Self-Organized Distributed Anomaly Detection System in Computer Systems Based on The Principal Components Method

Bohdan Savenko [a], Antonina Kashtalian[a], Tomas Sochor[b], Andrii Nicheporuk[a]

[a] *Khmelnytskyi National University, Institutska str., 11, Khmelnytskyi, 29016, Ukraine*
[b] *Prigo University, Havirov, Czech Republic*

Abstract

In order to solve the scientific problem of improving the efficiency of anomaly detection in computer systems, the manifestations of which are due to the effects of malicious software and computer attacks, a method and distributed anomaly detection systems were developed based on synthesis of self-organization and centralization principles. The results are the basis for creating distributed systems to detect computer attacks and malware. Also, they can be used to create such a class of intrusion detection system as a honeynet.

As a result, the peculiarities of anomaly manifestation in computer systems under the conditions of malicious software operation and computer attacks in local computer networks are studied and modern methods of anomaly detection, their features and methods of creation and architecture of distributed systems are analyzed. The architecture model of the distributed anomaly detection system in computer systems has been improved, which synthesizes the requirements of distribution, centralization and self-organization, to create distributed systems and their components, which will operate under the leadership of one center distributed between different components and decide independently. the presence of an anomaly. To ensure the integrity of the distributed system, a method was developed to maintain the integrity of the self-organized distributed system to detect anomalies in computer systems, based on which the system could change its architecture without user intervention, and determine strategies for further work. To detect computer attacks and malware, the method of centralized detection of distributed anomalies by the main components search algorithm has been improved to reduce the dimensionality from the moment of receiving and sending data to the decision center of the system. The results were implemented in the appropriate software, with which detection experiments were conducted, which showed improved reliability in the detection of computer attacks and malicious software.

Keywords 1

Distributed systems, self-organized systems, anomaly detection, principal components method

1. Introduction

An important area that requires the development of methods and means of combating malicious activity is the area related to the functioning of computer networks, because they are used in almost all enterprises, organizations and institutions. Problems with their operation caused by malicious software and computer attacks, and even worse, by concealing the presence of an attacker, can lead to financial losses for businesses, organizations and institutions. Investigations of malicious manifestations in corporate and local networks can be conducted using the apparatus of mathematical statistics.

Corporate and local networks of enterprises, organizations and institutions can have a large number of computers and to study the processes that take place in them, including malicious, you need effective

IntelITSIS'2022: 3rd International Workshop on Intelligent Information Technologies and Systems of Information Security, March 23–25, 2022, Khmelnytskyi, Ukraine
EMAIL: savenko_bohdan@ukr.net (B. Savenko); yantonina@ukr.net (A. Kashtalian); tomas.sochor@osu.cz (T. Sochor); andrey.nicheporuk@gmail.com (A. Nicheporuk)
ORCID: 0000-0001-5647-9979 (B. Savenko); 0000-0002-4925-9713 (A. Kashtalian); 0000-0002-1704-1883 (T. Sochor); 0000-0002-7230-9475 (A. Nicheporuk)

methods and appropriate means of processing the received data about events. The effectiveness of combating malicious manifestations is achieved through a comprehensive approach focused on the integration of detection methods and systems in which they are implemented. For attackers, such approaches significantly complicate the achievement of effectiveness. In general, the appearance in computers or computer systems and networks of malicious software or computer attacks, possibly malicious actions of users, in addition to the direct occurrence of technical malfunctions of hardware devices, exhausts many objects that can attract attention by their unusual behavior. Event processing requires a distributed system that collects and processes data and outputs detection results. Given the need to process data quickly and without human intervention, such a distributed system should be self-organizing.

The paper proposes the use of self-organized distributed systems, developed according to the principles of centralization and self-organization, to detect anomalies in computer systems.

2. Analysis of known solutions

Many researchers are paying attention to the study of anomalies in computer systems to detect malware and computer attacks. They have developed many different detection methods [1-9]. In order to develop new or improve known solutions to detect anomalies in the COP, it is necessary to establish the advantages and disadvantages of known developed methods. Identifying unsolved subtasks that affect the achievement of an effective detection result and strategies to address some of the shortcomings of known methods will increase the reliability and improve the efficiency of detection. Consider the known methods of detecting malicious software and computer attacks in the COP, which are based on the establishment of abnormal states. We will consider those of them that are proposed by well-known researchers in this field. [1] presents an in-depth analysis of four main categories of anomaly detection methods, which include classification, statistics, information theory, and clustering. The focus is on research problems with datasets used to detect network intrusions. The results of the study make it possible to link classification, statistical data processing and clustering with the scientific task in terms of developing applied methods for processing input data in order to detect network anomalies. The problem of matching low-dimensional multidimensional objects to high-dimensional data is considered in [2] both from a theoretical and computational point of view. As datasets become more heterogeneous and complex, the spaces used to approximate them need to become more heterogeneous. This paper reflects the results of work with transitions to the changed bases, which is relevant in terms of the task of detecting anomalies and reducing the dimensionality of the data. Also, the computational complexity of such transformations is analyzed to estimate the required computing resources. With the spread of the Internet of Things through wireless sensor networks, a huge amount of sensor data is generated at unprecedented speeds, leading to a very large amount of explicit or implicit information [3]. When analyzing such sensor data, it is especially important to accurately and efficiently detect not only individual abnormal behaviors, but also abnormal events (ie behavioral patterns). However, most previous work has focused only on detecting anomalies, while ignoring the relationship between them. Even in approaches that take into account the correlation between anomalies, most ignore the fact that the anomaly in the state of these sensors changes over time. This paper [3] proposes an uncontrolled method of detecting context anomalies on the Internet of Things using wireless sensor networks, which takes into account both the status of a dynamic anomaly and the correlation between anomalies based on context in their spatial and temporal neighbors. Also, the efficiency of the proposed method in the model of anomaly detection is investigated. The introduction of information about anomalies and abnormal manifestations is important from the point of view of taking into account the dynamics of data acquisition. In [4], the process of detecting unexpected elements or events in data sets that differ from the norm is considered as a search for anomalies. Unlike standard classification problems, anomaly detection is often applied to unlabeled data, taking into account only the internal structure of the data set. This problem is known as uncontrolled anomaly detection and is addressed in many practical applications, such as network intrusion detection, fraud detection, and biology and medicine. This paper presents the results of the evaluation of 19 different anomaly detection algorithms on 10 different datasets from several application domains. The work is important for research on the uncontrolled detection of anomalies.

