Algorithm for Determining Saddle Point in Game Theory Problem of Choosing Software for Information Security on Computer Network Servers

Aleksandr Yu. Bykov[1], Maksim V. Grishunin[1], Evgenij G. Fedorov[1] and Irina A. Markova[1]

[1] *Bauman Moscow State Technical University, 5/1 2nd Baymanskay ul., Moscow, 105005, Russia*

Abstract

The game formulation of the problem of two players is presented: a defender and an attacker. The defender selects the information security software for the computer network servers, taking into account the importance of the processed information, and the attacker selects the possible types of attacks. The mathematical formulation of the problem is a discrete-continuous game, the set of solutions of the defender is discrete, and the set of solutions of the attacker is continuous. The game is a zero-sum game, with the defender's damage used as a quality indicator. The defender, given the attacker's solution, solves the Boolean programming problem, and the attacker, given the defender's solution, solves the linear programming problem. An algorithm for finding the saddle point in a mixed strategy for the defender, in a pure strategy for the attacker, based on reducing the continuous problem of the attacker to a discrete one, is proposed. The algorithm is based on the ideas of the Brown-Robinson method, but without explicit construction of the game matrix. To reduce the computational resources of the processor required for the operation of the algorithm, it is proposed that after a given number of steps a new solution is not obtained, not to solve optimization problems, but to search for solutions by brute force among the previously found solutions. An example of solving the problem is given.

Keywords

Information security, discrete-continuous game, zero-sum game, boolean programming, linear programming, mixed strategy, pure strategy

1. Introduction

Mathematical models originating from game theory frequently arise when solving different problems in information security field. The most common model class has two players: defender and attacker, but there can be more players. Let's look at some examples.

In [1] dynamic game theory is used to protect the privacy of medical data in internet of things.

In [2] attack and defense are viewed as a dynamic process in real time, a differential game theory attack-defense model for choosing a strategy of network defense is used.

In [3] dynamic games are used to discover Advanced persistent threats (APTs). Two players are making subjective decisions about choosing scanning interval and attack interval.

In [4] a multi-stage dynamic game of defender and attacker for resource-heavy discovery of Advanced persistent threats (APTs). The game evolves on an information flow graph, whose nodes are processes and objects (e.g., file, network endpoints) in the system and the edges capture the interaction between different processes and objects.

In [5] a model for deceiving a potential attacker is considered. The defender is using differential mechanisms for scrambling system configurations. The attacker is using Bayesian approach for outputting real system configurations.

BIT-2021: XI International Scientific and Technical Conference on Secure Information Technologies, April 6-7, 2021, Moscow, Russia
EMAIL: abykov@bmstu.ru (A. 1); grishunin-mv@ya.ru (A. 2); fedorov.evg.msu@gmail.com (A. 3); gurina.irina.94@gmail.com (A. 4)
ORCID: 0000-0001-6336-5603 (A. 1); 0000-0001-9544-017X (A. 2); 0000-0003-2261-9691 (A. 3); 0000-0002-2297-5840 (A. 4)

In [6], an equilibrium is sought in a two-level hierarchical game with non-opposite interests of many players. A problem of protecting internet of things users from risks of breaching information security is studied. Top level contains a centralized defender, while defenders with rights of ordinary users are placed on the bottom level. Both centralized and decentralized models are studied.

In [7] a problem of protecting cyber-physical power systems from exploitation risks is studied. Time delay of system recovery and distributed denial of service are considered as risks. A three stage defender-attacker-defender model of mathematical programming based on dynamic game theory with complete information is proposed.

In [8] a problem of data transmission in noisy channels is studied. A trade-off between transmission speed and reliability is sought. Stochastic-mean-field-game is considered. Message codes play the role of agents.

In [9] a game between a system administrator and an attacker is studied. This is an asymmetric information game with zero sum between two players that are unequally informed about the game: informed defender and uninformed attacker.

In [10] a mobile ad-hoc network in which network nodes are in motion is studied. Evolutionary games for stimulating cooperation between mobile modes are used.

In [11] a perspective theory describing human behavior during risk-involved decision making is used for discovering Advanced Persistent Threats. The game uses Nash equilibrium criterion.

In [12] Nash equilibrium is searched in order to ensure performance of blockchain system.

In [13] a fast defense system against DDoS and port scanning attacks in software-defined networking is studied. Three approaches to attack discovery are compared: particle swarm optimization, multi-layer neural network and discrete wavelet transform.

In most cases considered models can be either discrete or continuous. Combined discrete-continuous models can also be used.

