=W_{14}W_{45}, \tag{10}$$

$\bar{W}_{3f}$ – the product of W functions describing the unsuccessful execution of only the identifier names obfuscation algorithm:

$$\bar{W}_{3f}=W_{14}W_{41}, \tag{11}$$

$\bar{W}_{12}$ – the product of W functions describing the successful sequential execution of the string literals and Boolean functions obfuscation algorithm:

$$\bar{W}_{12}=W_{12}W_{23}W_{35}, \tag{12}$$

$\bar{W}_{12f}$ – the product of W functions describing the sequential execution of the string literals and Boolean functions obfuscation algorithm:

$$\bar{W}_{12f}=W_{12}W_{23}W_{31}, \tag{13}$$

$\bar{W}_{13}$ – the product of W functions describing the successful sequential execution of the string literals and identifier names obfuscation algorithm:

$$\bar{W}_{13}=W_{12}W_{24}W_{45}, \tag{14}$$

$\bar{W}_{13f}$ – the product of W functions describing the sequential execution of the string literals and identifier names obfuscation algorithm:

$$\bar{W}_{13f}=W_{12}W_{24}W_{41}, \tag{15}$$

$\bar{W}_{23}$ – the product of W functions describing the successful sequential execution of the Boolean functions and identifier names obfuscation algorithm:

$$\bar{W}_{23}=W_{13}W_{34}W_{45}, \tag{16}$$

$\bar{W}_{23f}$ – the product of W functions describing the sequential execution of the Boolean functions and identifier names obfuscation algorithm:

$$\bar{W}_{23f}=W_{13}W_{34}W_{41}, \tag{17}$$

$\bar{W}_{123}$ – the product of W functions describing the successful sequential execution of the string literals, Boolean functions, and identifier names obfuscation algorithm:

$$\bar{W}_{123}=W_{12}W_{23}W_{34}W_{45}, \tag{18}$$

$\bar{W}_{123f}$ – the product of W functions describing the sequential execution of the string literals, Boolean functions and identifier names obfuscation algorithm.

$$\bar{W}_{123f} = W_{12}W_{23}W_{34}W_{41}, \quad (19)$$

The formed table of characteristics of the branches considered in the GERT model of branches and distribution parameters is presented in Table 1.

As can be seen from the expression (4), the mathematical formalization of the resulting equivalent moment function seems to be a cumbersome expression. In this regard, the problem arises of a generalized mathematical formalization of the resulting expressions for calculating equivalent transfer functions.

Substituting the values of expressions (6)–(19) into the resulting W function (5), the formula of impressive size is obtained. For its "normalization", we introduce the following replacement:

$$p_{\bar{b}}^{\bar{a}}(t) = P_{a_1}\int_{-\infty}^{\infty} e^{isx}\frac{x^{k_{b_1}-1}\cdot e^{\frac{-x}{\theta_{b_1}}}}{\theta_{b_1}^{k_{b_1}}\cdot\Gamma\left(k_{b_1}\right)}\mathrm{d}x\times$$

$$\times P_{a_2}\int_{-\infty}^{\infty} e^{isx}\frac{x^{k_{b_2}-1}\cdot e^{\frac{-x}{\theta_{b_2}}}}{\theta_{b_2}^{k_{b_2}}\cdot\Gamma\left(k_{b_2}\right)}(x)\mathrm{d}x\cdot\ldots\times$$

$$\times P_{a_n}\int_{-\infty}^{\infty} e^{isx}\frac{x^{k_{b_n}-1}\cdot e^{\frac{-x}{\theta_{b_n}}}}{\theta_{b_n}^{k_{b_n}}\cdot\Gamma\left(k_{b_n}\right)}(x)\mathrm{d}x, \quad (20)$$

where $p_{\bar{b}}^{\bar{a}}(t)$ represents the product of W functions describing successful and unsuccessful execution of algorithms described by variables of expressions (6)–(19);

$\bar{a} = (a_1, a_2, a_3, .., a_n)$ – a list (array) of transition probabilities (loop arcs),

$\bar{b} = (b_1, b_2, b_3, .., b_n)$ – a list (array) of coefficients of the corresponding generating function of transition moments.