Detecting and processing anomalies for real-time big data is a challenge. The amount and speed of data in many systems complicates typical algorithms for scaling and saving their characteristics in real time [5]. The prevalence of data combined with the problem that many existing algorithms consider only the content of the data source and not the content. The solutions proposed in the paper determine the context of anomaly detection. It consists of two different steps: content discovery and context discovery. A content detector is used to detect anomalies in real time. The context detector is used to truncate the results of the content detector, detecting those anomalies that are considered both semantic and contextually abnormal. The context detector uses the concept of profiles, which are groups of similarly grouped data points generated by a multidimensional clustering algorithm. The study was evaluated on the basis of experiments conducted for two real data sets of sensors. The results of this work [5] are important in the context of the importance of sensor content processing. In [6], a gradual method of uncontrolled anomaly detection is proposed, which allows to quickly analyze and process big data in real time. Evaluation of the data set during the experiment shows that the method converges with its stand-alone counterpart for infinitely increasing data streams. In [7], the known solutions for detecting anomalies are analyzed, especially for data with large sizes and mixed types, where the detection of anomalous patterns or behaviors is a non-trivial work. As a result, the importance of this work in the study of known approaches and their comparison, which will take into account these results when choosing promising solutions. The paper [8] focuses on the early detection of unexpected observations in the physical infrastructure, which is of great importance to prevent system failure and further losses. However, modern techniques for detecting anomalies in the existing infrastructure monitoring platform mainly depend on the fixed threshold method. The obvious disadvantage of this method is that it usually leads to a high level of error detection. In this study, an approach to the detection of statistical anomalies was introduced to the monitoring of physical infrastructure. The paper considers three important types of anomalies found on the infrastructure monitoring platform, namely naive point anomalies, contextual point anomalies and level shifts. The paper proposes to use a developed method based on the Gaussian model to detect these three anomalies. Whereas the proposed method can effectively detect only naive point anomalies; an improved approach is proposed, which combines the results of statistical tests on the initial and excellent monitoring data. The results of the proposed methods on the real data set are evaluated. The results show that an optimized approach to detecting anomalies has good accuracy and can significantly reduce the rate of incorrect detection. The obtained results give an understanding of the treatment of three types of anomalies. One of the current problems in detecting anomalies is the ability to detect and distinguish between both point and collective anomalies within a sequence of data or time series [9]. Authors in [9] developed a method and tools to provide users with a choice of anomaly detection methods, and, in particular, provides the implementation of the recently proposed family of anomaly detection algorithms. The article [9] describes the implemented methods, as well as their application to the simulated data, as well as real examples of data contained in the package. The division into point and collective anomalies, as well as methods of their detection is important in the development of methodology for detecting anomalies.

Building distributed systems in local computer networks, along with developing new methods or improving known ones to detect anomalies, is important because the efficiency of their operation depends on the efficiency of detecting anomalies and responding to them. In addition, effective integration in the implementation of the anomaly detection method into a distributed system can improve the overall efficiency of anomaly detection and response. Consider the known methods related to the design of distributed systems and optimization strategies to improve their efficiency. For distributed systems, the cost of communication is the most commonly used metric to assess the efficiency of operations in distributed algorithms for messaging environments [10]. The constant assumption is that the cost of calculations in the components is insignificant compared to the cost of communication. However, in many cases, the implementation of operations rely on complex calculations that should not be ignored. Therefore, a more accurate assessment of performance should take into account both computational costs and communication costs. [10] focuses on the efficiency of read and write operations in atomic shared memory read / write emulations in an asynchronous environment that transmits a fault-prone message. The paper develops and proposes a new computable predicate and an algorithm for its calculation for linear time. The results published in [10] give a new meaning to the term velocity, evaluating both the relationship and the efficiency of calculations of each operation and are important for building distributed systems in terms of organizing communication between components and calculations in them. In [11] the efficiency of poorly coordinated but fast-responding distributed data warehouses is analyzed. The relevance of this area is justified by the problems associated with consistency between the development of complex applications and obtaining only weak guarantees of consistency in data warehouses. The coherence compromise aims to achieve both strong coherence and low delays in the general case. In distributed storage systems, the general concept of almost strong consistency in terms of developing fast-reading algorithms has been studied in [11], while guaranteeing probabilistic atomicity with a clearly limited number of records at a time. A common case of this problem is when multiple clients can write data to distributed repositories. An important indicator in this case is the data obsolescence limit and the probability of atomicity violation, decomposing inconsistent readings into read inversion and write inversion patterns. The result of this work is important for the organization of distributed data warehouses and coordination of information in them according to the criterion of relevance. [12] presents a distributed system for managing very large amounts of structured data distributed on many product servers, while providing highly available services without any point of failure. This system can run on infrastructure with hundreds of nodes distributed across different data centers. It does not support a complete relational data model, but provides customers with a simple data model that supports dynamic control over data placement and formatting.