In [14] a coalition-free game with conflicting interests in the data transmission network with possible loss of communication is studied. Data source and network nodes are considered as players. In order to reduce transmission delays, hosts are stimulated by the source to cache files. A search for equilibrium Nash states is carried out, both for pure/mixed strategies and for discrete/continuous sets of strategies.

In [15] a game between intruder and subjects responsible for security, who exchange data on the network, is studied. Data confidentiality breach is possible during this exchange. The gains from cooperation and loss of confidentiality during information exchange are measured. Both discrete and continuous games are considered.

Some other mathematical models that are applicable to information security problems are studied in [16-25].

A discrete-continuous game of choosing which software to install on network servers for information security is studied below. It takes into consideration the importance of data stored and processed on these servers.

The problem under consideration is relevant, since there are many different software tools for protecting information that have different protection efficiency and require different resources. Because of this, the problem of choosing these tools arises, and defining this problem in terms of game theory allows us to take into account possible actions of the attacking side.

The goal of this paper is to provide an information security specialist with mathematical software, that would take into account the importance degree of processed information and actions of the attacking side, for justifying the choice of information security software.

The main tasks of the paper are: formulating mathematical game theory problem of choosing software; proposing algorithms for solving it, carrying out computational experiments; deriving conclusion.

2. Problem Description

Let us consider a distributed computer system comprising many servers and providing different services on the network. Servers run some target tasks for which they are intended. Required software

including OS and applications is installed on the servers. There exists a minimal configuration of application and system programs enabling each server to run its target tasks. These programs are implemented as processes. Minimal configuration requires availability of certain computing resources on server (cpu time, memory, disk space, networks).

Besides the main processes (required to run target tasks) maintenance processes are being launched. These processes can be for example security enforcing processes such as antivirus software, DDoS prevention tools, data integrity control programs, privacy programs, programs for protection against unauthorized network connections, etc. Auxiliary or maintenance processes also have some CPU time and memory requirements and cannot be all run at the same time due to lack of resources, however usually there is no reason to run them all. The problem arises of choosing running processes and installed programs.

Server information resources are subject to attacks, which implement threats with different possible goals. Several types of attacks can be distinguished, for example DDoS attacks, confidential information access attacks, data distortion attacks (integrity violation), etc. Estimates of potential damage for server owner in case of attack success exist for each server depending on importance of data stored and processed on a server. Using auxiliary processes can prevent damage to servers from incoming attacks completely, or can partially mitigate their consequences (possible damage).

Let us consider the problem of choosing auxiliary processes from the standpoint of game theory. There is a defender, whose solution to the problem consists choosing which auxiliary processes to run on each server, and an attacker, whose solution consists of choosing attack types. Possible limitations on attacker resources can also be considered, if such information is available. Let's consider a game of two players, we will regard defender damage estimate as an indicator of choice for both players, therefore the game is a zero sum game.

The problem can be summarized as follows.

Given:

A set of servers running target tasks.

A set of auxiliary processes for enforcing information security.

A set of possible attack types that can be carried out on systems' information resources.

A damage estimate in case of attack success is defined for each server and each attack type depending on importance of data stored and processed on a server.

The following computational resource requirements are defined for each server and each auxiliary process:

- CPU load ratio that depends on the servers' computational power
- Memory usage
- Other resources are allowed

A "process cost" is defined in case a corresponding program has to be purchased.

For each server its memory size and computational power, which can be allocated to auxiliary processes (while taking into account resources needed to run main processes), as well as total defender funds for buying software are defined.

For each server and type of attack a damage estimate in case of attack success is defined.

For each auxiliary process and attack type the probability or possibilities of prevention (in terms of fuzzy sets) of an attack or a possible damage mitigation value if the process is running are defined.

For each attach type financial burden on attacker for carrying it out are defined. Other attacker resource (for example computational resources) costs estimates can be considered if such information is available.

Limit on attackers' financial resources is defined (limit on other types of resources can be considered if such information is available).

Required:

In order to minimize possible damage for defender – determine which auxiliary processes should be run on each server, considering possible attacker actions, while also meeting requirements on servers' computational power, memory and possibly other server resources, and on defenders' financial resources needed for auxiliary software purchase.

3. Mathematical Formulation of the Problem

3.1. Initial data

1. $S = \{s_1, s_2, \dots, s_n\}$ – set of servers on a distributed computer system, $N = \{1,2, \dots, n\}$ - set of indices enumerating these servers.
2. $A = \{a_1, a_2, \dots, a_m\}$ – set of attack types (implementations of security threats) on the system, $M = \{1,2, \dots, m\}$ - set of indices enumerating these attack types.
3. $\Pi = \{\pi_1, \pi_2, \dots, \pi_\lambda\}$ – set of auxiliary processes, which can be run on servers for security enforcement, $\Lambda = \{1,2, \dots, \lambda\}$ – set of indices enumerating these processes.
4. $R^{(d)} = \left\{r_1^{(d)}, r_2^{(d)}, \dots, r_{l^{(d)}}^{(d)}\right\}$ – set of limited defender resources, $L^{(d)} = \{1,2, \dots, l^{(d)}\}$ - set of indices enumerating these resources.