Thus, the resulting expression for calculating equivalent transfer functions can be described as:

$$W_E(t) = \frac{\left(p_{(1,2)}^{(1,6)}(t) + p_{(1,2)}^{(2,9)}(t) + p_{(1,1)}^{(3,1)}(t) + p_{(1,2,1)}^{(1,4,1)}(t) + p_{(1,1,2)}^{(1,5,9)}(t) + p_{(1,1,1)}^{(2,8,11)}(t) + p_{(1,1,1,1)}^{(1,5,8,11)}(t)\right)}{1-\left(p_{(1,3)}^{(1,7)}(t) + p_{(1,3)}^{(1,10)}(t) + p_{(1,3)}^{(1,12)}(t) + p_{(1,2,3)}^{(1,4,12)}(t) + p_{(1,1,3)}^{(1,5,10)}(t) + p_{(1,1,3)}^{(2,8,12)}(t) + p_{(1,1,1,3)}^{(1,5,8,12)}(t)\right)}. \quad (21)$$

Using the probability density function (4), we obtain the probability density graph shown in Fig. 4. At the same time, the probabilities were chosen as follows:

$P_1=P_2=P_3=1/3$; $P_4=0.05$; $P_5=0.8$;

$P_6=P_{12}=0.1$; $P_7=P_{10}=0.05$; $P_8=0.8$;

$P_9=0.15$; $P_{11}=0.9$.

The network can be built so that if in some state i it is possible to start one of several subsequent operations, then the probabilities of the start p_{ij} of any of these operations form a complete group of incompatible events:

$$\sum_j p_{ij} = 1, \forall i. \quad (22)$$

In this case, the probability of running the entire network from source to sink is 1.

To show that all nodes satisfy the condition (22), we calculate the probability of completing the entire process, which is calculated by the formula:

$$p_E = \left.\frac{W_E(s)}{M_E(s)}\right|_{s=0} = W_E(0) = 1, \quad (23)$$

where $M_E(s)$ is the generating function; and

$$M_E(s)\big|_{s=0} = \left.\int_{-\infty}^{\infty} e^{sx} f_E(x)\mathrm{d}x\right|_{s=0} = \int_{-\infty}^{\infty} f_E(x)\mathrm{d}x = 1, \quad (24)$$

where $f_E(x)$ are the distribution densities.

Table 1

Characteristics of transitions between the states of the GERT network of programming modules obfuscation and deobfuscation processes

No.	Branch	W function	Transition probability	Probability density coefficients (2)
1	(1, 2)	W_{12}	P_1	$k=k_1$; $\theta=\theta_1$
2	(2, 3)	W_{13}	P_2	$k=k_1$; $\theta=\theta_1$
3	(1, 4)	W_{14}	$P_3=1-P_1-P_2$	$k=k_1$; $\theta=\theta_1$
4	(2, 4)	W_{24}	P_4	$k=k_2$; $\theta=\theta_2$
5	(2, 4)	W_{23}	P_5	$k=k_1$; $\theta=\theta_1$
6	(3, 4)	W_{34}	P_8	$k=k_1$; $\theta=\theta_1$
7	(4, 5)	W_{45}	P_{11}	$k=k_1$; $\theta=\theta_1$
8	(3, 5)	W_{35}	P_9	$k=k_2$; $\theta=\theta_2$
9	(2, 5)	W_{25}	P_6	$k=k_2$; $\theta=\theta_2$
10	(4, 1)	W_{41}	$P_{12}=1-P_{11}$	$k=k_3$; $\theta=\theta_3$
11	(3, 1)	W_{31}	$P_{10}=1-P_8-P_9$	$k=k_3$; $\theta=\theta_3$
12	(3, 2)	W_{32}	$P_7=1-P_4-P_5-P_6$	$k=k_3$; $\theta=\theta_3$

Fig. 4 shows the density graph of the runtime of the entire programming module obfuscation and deobfuscation process taking into account variations in the values of k and θ. Integrating the probability density function, we obtain a distribution function, the graph of which is shown in Fig. 5.

The expectation and variance of the obtained functions are calculated according to the formulas:

$$\mu = W_E(t)\mathrm{d}t\big|_{t=0}, \quad (25)$$

$$\sigma^2 = W_E(t)\mathrm{d}^2t\big|_{t=0} - \mu^2. \quad (26)$$

The obtained calculation results are described in Table 2.