In [13] the created optimal communication protocols in three scenarios are presented, which is important for maintaining the integrity of the distributed system. The problem of complexity in the organization of communication in search of approximate maximum agreement in the multilateral model of messaging is relevant [14]. The problem of maximum agreement is one of the most fundamental combinatorial problems of a graph with various programs, and the authors' work in [14] is devoted to solving the problem of estimating its complexity. The work [15], which plays a central role in distributed computing, is devoted to the problems of identifying network structures and their topologies. In [16] various settings of distributed calculations and corresponding methods for them are investigated. In [17], the computational power of population protocols for some unreliable or weaker interaction models was investigated. This is necessary to organize the maintenance of the integrity of distributed systems. For many years, an elegant, computation-based hierarchy of consensus has been the best explanation for the relative strength of different objects. Because true multiprocessors allow you to apply the various instructions they support to any memory location, the ability to combine instructions supported by different objects rather than looking at a collection of different objects is important. This paper proposes a classification of the relative power of multiprocessor synchronization instruction sets based on the complexity covered by the minimum number of unlimited memory slots required to resolve a seamless consensus using different instruction sets [18]. Studies in [19] are devoted to the study of the complexity of reporting implicit leader elections in synchronous distributed networks with a diameter of two [19]. The results characterize the complexity of reporting the leader's election on the diameter of the graph [19]. In [20] the problems of planning in the data flow model of distributed transaction memory are considered. Objects shared by transactions are moved from one network node to another following network paths. The authors investigated how the transfer of objects in the network affects the completion time of all transactions and the total cost of communication. The authors have developed scheduling algorithms that work almost optimally and are time-efficient for communication. In [21] a communication model is considered, in which in each round each agent extracts information from several randomly selected agents. Thus, the aim is to determine the smallest amount of information found in each interaction (message size), which, however, allows you to efficiently and reliably calculate the main tasks of information dissemination. The developed protocol uses only 3 bits per interaction. [22] proposes a new technique of functionally distributed malware that dynamically distributes its functions among many software components to bypass various security mechanisms, such as whitelisting and detecting antivirus behavior. To evaluate this approach, the authors have implemented a tool that automatically generates such instances of malicious software, and conducted a series of experiments that show the risks. Detect targeted malware with antivirus, IDS, IPS and special detection tools is quite difficult [23]. The authors compare different machine learning methods used to analyze malware, focusing on static analysis. In [24], the authors develop a formal system of passive testing of software systems, where the parties communicate asynchronously. In [25-29], the authors also investigate protocols for distributed systems.

3. Architecture of a self-organized distributed anomaly detection system in computer systems

Detection of anomalies in computer systems caused by malicious software or as a result of computer attacks requires not only effective methods that will establish the presence of malicious anomalies with a high degree of reliability, but equally important is the system in which the implemented methods.

The requirements for the system to be implemented must be set in such a way that in the future, such a system can maintain its performance in the event of malicious software or as a result of computer attacks. If the system in the conditions of malicious manifestations will not be able to maintain its efficiency, then the methods that are embedded in it to detect anomalies will not be applicable. Therefore, the requirements for the system of such purpose should be formed taking into account not only the specifics of application, but also taking into account the operating environment in which, for example, malicious manifestations will take place.

The system of detecting anomalies in computer systems to realize the possibility of attracting information from different computer stations connected to the local network must have a distributed architecture, because in this case it will be able to take advantage of attracting more computing power by combining computing resources of all computer stations in which its components are installed, compared to malicious manifestations that may occur or be carried out in one or more nodes in the network. Distribution in its network architecture in computer stations provides advantages over malicious software or computer attacks on computer systems on the network due to the ability to primarily ensure the functioning of components in network nodes that are not attacked or affected by malicious software , and therefore may be involved in the detection process, even if part of the system is lost due to the removal of its components from a safe state due to loss of control over nodes in the network due to malware or due to a successful computer attack. But the distribution of system components between different computer stations in the system architecture has a drawback, which is primarily related to the time spent on data collected for processing to the decision center or centers, and then return the results in the form of a decision on further actions of system components. Building an appropriate system architecture that takes into account the balance of advantages and disadvantages of such an architecture and the impact on its components of malicious software or computer attacks is an urgent scientific task.

The method of organizing the interaction of system components is important for the rapid interaction of system components, as well as the optimization of connections between parts of the system in the process of rapid response to abnormal manifestations. If the events took place within one computer station, then the decision on the presence of abnormal manifestations would be made directly in it according to a certain implemented method in the system, which is located in the computer station and with other nodes in the network has no communication. relating to the detection of anomalies. But in such a host case there is no guarantee that the time of processing events, establishing the fact of abnormal manifestations will be significantly less or less than the time spent on communication and sending messages between components of the distributed system, provided that the effectiveness of methods implemented in it, and interactions between its components may be faster when processing events associated with abnormal manifestations. Thus, the detection of anomalies by individual host systems in individual computer stations compared to the detection of distributed systems at many nodes in the network over time can be different, in particular, both larger and smaller. In addition, the host system in a computer station may not be able to cope with malicious manifestations on its own, and its work on detecting malicious manifestations can be significantly delayed.