Resource set R is divided into two disjoint subsets $R^{(d)} = R^{(pr)} \cup R^{(sh)}$, $R^{(pr)}$ being a set of private server resources (each server has its own resources), and $R^{(sh)}$ being a set of shared server resources that are distributed among all servers. We introduce separate enumeration for private and shared resources with two sets of resource indices $L^{(pr)} = \{1,2, \dots, l^{(pr)}\}$ and $L^{(sh)} = \{1,2, \dots, l^{(sh)}\}$ respectively.

5. $R^{(a)} = \{r_1^{(a)}, r_2^{(a)}, \dots, r_{l^{(a)}}^{(a)}\}$ – set of limited attacker resources. $L^{(a)} = \{1,2, \dots, l^{(a)}\}$ - set of indices enumerating these resources.

3.2. Set elements parameters and relations between them

1. $w_{ij} \geq 0, \forall i \in N, j \in M$ – estimate of damage to defender on i-th server if attacker was successful in carrying out j-th type of attack.
2. $p_{jk} \in [0,1], \forall j \in M, k \in \Lambda$ – probability (or possibility) of preventing j-th attack (threat implementetion) if k-th process is running on any server or defender damage mitigation value.
3. $v_{jik}^{(pr)} \geq 0, \forall j \in L^{(pr)}, k \in \Lambda, i \in N$ – value of j-th private resource (performance or memory) required to run k-th process on i-th server.
4. $V_{ji}^{(pr)} \geq 0, \forall j \in L^{(pr)}, i \in N$ – maximum value of j-th private resource on i-th server, that can be spent on auxiliary processes, taking into account the ammount of the resource required to run main processes running target tasks.
5. $v_{jik}^{(sh)} \geq 0, \forall j \in L^{(sh)}, k \in \Lambda, i \in N$ – value of j-th shared resource (for example, financial) required to run k-th process on i-th server.
6. $V_j^{(sh)} \geq 0, \forall j \in L^{(sh)}$ – maximum value of j-th shared resource for all servers, that can be used for auxiliary processes.
7. We will combine some computer processes (applications) into homogeneous groups, for each group it makes sense to install only one of applications belonging to it. For example, There can be several antivirus applications, etc. $G = \{1,2, \dots, g\}$ – indices of these groups. For their definition let's introduce a boolean matrix $B_{<\lambda\, x\, g>} \|b_{ki}\|, \forall\, k \in \Lambda,\ i \in G,\ b_{ki} = 1$ if k-th application belongs to i-th group and $b_{ki} = 0$ if otherwise.
8. $v_{kij}^{(a)} \geq 0, \forall k \in L^{(a)}, j \in M, i \in N$ - value of k-th attacker resource required for j-th attack type on i-th server.
9. $V_k^{(a)} \geq 0, \forall k \in L^{(a)}$ - maximum quantity of k-th attacker resource, that he can use for all attack types (only financial resource can be considered in the simplest case).

3.3. Parameters

Let's introduce a boolean value $x_{ki} \in \{0,1\}, \forall k \in \Lambda, i \in N$, for the defender, $x_{ki} = 1$ if k-th process is running on i-th server and $x_{ki} = 0$ otherwise. These variables form the vector $\vec{X}$.

Let's introduce a value $y_{ij} \in [0,1], \forall i \in N, j \in M$ for the attacker. Its value can be interpreted as a probability or a possibility (in terms of fuzzy sets) of carrying out j-th attack type on i-th server. These variables form the vector $\vec{Y}$.

3.4. Players indicators

During damage calculation we will assume that probability (safety degree in case of fuzzy description) of defending i-th server from j-th attack type while using several auxiliary processes is determined by a process with maximum probability: $P_{ij}(\vec{X}) = \max\limits_{k \in \Lambda}\{p_{jk}x_{ki}\}, \forall j \in M, i \in N$. Therefore, an estimate of mitigated damage on all servers is $U_{\text{пр}}(\vec{X}, \vec{Y}) = \sum_{i \in N}\sum_{j \in M} w_{ij}P_{ij}(\vec{X})y_{ij} = \sum_{i \in N}\sum_{j \in M} w_{ij}\max\limits_{k \in \Lambda}\{p_{jk}x_{ki}\}y_{ij}$.

