Algorithm for predicting the health of an information system after exposure to cyberattacks

A V Dushkin¹, S S Kochedykov²

¹National Research University of Electronic Technology, Moscow, Zelenograd, Shokin square, 1, 124498, Russia

²Voronezh Institute of High Technologies, Voronezh, Lenin str., 73A, 394043, Russia

E-mail a_dushkin@mail.ru

Abstract. The presented work proposes a systematic approach to the development of software for cyberattack detection systems and predicting the health of the information system. A formalized descriptive model of an intrusion detection system is presented. A complex of models is described that allow making diagnostic decisions about the state of software performance.

Keywords: Information security; Information system; Information; Cyberattack; Software

1. Introduction
The analysis performed allows us to state the fact that the information system (IS) [1] is an object of destructive influences (DI) aimed at violating information security. An example of such destructive information influences are cyber attacks [2], with the help of which criminals try to penetrate the databases, block or disable electronic objects, disrupt the work of the security service, destabilize the work of the IS. In these conditions, in the course of the structural-parametric synthesis of IS, the issues of comprehensive information security support come to the fore by: a) timely detection, identification and monitoring of destructive influences on IS [3]; b) operational control of the functional efficiency of the IS in conditions of destructive influences [4]; c) assessing the impact of cyberattacks on the functioning of IS and managing protection measures and actively countering cyberattacks [5].

2. Formulation of the problem
We represent the structure of the IS in the form of a classical unit [6-8]. With its help, it is possible to give a unified description of the deterministic and stochastic objects both in continuous and in discrete time, subject to development. For the purpose of a formalized description of information processing processes in an IS and counteraction to destructive influences, we will proceed from the fact that a set of functional components (FC) make up the designed IS. They can be developed independently or purchased from other manufacturers. At the same time, the set of externally system characteristics of these functional components (FC) is set by control commands, performed functions, needs for information and resource support [9-11].

It is necessary to develop an intrusion detection system (IDS) model that can be used to describe the processes and algorithms for diagnosing the operability of IS software.
3. Formalized description of information processing processes and intrusion detection systems

Let’s depict in fig. 1 the structure of the IS software, taking into account the parameters controlled by the IDS.

![Diagram](image)

Figure 1. Block diagram of the IS software, taking into account the parameters controlled by IDS

A description of a structured IDS that maintains the required level of security in a cyberattack environment. \( C_{I,Is} = \{c_{I,Is} \times T\} \) - set of control actions \((C_{I,Is} \subseteq C_I \subseteq C)\) on the interface at different points in time \( T \) from the FC control subsystem (IDS) at the level \( I \). With its help, there is a rational distribution of \( I \) level resources \( c_{I,Is} : R_I \rightarrow \{R_{I,Is}\} \), as well as control of the parameters of the interface functioning \( O_{I,Is} \).

When determining the efficiency of the interface, it is necessary to adhere to the following rule: all quality indicators must be higher or equal to the required values:

\[
\{q_{I,Is} \geq q_{I,Is}^{\text{required}}\} \in Q_{I,Is}.
\]  

(1)

The compliance of all QoS indicators from the set \( Q_{I,Is} \) with the efficiency criterion (1) determines the quality of the interface.

The required level of the criterion can be provided by a variety of control actions \( C_{I,Is} \) that depend on the allocation of interface \( Isc_I \) resources \( R_{I,Is} \), or by changing the parameters \( P_{a_{I,Is}} \) of the algorithms \( A_{I,Is} \) on the basis of which the interface functions.

The software quality control subsystem, implemented by IDS through the interface \( Isc_I \) observation channel \( Ch_{I,Isc}^{(mon)} \), collects data on the interface state \( pos_{I,Is} \) by fully or partially determining the output parameters \( P_{I,Is}^{(out)} \) and quality indicators \( Q_{I,Is} \) for the interface \( Isc_I \).

At the same time, the purpose of monitoring is to determine the state \( pos_{I,Is} \) of the interface \( Isc_I \) and control the impact accordingly \( C_{I,Is} \). Channel observation \( Ch_{I,Isc}^{(mon)} \) can be represented by a vector that includes many observed parameters:

\[
CH_{I,Isc}^{(mon)} = \{P_{I,Is,Cha}^{(out)} \times Q_{I,Is,Cha}\},
\]  

(2)

where \( P_{I,Is,Cha}^{(out)} \subseteq P_{I,Is}^{(out)} \) and \( Q_{I,Is,Cha} \subseteq Q_{I,Is} \) are the components of the sets \( P_{I,Is}^{(out)} \) and \( Q_{I,Is} \) determined by the software quality control subsystem.