We will choose the type of network system from the considered and analyzed one, which will include the need to solve problems of detecting anomalies in the network and in each computer station connected to the network. The choice of this type of anomaly detection system based on research conducted from sources [3-7] will allow to detect anomalies and means of the host part of the system with full functionality and involvement of the network part of the system, which will provide additional computing power and implemented methods. This choice of the direction of the anomaly detection system to the nodes in the network and the network itself will determine its appropriate architecture, a feature of which will be the distribution in the network. The location of such a system will choose the local network. Scaling of the distributed system of detection of anomalies in the corporate network, if necessary, can be carried out within its individual segments of local networks not only independently of each other, but also integrated. The need to localize the location of the distributed system is justified by the fact that information about network nodes is known to the administrator and can be taken into account in the architecture of the distributed system when configuring it during installation. Localization of the location of the distributed system allows to gain an advantage over malicious manifestations in individual computer stations not only by attracting more computing power, but also by deciding on the presence of an anomaly not directly in the attacked computer station, but in a separate center, which significantly increases confidence in the result.

Decision-making by the anomaly detection system should be carried out either in one center or in distributed centers at different levels of the hierarchy. If decision-making will take place only in one center, then all the information should be sent to him, waiting for processing, decision-making and sending it to other components of the system for further action. All this can significantly slow down the system as a whole, if the center accumulates many tasks from different components of the system, and can lead to loss of relevance of the decision, because processes in the network and its individual nodes are fast and therefore require dynamic response. Determining the place of decision-making on issues related to the operation of the system or the results of processing events in the network and its nodes using the methods implemented in the system should be divided according to their importance and attribution to part of the system or the whole system. The best solution in this case would be a decision where the decision-making center would be closest to the part of the system that needs it. In this case, the decision-making center should be distributed between the levels in the system components. To achieve this, it is necessary to build hierarchical levels in the architecture of the system. Although components at different levels may communicate with each other, at each level of the hierarchy there will be decision-making centers that will be able to make decisions only on certain clearly defined issues according to the initial data collected from the system components. But not always clearly defined functions, such as responding to established abnormalities, can only be attributed to the decision-making center, which is in the lower level of the hierarchy. Such responses to abnormalities should be coordinated either from the beginning or after the initial response by the main decision-making center of the whole system. Also, other decision-making centers should also be promptly informed about the establishment of abnormal manifestations in a particular network node. Thus, the correct distribution in the system of decision-making centers will determine its effectiveness and the ability to respond quickly to detected abnormalities.

Self-organization as a characteristic feature of the designed anomaly detection system is necessary, because events in computer systems occur very quickly and the response to them should be such that the result of processing and decision-making was relevant, not late. Although it may be delayed and taken into account, it is often necessary to respond quickly to the effects of malware and computer attacks. If the proposed outcome of event processing to identify anomalies for final decision-making in each network node in which system components are installed relied solely on the system administrator or cybersecurity specialist, then most events would be processed with significant delay. Such involvement in the decision-making of the network system administrator or cybersecurity specialist may be required at the stage of analysis of the log of detected anomalous events, registered by the system and its decisions. But efficiency in decision-making is best placed on the system, so it should be based on the ability to self-organize to determine their next steps. In general, such a characteristic feature of the system as self-organization may include mechanisms not only to determine the next steps, but also mechanisms for dynamic restructuring of its architecture depending on the effects of malicious software, computer attacks and the results of processed events to detect anomalies. Self-organization of the system at the level of mechanisms and functions embedded in the system can be realized as part of the main decision-making center, ie the part of the center that is at the top level of the hierarchy.

Given such characteristic properties and features of the designed system as self-organization and distribution, it is necessary to address the issue of dynamic formation of the system from the available active components for some time. This should be designed accordingly, both in the case of initial formation and in the case of long-term use of the system and changes in its architecture depending on changes in its active components and response to abnormalities.

The specifics of the design system to detect anomalies in computer systems are related to the destructive effects of malicious software and computer attacks, and these destructive effects may affect the operation of the designed system to disable it or its components or distort the transmitted data between system components. The difficulty of detecting computer systems protection by an attacker is, in particular, the lack of information about the means of protection of attacked systems. This allows effective protection and detection of abnormal manifestations of such systems without loss of appropriate functionality to detect anomalies or part of the functionality. Given the design of the system as distributed in the nodes of the network, it becomes important to organize the proper interaction of the components of the designed system based on a new network protocol or improvement of existing, but which would take into account the specifics of tasks and complicated by malware or which will exchange information between system components. The protocol that will regulate the exchange of messages between the components of the designed system should have additional elements to confirm the receipt of the message, take into account the dynamic formation of the system from its components at different times, avoid blocking the system component in case of message delay. That is, the requirements for the protocol of information exchange between system components must be different than those known, for example, IRC. Such a protocol requirement is also required to maintain the integrity of the designed system.

The architecture of the designed system in the process of its operation should be dynamically formed from the mandatory component, in which part of the center of the upper level of the hierarchy, and part of the system components, not necessarily all components. It is not necessary that all components of the designed system in the process of its operation are available and active. Some of them may be in disconnected computer stations or during the operation of the system, some of the computer stations with its components users will turn off. In this case, the system must continue to perform its functions.

The operation of the host components of the system separately in the absence of the system center and their full-fledged actions to detect abnormalities by means of implemented functions must be maintained, as the center may be temporarily unavailable and then the whole system can not resist malware and computer attacks. In such cases, it is important to implement a horizontal interface between the components of the lower hierarchical system. This would make it possible to more effectively combat malicious acts in the absence of the center. For the organization of such interaction of components the corresponding protocol and processing of such events by parts of the center which is in components of the designed system is necessary.