Thus, as part of the study, a unified GERT model of the programming modules obfuscation process is developed. This model differs from the known by the paradigm of using the mathematical apparatus of the gamma distribution as the key one at all stages of modeling the obfuscation process. This made it possible to achieve model unification in conditions of GERT network modification.

Table 2

Characteristics of the probability density function – expectation and variance

No.	k	θ	μ	σ^2
1	[1, 2, 1]	[0.7, 0.6, 1.5]	2.42	3.4
2	[1, 2, 1]	[2.7, 1.8, 3.5]	8.8	39.51
3	[2, 4, 3]	[2.7, 1.8, 3.5]	18.19	133.54
4	[2, 4, 3]	[0.7, 0.6, 1.5]	5.07	13.34

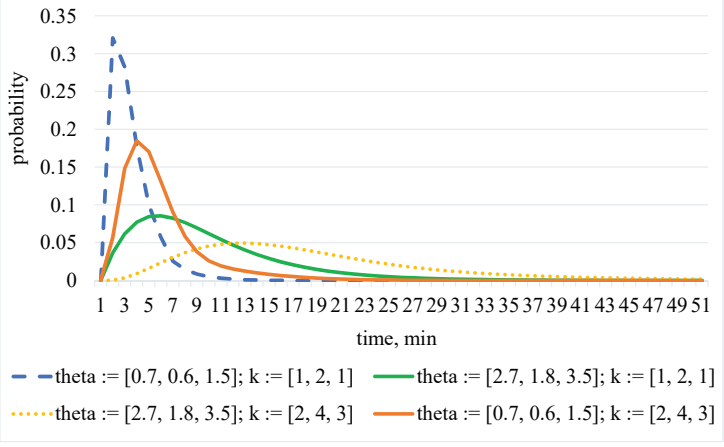

Fig. 4. Density graphs of the runtime of the programming modules obfuscation and deobfuscation process using the GERT network, gamma distribution of transition probabilities with different values of the coefficients k and θ

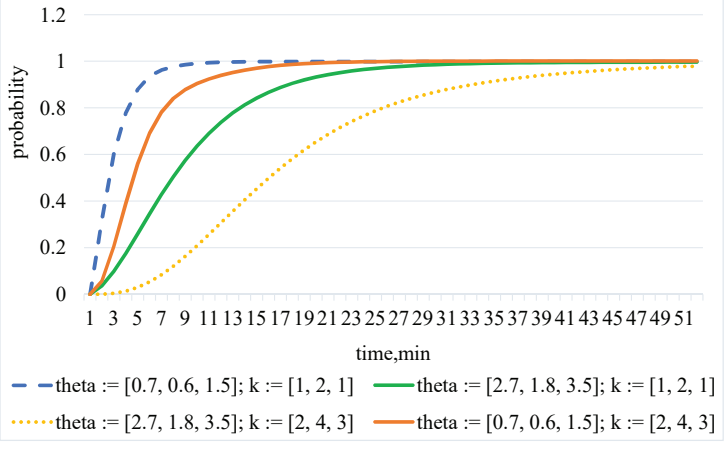

Fig. 5. Graphs of the distribution functions of the programming modules obfuscation and deobfuscation process using the GERT network, gamma distribution of transition probabilities with different values of the coefficients k and θ

6. Study of the unified GERT model with a modified number of nodes

The developed process of obfuscation and deobfuscation of programming modules consists of 5 nodes. Consider the behavior of the system when the number of nodes changes.

The developed GERT networks of programming modules obfuscation and deobfuscation processes with a changed number of nodes are presented in Fig. 6, 7.

When changing the number of nodes, the following factors were taken into account:

– the degree of connectivity of the nodes of the new process is comparable to the original one;

– changing the process complexity leads to a change in the number of elements of the array of coefficients k, while the values of the first and last elements of the array k are identical to the original ones;

– changing the process complexity leads to a change in the number of elements in the array of coefficients θ, while the values of the first and last elements of the array θ are identical to the original ones.

The generated tables of characteristics of the branches considered in the GERT model and distribution parameters are presented in Table 3, 4.