Current state of the interface \( pos_{I,Is} \) under parametric narrowing conditions...
\[
p_{\text{fc}} \rightarrow \{ \{ q_{\text{fc}} \}, \{ p_{\text{fc}}^{(i)} \} \} \rightarrow \{ \{ q_{\text{Ch1 fc}} \}, \{ p_{\text{Ch1 fc}}^{(i)} \} \} \rightarrow \text{Ch1}_{\text{fc}}^{(\text{mon})}
\]

and the narrowing of the reaction due to the limitation of the observation time \( 0 \leq T \leq t^{\text{observation}} \) in expression (2) can be determined on the basis of the theory of observation of dynamic systems [12-14] and control systems [15].

Based on the foregoing, we will consider the features of diagnosing the operability of the FC software of the IS following dynamic model.

4. Development of a complex of dynamic models

Based on the results of the model for diagnosing the operability of the IS software, a dynamic model is constructed, the purpose of which is to describe the cause-and-effect relationships between the IS FC, taking into account the space-time dependencies.

To create a DM, we will use the apparatus of Petri-Markov nets [16, 17]:

\[
M_{\text{d}} = \{ \Psi, \Omega \}. \tag{4}
\]

For the most complete description of the structure of the diagnostic process, it is advisable to use a Petri net

\[
\Psi = \{ I, O, A, Z \}, \tag{5}
\]

where we introduce the following notation of finite sets: \( I \) – system inputs; \( O \) – system outputs; \( A \) – positions; \( Z \) – transitions.

The process that determines the probabilistic characteristics of software diagnostics is denoted as

\[
\Omega = \{ P, f(p), \Lambda \}, \tag{6}
\]

where \( P \) is the probability matrix; \( f(p) \) is the matrix of the probability distribution function; \( \Lambda \) is a matrix of logical conditions.

In fig. 2 shows a diagram of making diagnostic decisions. It displays the operation of the software diagnostics algorithm and risk assessment of the system operability [18, 19].

The stages of the diagnostic process are the vertices of the graph. The sequence of actions is as follows: obtaining diagnostic data \( \alpha_1 \); additional tests \( \alpha_2 \); determination of functional diagnostic parameters \( \alpha_3 \); \( \alpha_4 \) data patterning; application of an alternative method of forming the \( \alpha_5 \) data pattern; state classification of IC \( \alpha_6 \); application of an alternative method for classifying the state of the IC \( \alpha_7 \); interpretation of \( \alpha_8 \) classification results.

The matrix of transition probabilities of the dynamic model can be represented in the following form:

\[
P = \begin{pmatrix}
  p_{11} & p_{12} & p_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
\end{pmatrix}. \tag{7}
\]
In this case, the vertices of the graph \( (a_1, \ldots, a_k) \) form the rows of the matrix, and the transitions \( (z_1, \ldots, z_{10}) \) – the columns. Transition \( z_1 \) will be triggered with probability \( p_{11} = 1 \) in case of receiving input information \( I_1 \). Since the transitions \( z_2 \) and \( z_3 \) are asynchronous, then

\[
p_{13} = 1 - p_{12}.
\]

By analogy for asynchronous transitions \( z_4, z_5; z_6, z_7; \) and \( z_8, z_9 \) can be written:

\[
\begin{align*}
p_{35} &= 1 - p_{34}; \\
p_{47} &= 1 - p_{46}; \\
p_{69} &= 1 - p_{68}.
\end{align*}
\]

Reachability condition for chain vertices

\[
\sum_{ij} p_{ij} > 0
\]

can be performed by introducing feedbacks for each step of the transition conditions. This leads to the fact that the sum of the probabilities of transitions along the columns of the matrix is always nonzero.

The resulting dynamic model can be used as the basis for a generalized classification algorithm.

In fig. 3 shows the decision-making logic for determining the degree of software performance in the form of a Markov chain.

![Graph for determining the degree of software performance](image)

Figure 3. Graph for determining the degree of software performance

Here, the health of the software is determined by the health of each component. Accordingly, the condition for the transition of \( N \) to the next degree \( S \) is sequentially checked. When constructing the graph, the following designations are adopted: \( N_{i1} \) – the component failure level is minimal, that is, the time of the component \( i \) function \( f \) execution \( t_f \leq \min(t_f) \); \( N_{i2} \) – component failure level is higher than the minimum; \( N_{i3} \) – average component failure level; \( N_{i4} \) – component failure rate above average; \( N_{i5} \) – component failure level is high; \( N_{i6} \) – maximum level, when the time of the component \( i \) function \( f \) execution \( t_f \geq \max(t_f) \); \( N_{i1}, N_{i2}, N_{i3}, N_{i4}, N_{i5}, N_{i6} \) – respectively, automatic elimination of a component malfunction in a short period of time, for a sufficient period of time, for a non-critical period of time, for a significant period of time, for a long period of time, requiring an additional function; \( N_{i6} \) – the malfunction is not eliminated, a complete software replacement is required for a long period of time.