The estimate of the full damage to the defender is given by a difference of estimate of maximum damage (without any defenses) and mitigated damage:

$$U(\vec{X}, \vec{Y}) = \sum_{i \in N}\sum_{j \in M} w_{ij}y_{ij} - \sum_{i \in N}\sum_{j \in M} w_{ij}\max_{k \in \Lambda}\{p_{jk}x_{ki}\}y_{ij} = \sum_{i \in N}\sum_{j \in M} w_{ij}y_{ij}\left(1 - \max_{k \in \Lambda}\{p_{jk}x_{ki}\}\right). \quad (1)$$

Defenders' goal is to minimize this indicator, attackers' – to maximize.

3.5. Limitations

Conditions for defender

Conditions on private resource usage on each server:

$$\sum_{k \in \Lambda} v_{jik}^{(pr)}x_{ki} \le V_{ji}^{(pr)}, \forall j \in L^{(pr)}, i \in N. \quad (2)$$

Condition on shared resource usage for all servers:

$$\sum_{i \in N}\sum_{k \in \Lambda} v_{jik}^{(sh)}x_{ki} \le V_{j}^{(sh)}, \forall j \in L^{(sh)}. \quad (3)$$

Condition limiting number of applications for each group to one on any server:

$$\sum_{k \in \Lambda} b_{kj}x_{ki} \le 1, \forall i \in N, j \in G. \quad (4)$$

Conditions for attacker

Condition on usage of attackers resources required for carrying out attacks:

$$\sum_{i \in N}\sum_{j \in M} v_{kij}^{(a)}y_{ij} \le V_{k}^{(a)}, \forall k \in L^{(a)}. \quad (5)$$

Presented formulation of the problem with indicator given by (1) and conditions for defender (2)-(4) and attacker (5) is a mixed discrete-continuous zero sum game. Defender is solving a boolean programming problem with indicator (1) being non-linear for $\vec{X}$ and conditions (2)-(4) assuming fixed attackers' solution $\vec{Y}$, while attacker is solving a linear programming problem with indicator (1) (non-linear for $\vec{Y}$) and conditions (5) assuming fixed $\vec{X}$.

4. Saddle Point Search Algorithm

For a game of two players with zero sum equilibrium state is determined by games' saddle point. For a continuous game of two players, in the case where possible player choice sets are convex and target function is linear, saddle point always exists for player indicator function [26]. In the discrete case, if player choice sets are finite, the game can be reduced to a matrix game [27]. For a matrix

game a saddle point can both be present or not present for pure strategies, but always exists for mixed strategies (a pure strategy is a special case of a mixed strategy).

4.1. Justification for saddle point existence in a discrete-continuous problem

The defenders' choices set is finite, while attackers' choices set is continuous. Analogous to [27], because attacker is solving a linear programming problem and this games' solution lies on a vertex of a polyhedron of possible choices with number of such vertices being finite, a continuous problem can be reduced to discrete one, if one considers polyhedron vertices as solutions. In this case the problem can be reduced to a matrix game in which a saddle point must always exist for mixed strategies.

4.2. Description of saddle point search algorithm using ideas from Brown-Robinson method

The idea of Brown-Robinson method is based on each player solving consecutive problems, and using total accumulated win or loss as the indicator. For multiple solutions of these problems a probability of each solution is estimated, with combination of those estimates determining a mixed strategy. Analogous to [26], the attacker with continuous solution set should calculate average $\vec{Y}$ among set of obtained solutions. For the defender we get probability estimates of obtaining each solution.

Let $\vec{Y}^{(1)}, \vec{Y}^{(2)}, \dots, \vec{Y}^{(g)}$ be a sequence of solutions that the attacker obtains while optimizing indicator (1), then the defender on $(g+1)$-th step must solve the problem of minimizing indicator:

$$U_{\text{з}}^{(g+1)}(\vec{X}) = \sum_{e=1}^{g} U(\vec{X}, \vec{Y}^{(e)}) = \sum_{e=1}^{g} \sum_{i \in N} \sum_{j \in M} w_{ij} y_{ij}^{(e)} \left(1 - \max_{k \in \Lambda} \{p_{jk} x_{ki}\}\right), \tag{6}$$

where $y_{ij}^{(e)}$ are components of vector $\vec{Y}^{(e)}$

Let $\vec{X}^{(1)}, \vec{X}^{(2)}, \dots, \vec{X}^{(g)}$ be a sequence of solutions that the defender obtains while optimizing indicator (1), then the attacker on $(g+1)$-th step must solve the problem of maximizing indicator:

$$U_{\text{н}}^{(g+1)}(\vec{Y}) = \sum_{e=1}^{g} U(\vec{X}^{(e)}, \vec{Y}) = \sum_{e=1}^{g} \sum_{i \in N} \sum_{j \in M} w_{ij} y_{ij} \left(1 - \max_{k \in \Lambda} \{p_{jk} x_{ki}^{(e)}\}\right), \tag{7}$$

where $x_{ki}^{(e)}$ are components of vector $\vec{X}^{(e)}$.