Using the expression (20), the resulting expressions for calculating equivalent transfer functions can be described as:

$$W_{E-1}(t)=\frac{p_{(1,2)}^{(1,4)}(t)+p_{(1,1,2)}^{(1,2,3)}(t)}{1-\left(p_{(1,3)}^{(1,5)}(t)+p_{(1,1,3)}^{(1,2,6)}(t)\right)}, \quad (26)$$

$$W_{E+1}(t)= =\frac{\begin{pmatrix} p_{(3,2)}^{(9,5)}(t)+p_{(3,2)}^{(12,8)}(t)+p_{(3,2)}^{(14,7)}(t)+ \\ +p_{(1,1,2)}^{(1,2,7)}(t)+p_{(1,1,3,2)}^{(1,2,11,5)}(t)+ \\ +p_{(1,3,2)}^{(1,10,5)}(t)+p_{(3,3,2)}^{(14,11,5)}(t)+ \\ +p_{(3,1,2)}^{(12,4,5)}(t)+p_{(1,1,1,2)}^{(1,2,3,8)}(t)+ \\ +p_{(3,1,2)}^{(14,3,8)}(t)+p_{(1,3,2)}^{(1,13,8)}(t)+ \\ +p_{(1,2)}^{(1,6)}(t)+p_{(1,1,1,1,2)}^{(1,2,3,4,5)}(t)+ \\ +p_{(1,3,1,2)}^{(1,13,4,5)}(t)+p_{(3,1,1,2)}^{(14,3,4,5)}(t) \end{pmatrix}}{1-\begin{pmatrix} p_{(1,4)}^{(1,15)}(t)+p_{(1,1,4)}^{(1,2,16)}(t)+p_{(1,1,1,4)}^{(1,2,3,17)}(t)+ \\ +p_{(1,1,1,1,4)}^{(12,3,4,18)}(t)+p_{(3,4)}^{(14,16)}(t)+ \\ +p_{(1,3,4)}^{(1,3,17)}(t)+p_{(3,1,4)}^{(14,3,17)}(t)+ \\ +p_{(3,4)}^{(12,17)}(t)+p_{(3,4)}^{(9,18)}(t)+ \\ +p_{(1,3,4)}^{(1,10,18)}(t)+p_{(3,3,4)}^{(14,11,18)}(t)+ \\ +p_{(3,1,4)}^{(12,4,18)}(t)p_{(1,1,3,4)}^{(1,2,11,18)}(t)+ \\ +p_{(1,3,1,4)}^{(1,13,4,18)}(t)+p_{(3,1,1,4)}^{(14,3,4,18)}(t) \end{pmatrix}}. \quad (27)$$

For each new process, the probability of implementing the entire process p_E, which is equal to 1, was calculated. The values of expectation and variance were also calculated, presented in Table 5.

Table 3

Characteristics of transitions between the states of the GERT network of programming modules obfuscation and deobfuscation processes with a reduced number of operating obfuscation functions

No.	Branch	W function	Transition probability	Probability density coefficients (2)
1	(1, 2)	W_{12}	P_1	$k=k_1$; $\theta=\theta_1$
2	(2, 3)	W_{13}	P_2	$k=k_1$; $\theta=\theta_1$
3	(3, 4)	W_{34}	P_3	$k=k_2$; $\theta=\theta_2$
4	(2, 4)	W_{23}	P_4	$k=k_3$; $\theta=\theta_3$
5	(2, 1)	W_{41}	$P_5=1-P_2-P_4$	$k=k_3$; $\theta=\theta_3$
6	(3, 1)	W_{31}	$P_6=1-P_3$	$k=k_3$; $\theta=\theta_3$

Based on the obtained equivalent transfer functions, the density graphs of the runtime of the entire process of obfuscation and deobfuscation of the programming module were constructed taking into account variations in the variables k and θ. These graphs, as well as the corresponding graphs of the distribution function, are presented in Fig. 8, 9.