Thus, the probabilities of diagnosing software operability are respectively equal:

\[
P_{S1} = p_{N_{i1}1} + p_{N_{i1}d1};
\]

\[
P_{S6} = p_{N_{i6}1} + p_{N_{i6}d6}.
\]

where \( p_{S1}, p_{S2}, p_{S3} \) – the probability of determining, respectively, the optimal, normal and high levels of software performance; \( p_{S4}, p_{S5}, p_{S6} \) – the probability of determining, respectively, degrees 1, 2, 3 of software failure. In Fig. 4, the graph for determining the diagnostic results of the software operability is presented, corresponding to the decision logic.
Depending on the degree of damage to target components in accordance with the methodology [20],
the following classification can be carried out: $M_1$ – the functions of the software components are not
impaired or slightly impaired; $M_2$ – functions of software components are moderately impaired; $M_3$ –
the functions of software components are significantly impaired.

![Figure 4. Graph for determining the diagnostic results of software performance](image)

Accordingly, the probabilities of diagnosing the degree of software operability are respectively equal
to:

$$
P_{D1} = p_{S4} + p_{M1};
$$
$$
P_{D2} = p_{S5} + p_{M2};
$$
$$
P_{D3} = p_{S6} + p_{M3};
$$

where $p_{D1}$ – probability of software malfunction (stage I); $p_{D2}$ – probability of software malfunction
(stage II); $p_{D3}$ – probability of software malfunction (stage III).

The decision-making logic in the form of a directed graph for determining the risk of software
performance is shown in Fig. 5. At the same time, the following factors influence the level of risk in
accordance with the methodology [21]: the degree of software performance in the presence of risk
factors $F$; defeat of target components $M$; list of faults $A$ of the IS associated with failure or partial
failure of the corresponding components.

![Figure 5. Oriented graph for determining the risk of software performance](image)

The corresponding probabilities of determining the level of risk of software performance can be
represented as follows:

$$
p_{R1} = p_{S4} r_1;
$$
$$
p_{R2} = p_{S5} r_2 = p_{S6} r_2 = p_{S4} F r_2 = p_{S5} F r_2 = p_{S6} F r_2 = p_{S6} F + p_{FR2};
$$
$$
p_{R3} = p_{FM} + p_{MR3} = p_{SM} + p_{MR3} = p_{SM} + p_{MR3} = p_{S6} M + p_{MR3};
$$
$$
p_{R4} = p_{MA} + p_{ARA} = p_{SA} + p_{ARA} = p_{SA} + p_{ARA} = p_{S6} A + p_{ARA};
$$

where $p_{R1}, p_{R2}, p_{R3}, p_{R4}$ – respectively, the probabilities of determining risks 1, 2, 3, 4.
Thus, the logic of the operation of the intrusion detection system subsystems that control and diagnose is described by a set of proprietary dynamic models and makes it possible to make a decision on the operability of the software. Probabilistic estimates obtained in the course of calculations can be used to determine the prior probabilities of diagnostic estimates. Thus, the paper describes a set of models that allow making diagnostic decisions about the state of software performance.

5. Conclusion
This paper describes the algorithm and models for diagnosing the IS operability in the context of cyber attacks. The implementation of these proposals in the form of software products makes it possible to start designing new intrusion detection systems for ICs that allow continuous diagnostics of the software performance. A complex of mathematical models, with the help of which it is possible to describe the processes of making managerial decisions when diagnosing IS components, make it possible to determine the level of software performance; assess risks taking into account risk factors, the health of software components and the list of IC malfunctions associated with the failure or partial failure of the corresponding components. Thus, the paper describes a set of models that make it possible to make diagnostic decisions about the state of software performance in the context of cyber attacks.