The problem with indicator (6) is a boolean programming problem for the defender. As indicator (6) is linear on components of $\vec{Y}^{(e)}$, therefore when optimizing we can use a middle point among vectors $\vec{Y}^{(0)}, \vec{Y}^{(1)}, \dots, \vec{Y}^{(g)}$ instead of their first sum. The problem with indicator (7) is a linear programming problem for the attacker.

Let's introduce a definition for the middle point, obtained on g initial algorithm steps. We denote solutions obtained on each of g initial algorithm steps for defender as $\vec{Y}^{(1)}, \vec{Y}^{(2)}, \dots, \vec{Y}^{(g)}$, then the middle point for attacker can be written as:

$$\vec{Y}^{(mid)} = \frac{1}{g} \sum_{e=1}^{g} \vec{Y}^{(e)}, \tag{8}$$

4.3. Let's Formulate the Saddle Point Search Algorithm

Step 0. We assume $\vec{Y}^{(sum)} = \|0,0,\dots,0\|^T$ (total vector for calculation sum $\sum_{e=1}^{g} \vec{Y}^{(e)}$), $\vec{X}^{(0)} = \|0,0,\dots,0\|^T$ - initial solution of the defender, any solution can be chosen, vector map $\vec{X}$ is empty. This map has vector $\vec{X}^{(k)}$ as key, and number of $\vec{X}$ occurrences in different steps of the algorithm as values.

Step k (k=1, 2, 3, …). We determine $\vec{Y}^{(k)}$ with $\vec{X}$ vector map obtained on current step, by searching linear programming problem for the attacker, maximize indicator (7) (for k=1 we maximize indicator (1) for $\vec{X} = \vec{X}^{(0)}$). Attackers' quality indicator value we denote as $U_{\text{н}}^{(k)}(\vec{Y}^{(k)})$, we assume $\vec{Y}^{(sum)} = \vec{Y}^{(sum)} + \vec{Y}^{(k)}$, $\vec{Y}^{(mid)} = \frac{\vec{Y}^{(sum)}}{k}$.

We determine $\vec{X}^{(k)}$ for fixed $\vec{Y}^{(mid)}$ by solving a boolean programming problem for the defender with indicator (6). Defenders' quality indicator value we denote as $U_{\text{з}}^{(k+1)}(\vec{X}^{(k)})$, obtained $\vec{X}^{(k)}$ we include into defender's solution map.

From second step onward we check the stopping criterion. If it is met, we stop the algorithm, obtained attackers' solution - $\vec{Y}^{(mid)}$, using obtained map for $\vec{X}$ vectors we calculate probability estimates for each vector in map, this will be the defenders' solution in mixed strategies. Pair of defenders' and attackers' solutions comprises an approximated saddle point. If stopping criterion is not met, we continue with the next step.

The condition $|\frac{U_{\text{н}}^{(k)}(\vec{Y}^{(k)})}{k-1} - \frac{U_{\text{з}}^{(k+1)}(\vec{X}^{(k)})}{k}| < \varepsilon$ can be used as a stopping criterion, where ε - some relatively small positive value, determining algorithm error, ε can be chosen as a percentage of the value $min\left\{\frac{U_{\text{н}}^{(k)}(\vec{Y}^{(k)})}{k-1}, \frac{U_{\text{з}}^{(k+1)}(\vec{X}^{(k)})}{k}\right\}$.

If the need arises, the algorithm can be reformulated thus, that on each step the optimization problem for the defender would be solved first, and for attacker – second.

4.4. Ways to reduce the required computing resources for algorithm and increase its precision

The Described algorithm requires substantial computational resources starting from certain dimension of the problem, because on each step the defender has to solve boolean programming problem with exponential computational complexity.

As shown by computational experiments, obtained solutions start to repeat on certain step, i.e. for a given relatively large step count no new solutions were obtained for both attacker and defender (attackers' solutions can also be saved in a map), similar approach was used in [27]. In this case we can stop solving optimization tasks, and start looking for solutions among obtained ones using brute force.

If there aren't too many obtained solutions in maps, we can compose a usual game matrix from them by filling it with indicator values for each solution pair for defender and attacker, and then solve this problem precisely or approximately by using this matrix.

5. Example

Table 1 shows the initial data obtained by the pseudorandom number generator. In the example, the number of objects is 5, the number of applications is 9, the number of threats is 3, for example, integrity, availability, and privacy violations.