Table 4

Characteristics of transitions between the states of the GERT network of programming modules obfuscation and deobfuscation processes with additional operating obfuscation function

No.	Branch	*W* function	Transition probability	Probability density coefficients (2)
1	(1, 2)	W_{12}	P_1	$k=k_1$; $\theta=\theta_1$
2	(2, 3)	W_{13}	P_2	$k=k_1$; $\theta=\theta_1$
3	(3, 4)	W_{34}	P_3	$k=k_1$; $\theta=\theta_1$
4	(4, 5)	W_{45}	P_4	$k=k_1$; $\theta=\theta_1$
5	(5, 6)	W_{56}	P_5	$k=k_2$; $\theta=\theta_2$
6	(2, 6)	W_{26}	P_6	$k=k_2$; $\theta=\theta_2$
7	(3, 6)	W_{36}	P_7	$k=k_2$; $\theta=\theta_2$
8	(4, 6)	W_{46}	P_8	$k=k_2$; $\theta=\theta_2$
9	(1, 5)	W_{15}	P_9	$k=k_3$; $\theta=\theta_3$
10	(2, 5)	W_{25}	P_{10}	$k=k_3$; $\theta=\theta_3$
11	(3, 5)	W_{35}	P_{11}	$k=k_3$; $\theta=\theta_3$
12	(1, 4)	W_{14}	P_{12}	$k=k_3$; $\theta=\theta_3$
13	(2, 4)	W_{24}	P_{13}	$k=k_3$; $\theta=\theta_3$
14	(1, 3)	W_{13}	$P_{14}=1-P_1-P_{12}-P_9$	$k=k_3$; $\theta=\theta_3$
15	(2, 1)	W_{21}	$P_{15}=1-P_6-P_2-P_{13}-P_{10}$	$k=k_4$; $\theta=\theta_4$
16	(3, 1)	W_{31}	$P_{16}=1-P_3-P_7-P_{11}$	$k=k_4$; $\theta=\theta_4$
17	(4, 1)	W_{41}	$P_{17}=1-P_4-P_8$	$k=k_4$; $\theta=\theta_4$
18	(5, 1)	W_{51}	$P_{18}=1-P_5$	$k=k_4$; $\theta=\theta_4$

Table 5

Probability density characteristics – expectation and variance

No.	Type	μ	σ^2
1	Original θ=[2.7, 1.8, 3.5]; *k*=[1, 2, 1]	8.8	39.51
2	Added node θ=[2.7, 1.8, 1.8, 3.5]; *k*=[1, 2, 2, 1]	11.11	44.68
3	Removed node θ=[2.7, 3.5]; *k*=[1, 1]	8.08	34.88

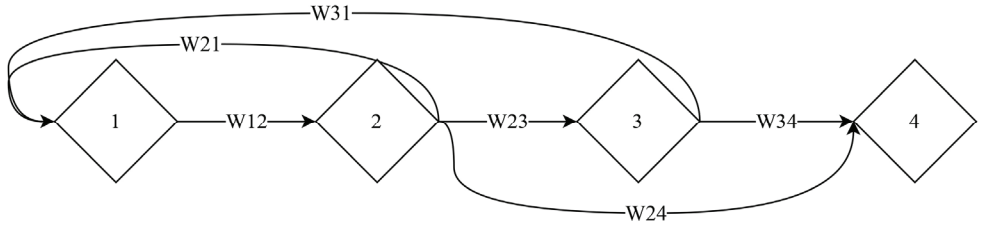

Fig. 6. Developed GERT network of programming modules obfuscation and deobfuscation processes with a reduced number of operating obfuscation functions