References
[1] National institute of standards and technology & national security agency (USA), Communication security establishment (Canada), Communications-electronics security group (United Kingdom), Bundesamt fur sichereit in der informationstechnik (Germany), Service central de la secure des systemes d'information (France), National communications security agency (Netherlands) 1999 Common criteria for information technology security evaluation Version 2.1 – URL: http://www.commoncriteriaportal.org/files/ccfiles/ccpart1v21.pdf
[2] Sowa J F 2020 Semantic Networks – URL: http://www.jfsowa.com/pubs/semnet.htm
[3] Kostogryzov A, and Korolev V 2020 Probability, combinatorics and control (UK London: IntechOpen) p 322 – DOI: 10.5772/intechopen.79802
[4] Kostogryzov A 2018 Probabilistic modeling in system engineering (UK London: IntechOpen) p 278 – DOI: 10.5772/intechopen.71396
[5] Novoseltsev V I 2017 Tool and algorithmic diagnostic devices of operability of actuation mechanisms of automated control systems Proceedings: 2nd International Ural conference on measurements (Chelyabinsk) pp 193-198 – DOI: 10.1109/URALCON.2017.8120709
[6] Kasatkina T I, Novoseltsev V I and Ivanov S A 2018 An improved method for predicting the evolution of the characteristic parameters of an information system Journal of Physics: Conference Series 973 012031 – DOI: 10.1088/1742-6596/973/1/012031
[7] Pavlov V A, Kasatkina T I and Shatovkin R R 2018 Algorithm for predicting the evolution of series of dynamics of complex systems in solving information problems Journal of Physics: Conference Series 973 012035 – DOI: 10.1088/1742-6596/973/1/012035
[8] Noev A N and Gubin I A 2018 Method of optimum channel switching in equipment of infocommunication network in conditions of cyber attacks to their telecommunication infrastructure Journal of Physics: Conference Series 1015 032101 – DOI: 10.1088/1742-6596/1015/3/032101
[9] Sumin V I and Smolentseva T E 2018 Mathematical Models to determine stable behavior of complex systems Journal of Physics: Conference Series 1015 032136 – DOI: 10.1088/1742-6596/1015/3/032136
[10] Sumin V I, Grechishnikov E V and Ivanov S V 2019 Determining the reliability of network information systems Journal of Physics: Conference Series 1203 012083 – DOI: 10.1088/1742-6596/1203/1/012083
[11] Goncharov I V, Goncharov N I and Parinov P A 2019 Opportunities to study the characteristics of information systems and manage them using modern technologies International conference
Actual problems of system and software engineering (Moscow) pp 95-105 – URL: http://ceur-ws.org/Vol-2514/paper55.pdf

[12] Kobzistyy S Yu and Markin P V 2019 Method of analysis the functional stability of the software of the infocommunication system in cyber-attack conditions International conference actual problems of system and software engineering (Moscow) pp 289-300 – URL: http://ceur-ws.org/Vol-2514/paper115.pdf

[13] Goncharov N I and Goncharov I V 2020 Mathematical modeling of the security management process of an information system in conditions of unauthor-ized external influences International conference on control systems, mathematical modelling, automation and energy efficiency (Lipetsk) pp 77-82 – DOI: 10.1109/SUMMA48161.2019.8947513

[14] Kobzistyy S Y, Novoseltsev V I and Smolyakova S D 2020 Algorithm and method for recognizing critical situations using semantic networks on critical information infrastructure facilities as a result of cyber attacks IEEE Conference of russian young researchers in electrical and electronic engineering (St. Petersburg and Moscow) pp 2066-2071 – DOI: 10.1109/EIConRus49466.2020.9039255

[15] Noev A N and Sumin V I 2020 Mathematical model for managing the dynamics of the development of information conflict in information systems International conference on control systems, mathematical modelling, automation and energy efficiency (Lipetsk) pp 88-93 – DOI: 10.1109/SUMMA48161.2019.8947546

[16] Dobryshin M M, Kochedykov S S, Novoselcev V I and Grechishnikov E V 2019 Algorithmic model of functioning of the system to detect and counter cyber attacks on virtual private network Journal of Physics: Conference Series 1203 012064 – DOI:10.1088/1742-6596/1203/1/012064

[17] Goncharov N, Goncharov I and Dushkin A 2019 Assessment of the conflict stability of information systems using mathematical modeling International Multi-Conference on Industrial Engineering and Modern Technologies (Vladivostok) pp 1-5 – DOI: 10.1109/FarEastCon.2019.8934011

[18] Parinov P, Goncharov N, Goncharov I and Dushkin A 2019 Simulation and evaluation of conflict interactions in information systems Journal of Physics: Conference Series 1203 012063 – DOI: 10.1088/1742-6596/1203/1/012063

[19] Parinov P, Goncharov N, Goncharov I, Dushkin A and Maximova M 2019 Modeling of information processes for modern information system security assessment IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (Moscow) pp 1758-1763 – DOI: 10.1109/EIConRus.2019.8656828

[20] Kochedykov S, Orlova D, Grechishnikov E and Dushkin A 2019 The mathematical model of cyber attacks on the critical information system Journal of Physics: Conference Series 1202 – DOI: 10.1088/1742-6596/1202/1/012013

[21] Markin P, Kochedykov S and Dushkin A 2019 Express Assessment Method for the Risk of Impaired Functional Stability Information and Communication System in Conditions Cyber Attacks IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (Moscow) pp 1754-1757 – DOI: 10.1109/EIConRus.2019.8657323