Table 1

Source data for the task obtained by the pseudorandom number generator

Objects \ Threats	Threat damage		
	Threat 1	Threat 2	Threat 3
Object 1	4580.43	3197.33	2444.14
Object 2	1205.57	4030.37	2383.47
Object 3	4378.77	4617.79	3061.22
Object 4	2747.12	3145.94	4061.98
Object 5	3529.50	2822.08	4833.49

Threats \ Applications	Applications (threat prevention probabilities)								
	1	2	3	4	5	6	7	8	9
Threat 1	0.79	0.96	0.73	0.77	0.71	0.91	0.71	0.83	0.91
Threat 2	0.92	0.77	0.97	0.87	0.73	0.90	0.98	0.71	0.80
Threat 3	0.73	0.72	0.81	0.71	0.90	0.92	0.83	0.73	0.76

Groups \ Applications	Application groups
	Application numbers
Group 1	1, 4, 7
Group 2	2, 5, 8
Group 3	3, 6, 9

Objects \ Applications	Applications (defender shared resource consumption)								
	1	2	3	4	5	6	7	8	9
Object 1	10.55	29.69	23.07	38.00	25.58	13.02	38.19	26.11	14.26
Object 2	27.37	15.60	27.56	23.66	33.65	26.19	28.81	36.90	34.97
Object 3	23.34	21.67	22.60	33.16	18.23	22.43	42.60	10.81	18.94
Object 4	39.41	34.07	49.80	39.66	30.45	22.33	37.83	26.25	31.22
Object 5	20.69	34.03	44.32	15.93	13.37	24.97	16.15	31.53	38.03
Total resource	250.53								

Defender's private resource (one resource)

Objects \ Applica-tions	Applications (private resource consumption for objects)									Total stock
	1	2	3	4	5	6	7	8	9	
Object 1	38.88	18.48	41.78	27.36	18.82	35.81	42.72	12.31	20.72	51.66
Object 2	39.64	43.36	11.90	23.73	46.27	33.48	40.73	12.30	37.69	62.34
Object 3	41.08	32.75	30.57	43.33	32.18	26.99	10.35	21.28	12.19	53.33
Object 4	28.28	13.78	15.12	20.27	29.39	32.05	42.84	47.70	20.65	61.30
Object 5	21.40	33.88	11.78	33.52	37.61	30.60	16.25	44.09	30.26	54.10

Objects \ Threats	Threats, resource consumption of the attacker for the implementation of threats		
	1	2	3
Object 1	13.67	38.20	18.03
Object 2	49.22	30.85	38.59
Object 3	42.70	32.79	22.24
Object 4	13.90	21.28	44.87
Object 5	38.48	39.61	24.97
Total resource	234.70		

The results of solving the problem are presented in Table 2. The table shows the saddle point: one solution for the attacker (the middle point found by the algorithm) and three solutions for the defender with estimates of their probabilities (a mixed strategy). The estimate of the indicator value is 2719.71.

Table 2
Results of solving the task

Intruder's solution		1.00 0.39 1.00 0.00 0.00 0.00 1.00 0.00 1.00 0.47 1.00 0.71 1.00 0.00 1.00	
Defender's solutions	0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 1 0 1 1 1 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 1 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
Probabilities of solutions	0.10	0.80	0.10

6. Conclusion

The report considers the mathematical formulation of the problem of choosing software protection tools for a server system. The problem is a discrete-continuous zero-sum game of two players: a defender and an intruder. The set of decisions of the defender is discrete and finite, the set of decisions of the attacker is continuous, including the uncertainty of the actions of the attacker from the point of view of the defender. The defender must solve the Boolean programming problem, and the intruder must solve the linear programming problem. An algorithm for finding a saddle point for a defender in mixed strategies, for an intruder in pure strategies is proposed. The algorithm is based on reducing the continuous problem for the attacker to a discrete one and is based on the ideas of the Brown-Robinson method. Computational experiments were carried out that demonstrated the efficiency of the algorithm, and a method was proposed to reduce the computational complexity of the algorithm by switching from solving optimization problems to finding solutions among previously found ones, after the solutions are repeated at a given number of steps.

The use of the proposed algorithm allows using software protection tools in the case of restrictions on various resources to significantly reduce the damage to the protection side from various types of attacks.

The reliability of the obtained results is confirmed by the correctness of the mathematical formulation of the problem, an explicit meaningful interpretation of both the problem statement and the obtained solutions, as well as by experimental verification of the obtained solutions to meet the required criteria.

7. References

[1] Fanyu Kong, Yufeng Zhou, Bin Xia, Li Pan, Limin Zhu. A Security Reputation Model for IoT Health Data Using S-AlexNet and Dynamic Game Theory in Cloud Computing Environment. IEEE Access, 2019, vol. 7, pp. 161822-161830. DOI: 10.1109/ACCESS.2019.2950731.