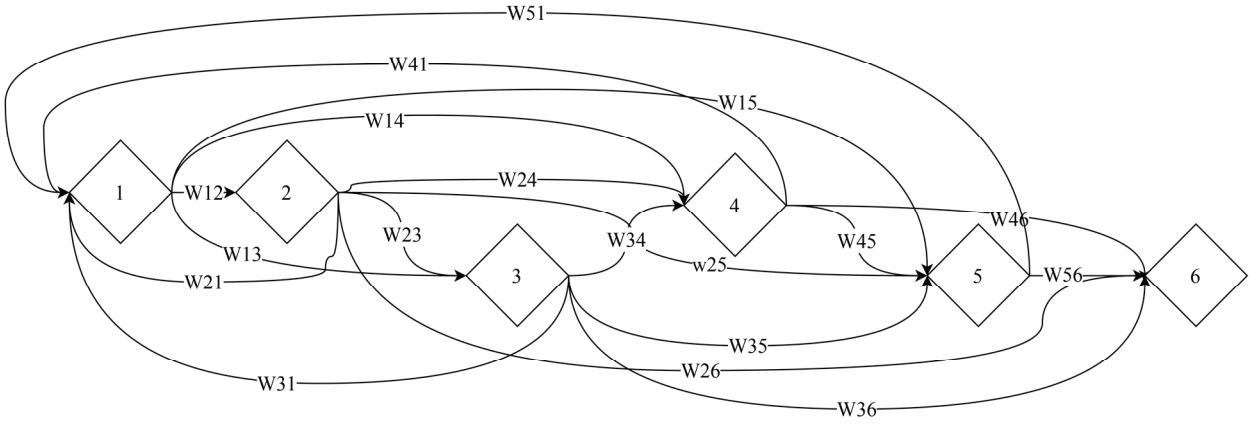

Fig. 7. Developed GERT network of programming modules obfuscation and deobfuscation processes with additional operating obfuscation function

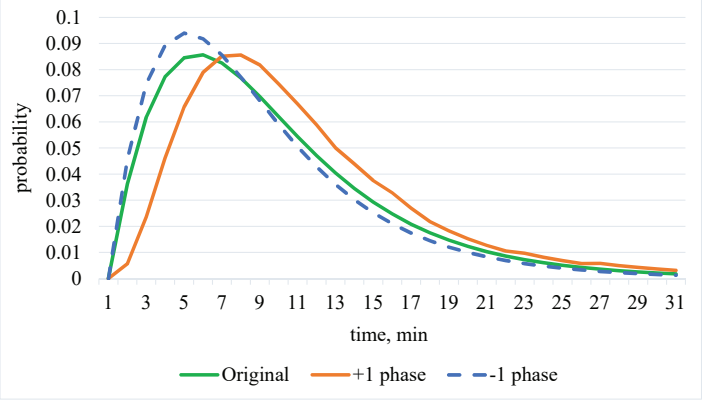

Fig. 8. Density graphs of the runtime of the programming modules obfuscation and deobfuscation process when using the GERT network, gamma distribution of transition probabilities

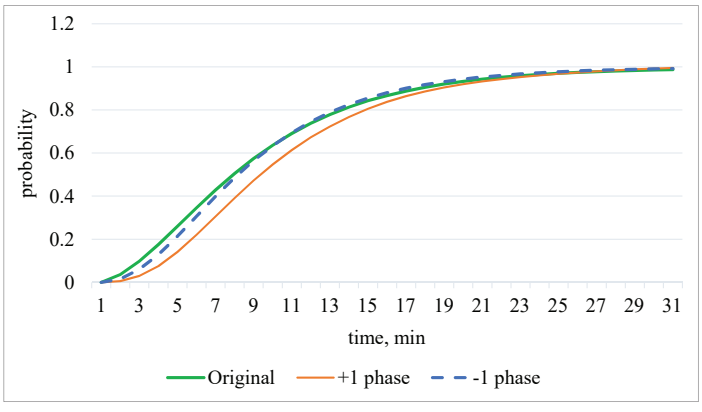

Fig. 9. Graphs of distribution functions of the programming modules obfuscation and deobfuscation process when using the GERT network, gamma distribution of transition probabilities

The results of the study showed that for the developed mathematical model, when adding another obfuscation process, the variance increases by 12 %, and when it is removed from the system, it decreases to 13 %. The expectation changes exponentially. So, when removing the node, the expectation decreases by 9 %, and when increasing by 1 node, the expectation increases by 26 %. This shows the insignificance of changes in the studied characteristics under the conditions of model modification and confirms the hypothesis of model unification when using the mathematical apparatus of gamma distribution as the main one. These results allow the developer to predict the behavior of the programming modules protection system in terms of runtime.

7. Discussion of the results of the study of the developed GERT model of the programming modules obfuscation process

A set of algorithms of programming modules obfuscation is synthesized (Fig. 1), which differs from the known ones by taking into account the variability of data types. The synthesis made it possible to present the obfuscation process as a whole (Fig. 2), as well as to formalize the obfuscation process in a convenient form for subsequent use in the developed GERT models (Fig. 3).