With such requirements for the architecture of the designed distributed system, in which the host parts of the system are located in computer stations must decide on the presence of anomalies and transmit the detection result to the center of the distributed system and the network part must perform tasks to detect anomalies in the local network. synthesize the following characteristic properties, methods and functionalities:

1) centralization (single decision-making center) in decision-making in the system;

2) the presence of levels of hierarchy in decision-making in the distributed parts of the center in all components of the system at network nodes;

3) distribution of system components in different nodes in the network;

4) self-organization of the system when deciding on further steps in the system and its components;

5) dynamic formation of the system in the process of its functioning, both during the initial formation and in the process of long-term use;

6) network protocol for the interaction of the components of the designed system;

7) dynamic formation of the architecture of the system in the process of its operation from the mandatory component, in which part of the center of the upper level of the hierarchy, and part of the components of the system, not necessarily all components;

8) the functioning of the host components of the system separately in the absence of the center of the system and their full-fledged actions to detect abnormal manifestations by means of implemented functions;

9) implementation of methods for detecting anomalies in computer systems in the designed system.

Synthesizing the selected characteristic properties, methods and functional capabilities, we obtain a self-organized distributed system that is able to function in a local computer network and solve problems to detect anomalies in computer systems when filling it with the appropriate functionality.

We present a self-organized distributed system as a set of its components. Let the set M_{SDS} be the set of components of a self-organized distributed system (SDS). We denote the components of the system by $M_{SDS,i}$, where i is the number of the system component. The center of the self-organized distributed system is denoted by $M_{SDS,0}$, ie it is an element of the set of system components at i = 0. Then, the set of components of a self-organized distributed system is set as follows:

$$M_{SDS} = \{M_{SDS,0}, M_{SDS,1}, M_{SDS,2}, \dots, M_{SDS,N}\}, \tag{1}$$

where N is the number of components of the self-assembled distributed system, excluding the component containing the center.

Thus, the total number of components of the self-organized distributed system is N + 1. The minimum number of components is one. In this case, the self-organized distributed system is scaled down to one component, which detects anomalies in one computer station and does not contain the functionality to establish the next steps of the whole system, even if this single system component contains the system center. If the system contains more than one component and among them there is no one that contains the center, then all of them also function to ensure the detection of anomalies in their computer stations, do not contain functionality to establish the next steps of the whole system, but exchange information about projects in which they identified sources of abnormal manifestations.

We present the synthesized architecture of a self-organized distributed system, taking into account its representation through many components and characteristic properties, methods of detecting anomalies, functional capabilities of the structural scheme, which is shown in Fig. 1. The architecture of the self-organized distributed system with the display of the system center as a component is shown in Fig. 2.

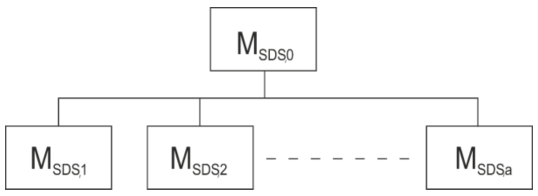
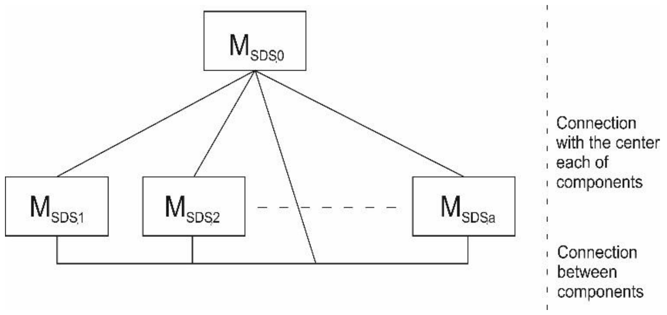


Figure 1: Architecture of a self-organized distributed system



M_{SDS0}
M_{SDS1}
M_{SDS2}
M_{SDSa}
Connection with the center each of components
Connection between components

Figure 2: Architecture of a self-organized distributed system with the mapping of the system center as a component

In the presented architecture of the self-organized distributed system, in contrast to the known solutions, the internal organization of interaction of parts of the center of the system between different levels of hierarchy and depending on the activity of system components at a certain time. This made it possible to divide some of the tasks from the center to a lower level of the hierarchy for decision-making, depending on the methods used to detect anomalies in a particular computer station. In addition, dividing the center's tasks into parts according to their purpose and in the absence of a higher-level component in which the center of the entire self-organized distributed system is located allows the exchange of messages about the source of objects that provoke abnormalities. This is important in the context of the specifics of the tasks to be solved by the system and the conditions of its operation, in particular under the destructive effects of malicious software. The image of the location of the decision-making center in the developed architecture of the self-organized distributed system is presented in Fig.3.

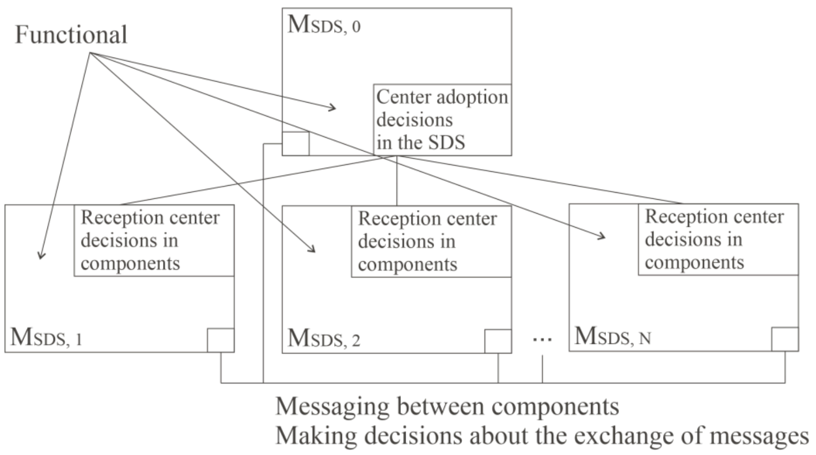

Functional
MSDS, 0
Center adoption decisions in the SDS
Reception center decisions in components
MSDS, 1
Reception center decisions in components
MSDS, 2
Reception center decisions in components
MSDS, N
Messaging between components
Making decisions about the exchange of messages