[2] Hengwei Zhang, Lv Jiang, Shirui Huang, Jindong Wang, Yuchen Zhang. Attack-Defense Differential Game Model for Network Defense Strategy Selection. IEEE Access, 2018, vol. 7, pp. 50618-50629. DOI: 10.1109/ACCESS.2018.2880214.

[3] Liang Xiao, Dongjin Xu, Narayan B. Mandayam, H. Vincent Poor. Attacker-Centric View of a Detection Game against Advanced Persistent Threats. IEEE Transactions on Mobile Computing, 2018, vol. 17, iss. 11, pp. 2512-2523. DOI: 10.1109/TMC.2018.2814052.

[4] Shana Moothedath, Dinuka Sahabandu, Joey Allen, Andrew Clark, Linda Bushnell, Wenke Lee, Radha Poovendran. A Game-Theoretic Approach for Dynamic Information Flow Tracking to Detect Multistage Advanced Persistent Threats. IEEE Transactions on Automatic Control, 2020, vol. 65, iss. 12, pp. 5248-5263. DOI: 10.1109/TAC.2020.2976040.

[5] Dayong Ye, Tianqing Zhu, Sheng Shen, Wanlei Zhou. A Differentially Private Game Theoretic Approach for Deceiving Cyber Adversaries. IEEE Transactions on Information Forensics and Security, 2020, vol. 16, pp. 569-584. DOI: 10.1109/TIFS.2020.3016842.

[6] Rui Zhang, Quanyan Zhu. FlipIn: A Game-Theoretic Cyber Insurance Framework for Incentive-Compatible Cyber Risk Management of Internet of Things. IEEE Transactions on Information Forensics and Security, 2019, vol. 15, pp. 2026-2041. DOI: 10.1109/TIFS.2019.2955891

[7] Boyu Gao;Libao Shi. Modeling an Attack-Mitigation Dynamic Game-Theoretic Scheme for Security Vulnerability Analysis in a Cyber-Physical Power System. IEEE Access, 2020, vol. 8, pp. 30322-30331. DOI: 10.1109/ACCESS.2020.2973030

[8] Makan Zamanipour. Fast-and-Secure State-Estimation in Dynamic-Control Over Communication Channels: A Game-Theoretical Viewpoint. IEEE Transactions on Signal and Information Processing over Networks, 2020, vol. 6, pp. 645-655. DOI: 10.1109/TSIPN.2020.3018327

[9] Lichun Li, Jeff S. Shamma. Efficient Strategy Computation in Zero-Sum Asymmetric Information Repeated Games. IEEE Transactions on Automatic Control, 2020, vol. 65, iss. 7, pp. 2785-2800. DOI: 10.1109/TAC.2019.2933396

[10] Burhan Ul Islam Khan, Farhat Anwar, Rashidah Funke Olanrewaju, Bisma Rasool Pampori, Roohie Naaz Mir. A Game Theory-Based Strategic Approach to Ensure Reliable Data Transmission With Optimized Network Operations in Futuristic Mobile Adhoc Networks. IEEE Access, 2020, vol. 8, pp. 124097-124109. DOI: 10.1109/ACCESS.2020.3006043

[11] Liang Xiao, Dongjin Xu, Narayan B. Mandayam, H. Vincent Poor. Attacker-Centric View of a Detection Game against Advanced Persistent Threats. IEEE Transactions on Mobile Computing. 2018. Vol. 17, Iss. 11. P. 2512-2523. DOI: 10.1109/TMC.2018.2814052

[12] Jiaxing Qi, Jing Yu, Shunfu Jin. Nash Equilibrium and Social Optimization of Transactions in Blockchain System Based on Discrete-Time Queue. IEEE Access, 2020, vol. 8, pp. 73614-73622. DOI: 10.1109/ACCESS.2020.2986084

[13] Marcos V.O. De Assis, Matheus P. Novaes, Cinara B. Zerbini, Luiz F. Carvalho, Taufik Abrãao, Mario L. Proença. Fast Defense System Against Attacks in Software Defined Networks. IEEE Access, 2020, vol. 6, pp. 69620-69639. DOI: 10.1109/ACCESS.2018.2878576

[14] Sidi Ahmed Ezzahidi, Essaid Sabir, Mohamed El Kamili, El-Houssine Bouyakhf. A non-cooperative file caching for delay tolerant networks: A reward-based incentive mechanism. 2016 IEEE Wireless Communications and Networking Conference, 2016. DOI: 10.1109/WCNC.2016.7565161