As part of the study, a unified GERT model of the programming modules obfuscation process is developed. This model differs from the known ones by the paradigm of using the mathematical apparatus of gamma distribution (4) as a key one at all stages of modeling the obfuscation process. This made it possible to achieve model unification under the conditions of GERT network modification (Fig. 4, 5). Unification allows adapting the modeling process to a possible complication of the structure and algorithms of obfuscation processes.

The results of the study showed that for the developed mathematical model, when adding another obfuscation process, the runtime variance increases by 12 %, and when removed from the system it decreases to 13 % (Fig. 8, 9). The runtime expectation changes exponentially. So, when removing the node, the expectation decreases by 9 %, and when increasing by 1 node, the expectation increases by 26 %. This shows the insignificance of changes in the studied characteristics under the conditions of model modification and confirms the hypothesis of model unification when using the mathematical apparatus of gamma distribution as the main one. These results allow the developer to predict the behavior of the programming modules protection system in terms of runtime. This allows reducing the time to decide on the feasibility of the obfuscation process when using flexible methodologies.

The studies show that the developed mathematical model is appropriate for mathematical modeling of systems that are formalized by at least four states. A decrease in the number of states leads to linearization of the process, for which the use of stochastic approaches to mathematical modeling leads to a deterioration in the accuracy of the results. Also, an additional decrease in the number of states leads to a decrease in the security of programming modules (Fig. 8, 9). So, Fig. 6 shows the four-state model described by six transitions (Table 3). Reducing the number of states by 1 will lead to a system having 3 transitions.

The recommended maximum number of states is 9 nodes. A further increase in the number of nodes leads to an excessive complication of the mathematical model, while the trends of varying the process runtime expectation and variance remain.

At this stage, the coefficients k, θ are selected empirically using expert knowledge. So, the input data are obtained as a result of an experiment conducted by a group of expert developers of NixSolutions secure software. Further, this limitation can be eliminated by calculating these coefficients for specific data protection algorithms. The elimination of these restrictions is associated with the direction of further research, which should be focused on the development of the procedure for adapting these coefficients to various business processes of programming modules obfuscation.

The development of this study consists in the design of a methodology for calculating the gamma distribution and its adaptation for the practical implementation of data protection algorithms. However, difficulties may arise associated with the existing limitations of stochastic modeling approaches.

8. Conclusions

1. A set of algorithms of obfuscation and deobfuscation of programming modules is synthesized, which differs from the known ones by taking into account the variability of data types. This made it possible to describe these processes at the upper strategic level of formalization.

2. As part of the study, a unified GERT model of the programming modules obfuscation process is developed. This model differs from the known ones by the paradigm of using the mathematical apparatus of gamma distribution as the key one at all stages of modeling the obfuscation process. This made it possible to achieve model unification in the conditions of GERT network modification.

3. The results of the study showed that for the developed mathematical model, when adding another obfuscation process, the runtime variance increases by 12 %, and when removed from the system it decreases to 13 %. The runtime expectation changes exponentially. So, when removing the node, the expectation decreases by 9 %, and when increasing by 1 node, the expectation increases by 26 %. This shows the insignificance of changes in the studied characteristics under the conditions of model modification and confirms the hypothesis of model unification when using the mathematical apparatus of gamma distribution as the main one. These results allow the developer to predict the behavior of the programming modules protection system in terms of runtime. This allows reducing the time to decide on the feasibility of the obfuscation process when using flexible methodologies.