Figure 3: Places of the decision-making center in the architecture of a self-organized distributed system

As a result, the designed architecture of the self-organized distributed system allows to increase its capabilities by filling with implemented methods to detect anomalies in computer systems. In the architecture of the system, the functions of its center are distributed among the components, which speeds up the processing of events by processing directly in the network node in which the analysis of the anomalous manifestation took place. In addition, the system is designed so that its components can share the results of treatment of abnormal manifestations and their identified sources.

Distribution of components of a self-organized distributed system highlights the problem of ensuring the integrity of the system and the effective interaction of components, as all components are located at different nodes in the network. In addition, maintaining the integrity of a self-organized distributed system through the organization of effective communication between them is an important task and not only under normal conditions. Effectiveness in the organization of such communication in maintaining the integrity of the system is especially important in the face of the destructive effects of malicious software and computer attacks. Therefore, the method of maintaining the integrity of a self-organized distributed system through effective communication between components should be based on a network protocol unknown to attackers and have a set of options for further steps throughout the system under appropriate conditions. The integration of these two components into the integrity maintenance method is necessary to improve security directly for the system itself compared to other operating systems operating under known network protocols.

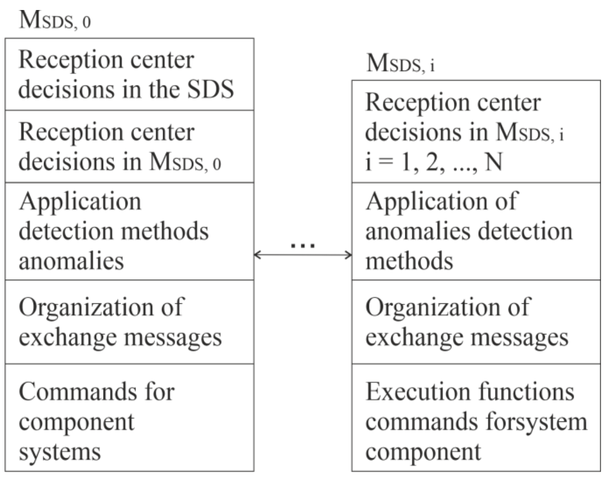

MSDS, 0
Reception center decisions in the SDS
Reception center decisions in MSDS, 0
Application detection methods anomalies
Organization of exchange messages
Commands for component systems
MSDS, i
Reception center decisions in MSDS, i i = 1, 2, ..., N
Application of anomalies detection methods
Organization of exchange messages
Execution functions commands forsystem component
...

Figure 4: Architecture of system components and connections between them

Consider first the formation of a network protocol for organizing the interaction of components. Consider the architectural features of the system in the part relating to the distribution of decision-making centers. In fact, it is from the decision centers that instructions will be sent on the transfer of data. Therefore, given the three types of relationships between components of the system, which depend on the levels of hierarchy in which they are located, we obtain different pairs of elements of three components in each as follows:

1) $(c_{SDS,0}, c_{SDS,i}, 1)$ – displays the transfer of the command from the center of the upper level, which is accepted by such notation as $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$, to the centers of the system in the components that are at the lower level, ie to the centers denoted by $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$, де $i = 1, 2, \dots, N$;

2) $(c_{SDS,i}, c_{SDS,0}, 2)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about possible anomalies, ie collected characteristics that need to be processed, to the center of the upper level $c_{SDS,0}$ for their processing;

3) $(c_{SDS,i}, c_{SDS,0}, 3)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about the results of processing the anomaly in this component to the center of the upper $c_{SDS,0}$;

4) $(c_{SDS,i}, c_{SDS,j}, 4)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about possible anomalies, ie collected characteristic features that require processing, to the center of the same level $c_{SDS,j}$ for their processing under the condition $j \neq i$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, N$;

5) $(c_{SDS,i}, c_{SDS,j}, 5)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about the results of processing the anomaly in this component, to the center of the same level $c_{SDS,j}$ for their processing under the condition $j \neq i$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, N$;

6) $(c_{SDS,i}, c_{SDS,j}, 6)$ – displays the transmission of the message (if there is information about the absence of components in the system with the center of the upper level, ie the actual decision center in the system) from the center of the lower level, which is denoted $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about possible anomalies, ie collected characteristics that need to be processed, to the center of the same level $c_{SDS,j}$ for their processing under the condition $j \neq i$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, N$;

7) $(c_{SDS,i}, c_{SDS,j}, 7)$ – displays the transmission of the message (if there is information about the absence of components in the system with the center of the upper level, ie the actual decision center in the system) from the center of the lower level, which is denoted $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ and where $i = 1, 2, \dots, N$, which contains information about the results of processing the anomaly in this component, to the center of the same level $c_{SDS,j}$ for their processing under the condition $j \neq i$, $i = 1, 2, \dots, N$, $j = 1, 2, \dots, N$;