[15] Richeng Jin; Xiaofan He;Huaiyu Dai. On the Security-Privacy Tradeoff in Collaborative Security: A Quantitative Information Flow Game Perspective. IEEE Transactions on Information Forensics and Security, 2019, vol. 14, iss. 12, pp. 3273-3286. DOI: 10.1109/TIFS.2019.2914358

[16] Klyucharev P.G. Grafy Pajzera v zadachah kriptografii i obrabotki informacii [pizer graphs in cryptography and in information processing]. Sbornik trudov desyatoj mezhdunarodnoj

nauchno-tekhnicheskoj konferencii «Bezopasnye informacionnye tekhnologii» (BIT-2019), Moscow, MGTU im. N.E.Baumana, 2019, pp. 176-179. (In Russ).
[17] Ponomarenko G.S., Klyucharyov P.G. Opredelenie obfuskacii JavaScript-programm s pomoshch'yu raskrasok na abstraktnyh sintaksicheskih derev'yah [detection of obfuscated javascript code based on abstract syntax trees coloring]. Matematika i matematicheskoe modelirovanie, 2020, no. 2, pp. 1-24. DOI: 10.24108/mathm.0220.0000218.
[18] Vishnevskij A.S., Klyucharev P.G. Obnaruzhenie celenapravlennyh atak veb-orientirovannoj obmannoj sistemoj, osnovannoj na algoritme antiklassifikacii [Anticlassification algorithm for targeted computer attack detection in web-oriented honeypot]. Nejrokomp'yutery: razrabotka, primenenie, 2020, vol. 22, no. 3, pp. 5-17. DOI: 10.18127/j19998554-202003-01.
[19] Andrey Vishnevsky, Petr Klyucharev. The Sound User Interface of Honeypot. CEUR Workshop Proceedings. 2019. Vol-2603. P. 83-87. URL: http://ceur-ws.org/Vol-2603/short18.pdf.
[20] Grigory Ponomarenko, Petr Klyucharev. JavaScript Programs Obfuscation Detection Method that Uses Artificial Neural Network with Attention Mechanism. CEUR Workshop Proceedings. 2019. Vol-2603. P. 100-104. URL: http://ceur-ws.org/Vol-2603/paper21.pdf.
[21] A. Varfolomeev. Strengthening the password authenticated key exchange protocols due to the use of asymmetric execution of cryptosystems. CEUR Workshop Proceedings. 2019. Vol-2603. P. 79-82. http://ceur-ws.org/Vol-2603/short17.pdf.
[22] M. Basarab, T. Buldakova, K. Smolyaninova, M. Sokolov. User Identification Based on the Vein Pattern in Biometric Immobilizer. CEUR Workshop Proceedings. 2019. Vol-2603. P. 1-5. URL: http://ceur-ws.org/Vol-2603/short1.pdf.
[23] A. Chilikov, A. Zhukov, A. Verkhovsky. The Research of Reversible Cellular Automata with a Finite Lattice. CEUR Workshop Proceedings. 2019. Vol-2603. P. 12-15. URL: http://ceur-ws.org/Vol-2603/short3.pdf.
[24] A. Markov, I. Sheremet. Enhancement of Confidence in Software in the Context of International Security. 2019. Vol-2603. P. 88-92. URL: http://ceur-ws.org/Vol-2603/paper19.pdf.
[25] A. Kalashnikov. Example of Using of Game-Theoretic Approach in Problems of Ensuring Cyber Security of Information Systems. Voprosy kiberbezopasnosti, 2014, No 1(2), pp. 49-54.
[26] Bykov A.Yu., Krygin I.A., Grishunin M.V., Markova I.A. Ob odnom algoritme poiska sedlovoj tochki dlya nepreryvnyh linejnyh igr primenitel'no k zadacham zashchity informacii [One saddle point search algorithm for continuous games applied to information security problems]. Vestnik Moskovskogo gosudarstvennogo tekhnicheskogo universiteta im. N.E. Baumana. Seriya: Priborostroenie [Herald of the Bauman MSTU. Ser. Instrument Engineering], 2020, no. 4 (133), pp. 58-74. DOI: 10.18698/0236-3933-2020-4-58-74.
[27] Bykov A.Yu., Grishunin M.V., Krygin I.A. Igrovaya zadacha vybora zashchishchaemyh ob"ektov i issledovanie algoritma poiska sedlovoj tochki na osnove modifikacii metoda Brauna-Robinsona [THE game problem of selection of assets to protect and research of saddle point search algorithm based on Brown-Robinson method modification]. Voprosy kiberbezopasnosti, 2019, no. 2 (30), pp. 2-12. DOI: 10.21681/2311-3456-2019-2-2-12.