References

1. On Protection of Information in Automated Systems (1994). Verkhovna Rada of Ukraine. Available at: https://zakon.rada.gov.ua/laws/show/80/94-%D0%B2%D1%80#top
2. On Copyright and Related Rights (1994). Verkhovna Rada of Ukraine. Available at: https://zakon.rada.gov.ua/laws/show/3792-12#Text
3. Galatenko, V. A. (2016). Osnovy informatsionnoy bezopasnosti. Moscow: Natsional'niy Otkrytiy Universitet «INTUIT», 267.
4. Raskin, L. G., Pustovoytov, P. E., Abdel'hamid Saed Ahmad, S. A. (2006). Markovskaya approksimatsiya nemarkovskih sistem. Informatsiyno-keruiuchi systemy na zaliznychnomu transporti, 1, 57–60. Available at: http://repository.kpi.kharkov.ua/bitstream/KhPI-Press/6801/1/2006_Raskin_Markovskaya.pdf
5. Denning, P. J., Lewis, T. G. (2016). Exponential laws of computing growth. Communications of the ACM, 60 (1), 54–65. doi: https://doi.org/10.1145/2976758
6. Strzałka, D., Dymora, P., Mazurek, M. (2018). Modified stretched exponential model of computer system resources management limitations – The case of cache memory. Physica A: Statistical Mechanics and Its Applications, 491, 490–497. doi: https://doi.org/10.1016/j.physa.2017.09.012
7. Kovalenko, O. V., Hochkin, N. I. (2015). Solution of the system of renewal Markov equations using the approximation of asymptotic series. Trudy MFTI, 7 (2), 5–19. Available at: https://mipt.ru/upload/medialibrary/26d/5-19.pdf
8. Lacasa, L., Mariño, I. P., Miguez, J., Nicosia, V., Roldán, É., Lisica, A. et. al. (2018). Multiplex Decomposition of Non-Markovian Dynamics and the Hidden Layer Reconstruction Problem. Physical Review X, 8 (3). doi: https://doi.org/10.1103/physrevx.8.031038
9. Distefano, S., Longo, F., Scarpa, M. (2017). Marking dependency in non-Markovian stochastic Petri nets. Performance Evaluation, 110, 22–47. doi: https://doi.org/10.1016/j.peva.2017.03.001
10. Jiang, S., Yang, S. (2016). An Improved Multiobjective Optimization Evolutionary Algorithm Based on Decomposition for Complex Pareto Fronts. IEEE Transactions on Cybernetics, 46 (2), 421–437. doi: https://doi.org/10.1109/tcyb.2015.2403131
11. Semenov, S., Sira, O., Kuchuk, N. (2018). Development of graphicanalytical models for the software security testing algorithm. Eastern-European Journal of Enterprise Technologies, 2 (4 (92)), 39–46. doi: https://doi.org/10.15587/1729-4061.2018.127210
12. Sheng, Z., Hu, Q., Liu, J., Yu, D. (2017). Residual life prediction for complex systems with multi-phase degradation by ARMA-filtered hidden Markov model. Quality Technology & Quantitative Management, 16 (1), 19–35. doi: https://doi.org/10.1080/16843703.2017.1335496
13. Hu, L., Liu, Z., Hu, W., Wang, Y., Tan, J., Wu, F. (2020). Petri-net-based dynamic scheduling of flexible manufacturing system via deep reinforcement learning with graph convolutional network. Journal of Manufacturing Systems, 55, 1–14. doi: https://doi.org/10.1016/j.jmsy.2020.02.004

14. Semyonov, S. G., Gavrilenko, S. Y., Chelak, V. V. (2017). Information processing on the computer system state using probabilistic automata. 2017 2nd International Ural Conference on Measurements (UralCon). doi: https://doi.org/10.1109/uralcon.2017.8120680
15. Burgelman, J., Vanhoucke, M. (2019). Computing project makespan distributions: Markovian PERT networks revisited. Computers & Operations Research, 103, 123–133. doi: https://doi.org/10.1016/j.cor.2018.10.017
16. Kovalenko, O. (2017). DOM XSS vulnerability testing technology. Ukrainian Scientific Journal of Information Security, 23 (2), 73–79. doi: https://doi.org/10.18372/2225-5036.23.11821
17. Shibanov, A. P. (2003). Finding the Distribution Density of the Time Taken to Fulfill the GERT Network on the Basis of Equivalent Simplifying Transformations. Automation and Remote Control, 64, 279–287. doi: https://doi.org/10.1023/A:1022267115444
18. Venkatesh, S., Ertaul, L. (2005). Novel Obfuscation Algorithms for Software Security. Proceedings of the 2005 International Conference on Software Engineering Research and Practice, 1, 209–215.
19. Mohsen, R., Miranda Pinto, A. (2015). Algorithmic Information Theory for Obfuscation Security. Proceedings of the 12th International Conference on Security and Cryptography. doi: https://doi.org/10.5220/0005548200760087