8) $(c_{SDS,0}, c_{SDS,j}, 8)$ – displays the transmission of the message from the center of the upper level, which is denoted by $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$, which contains information about possible anomalies, ie collected characteristic features that require processing, to the center of the lower level $c_{SDS,j}$ for their processing while $j = 1, 2, \dots, N$;

9) $(c_{SDS,0}, c_{SDS,j}, 9)$ – displays the transmission of the message from the center of the upper level, which is denoted by $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$ and where $i = 1, 2, \dots, N$, which contains information about the results of processing the anomaly in this component, to the center of the lower level $c_{SDS,j}$ to carry out their processing while $j = 1, 2, \dots, N$;

10) $(c_{SDS,i}, c_{SDS,0}, 10)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$ and where $i = 1, 2, \dots, N$ to the center of the upper level $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$ for $i = 1, 2, \dots, N$ to notify about the inclusion of a computer station in the network and the successful launch of software system components, ie notification of readiness to start work as part of the system;

11) $(c_{SDS,0}, c_{SDS,i}, 11)$ – displays the transmission of the message from the center of the upper level, which is denoted by c_ (SDS, 0) to the center of the lower level $c_{SDS,i}$ and where $i = 1, 2, \dots, N$ for $i = 1, 2, \dots, N$ in order to notify the activation of the computer station in the network and the successful launch of the software system components, ie the message of readiness to start work as part

of the system in which it operates system center, ie updating data on the currently available architecture;

12) $(c_{SDS,i}, c_{SDS,0}, 12)$ – displays the transmission of the message from the center of the lower level, which is denoted by $c_{SDS,i}$ and where $i = 1, 2, \ldots, N$ to the center of the upper level $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$ in order to notify about the correct shutdown of the computer station in the network and the successful completion of the software system components while maintaining the characteristics of the computer station profile online;

13) $(c_{SDS,0}, c_{SDS,i}, 13)$ – displays the transmission of the message from the center of the upper level $c_{SDS,0}$ to the center of the lower level $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$, $c_{SDS,0} \in M_{SDS,0}$ for $i = 1, 2, \ldots j - 1, j + 1, \ldots, N, j \neq i$ to report j-th computer station in the network and the successful completion of the software j-th component of the system while maintaining the characteristics of the profile of the computer station in the network, notifications to all other active components of the system about the existing system architecture;

14) $(c_{SDS,j}, c_{SDS,i}, 14)$ – displays the transmission of the message from the lower level center, denoted by $c_{SDS,j}$ to the remaining active lower level centers $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ for $i = 1, 2, \ldots j - 1, j + 1, \ldots, N, j \neq i$ in order to report the correct shutdown of the j-th computer station in the network and the successful completion of the software of the j-th component of the system with the preservation of the characteristic features of the profile of the computer station in the network;

15) $(c_{SDS,0}, c_{SDS,i}, 15)$ – displays the transmission of the message from the center of the upper level $c_{SDS,0}$, $c_{SDS,0} \in M_{SDS,0}$ to the center of the lower level $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ for $i = 1, 2, \ldots j - 1, j + 1, \ldots, N, j \neq i$ in order to report problems in j - that computer station in the network and send it a command to block the software in it and forcibly shut down the software j-th component of the system while maintaining the characteristics of the profile of the computer station in the network, notifying all other active components systems on changing the existing architecture of the system associated with the removal of the j-th component of the system;

16) $(c_{SDS,j}, c_{SDS,i}, 16)$ – displays the transmission of the message from the centers of all levels to other centers of all levels $c_{SDS,i}$, $c_{SDS,i} \in M_{SDS,i}$ for $i = 0, 1, 2, \ldots j - 1, j + 1, \ldots, N, j \neq i$ in order to inform about the architecture of the formed distributed system, which includes all the initially specified components and their readiness to perform the specified functions or continue to work.

All messages between system components must be processed by the decision centers in the components, regardless of the level of the hierarchy, and only after processing messages or approving the received commands they are processed or executed by other parts of system components.

From the start of computer stations in the network, a situation may arise when a certain computer station in which the decision center of the system is located, first turns on or will be turned on before other computer stations in which the rest of the system, then the event described by three elements $(c_{SDS,i}, c_{SDS,0}, 10)$, $c_{SDS,i} \in M_{SDS,i}$, $i = 1, 2, \ldots, N$, $c_{SDS,0} \in M_{SDS,0}$, will take place as planned and further steps in the transmission of messages will be standard.

If it turns out that the computer station in which it finds the component with the decision-making center of the system, will turn on later than those computer stations in which the other components of the system, then the event described by the three elements $(c_{SDS,0}, c_{SDS,i}, 11)$, $i = 1, 2, \ldots, N$ will occur in a non-standard way. This is due to the fact that the late start-up or failure of the station can be caused by the destructive effects of malicious software or computer attacks. System components in which the center belongs to the lower level, will establish the absence of components with the center of the upper level of the system and will exchange messages with each other until the system center again notifies them of its activity and readiness. But the confirmation of such actions will be carried out according to a certain algorithm to prevent external interference by substituting components with the center of the upper level of the system. If certain computer stations shut down after a certain period, the components present in them will send messages and commands coming from the top-level system center to the lower-level system center regarding the following events or states: suspend components; process the data of characteristic features for the study of anomalies; restore components; send a component with the center of the lower level of the message or the specified command, ie to make an indirect reference to another component of the system; provide data on the current state of the computer station, i.e. the profile of its characteristics and the studied features of the anomaly. In addition to commands, the top-

level center of the system may receive messages with information that needs to be processed, forwarded to another component, or stored.
A graph with transition arcs, which takes into account three types of relationships between system components, depending on the levels of hierarchy on which they are located, is shown in Fig. 5.
The mapping of relationships between distributed system components and events that can be processed by the system and specified by elements (1-16) on the graph with transition arcs is complete and does not contain hanging vertices of the first degree, so the described events are sufficient to ensure operation distributed system and can be implemented in the steps of the method of maintaining the integrity of a self-organized distributed system. Adding new events that can occur in the system or be processed in it is possible, because the links in the column reflect the closure of all events, and their increase will actually increase the functionality of the system itself, as the number of components will not increase.

The list of further steps of the system will be determined by the states, which may include a self-organized distributed system or its components. Also, these states will depend on the number of active components of the system, the state of computer stations in the network. And, at the same time, these states will be intermediate in time, as the system will dynamically change its architecture and move from state to state. The states depend on the states of the system components, both active and disabled or removed by the system.

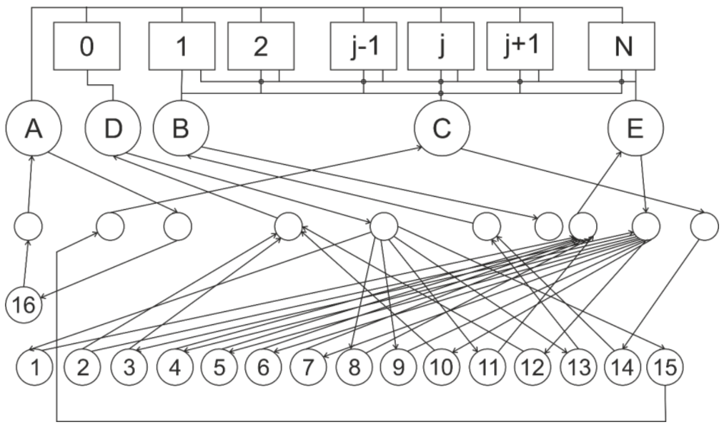


Figure 5: Graph with transition

The method of maintaining the integrity of the architecture of a self-organized distributed system in local computer networks includes the following iterative steps:

- step 1: execution $(c_{SDS,i}, c_{SDS,0}, 10)$ transmission of the message from the center of the lower level to the center of the upper level, $c_{SDS,0} \in M_{SDS,0}$ for $i = 1, 2, \dots, N$ in order to notify about the activation of the computer station in the network and the successful launch of the software components of the system, ie the message about the readiness to start work as part of the system;
- step 2: execution $(c_{SDS,0}, c_{SDS,i}, 1)$ to transfer the command from the center of the upper level to the centers of the system in the components that are at the lower level, and receive confirmation of receipt of the command;
- step 3: execution of the command in the component with the center of the lower level and sending the report to the component of the system with the center of the higher level;
- step 4: execution $(c_{SDS,i}, c_{SDS,0}, 2)$ and execution $(c_{SDS,i}, c_{SDS,j}, 4)$ to send a message from the center of the lower level, in which contains information about possible anomalies, ie collected characteristic features that require treatment, to the centers of the upper and lower levels for their treatment;
- step 5: execution $(c_{SDS,i}, c_{SDS,0}, 3)$ and execution $(c_{SDS,i}, c_{SDS,j}, 5)$ to send a message from the lower level center, in which contains information on the results of processing the anomaly to the centers of the upper and lower levels for their processing;

- step 6: execution $(c_{SDS,i}, c_{SDS,j}, 6)$ to send a message from the lower level center, which contains information about possible anomalies, ie collected characteristics that need to be processed, to the lower level centers to carry out their processing in the absence of a top-level center;

- step 7: execution $(c_{SDS,i}, c_{SDS,j}, 7)$ to send a message from the lower level center, which contains information about the results of processing the anomaly to the lower level centers to process them in the absence of the center upper level;

- step 8: execution $(c_{SDS,0}, c_{SDS,j}, 8)$ to send a message from the center of the upper level, which contains information about possible anomalies, ie collected characteristics that need to be processed, to the centers of the lower levels to carry out their processing;

- step 9: execution $(c_{SDS,0}, c_{SDS,j}, 9)$ to send a message from the center of the upper level, which contains information about the results of processing the anomaly to the centers of the lower levels to process them;

- step 10: execution $(c_{SDS,0}, c_{SDS,j}, 11)$ to send a message from the center of the upper level to all active components in order to notify the activation of the computer station in the network and the successful launch of software j- that component of the system, ie the message of readiness to start work as part of the system in which the center of the system operates;

- step 11: execution $(c_{SDS,i}, c_{SDS,0}, 12)$ to transmit a message from the center of the lower level to the center of the upper level in order to notify the correct shutdown of the computer station on the network and successful completion of software providing system components while preserving the characteristics of the computer station profile in the network;

- step 12: execution $(c_{SDS,0}, c_{SDS,i}, 13)$ to transmit a message from the center of the upper level to the center of the lower level in order to report the shutdown of the j-th computer station in the network and successful completion operation of the software of the j-th component of the system with the preservation of the characteristic features of the profile of the computer station in the network, ie notification of all other active components of the system about the existing system architecture;

- step 13: execution $(c_{SDS,j}, c_{SDS,i}, 14)$ to transmit a message from the lower level center to the rest of the active lower level centers in order to notify the correct shutdown of the j -th computer station in the network and successful completion of the software of the j-th component of the system while maintaining the characteristics of the profile of the computer station in the network;

- step 14: execution $(c_{SDS,0}, c_{SDS,i}, 15)$ to transmit a message from the center of the upper level to the center of the lower level in order to report problems in the j - th computer station in the network and send commands to block the software in it and forcibly terminate the software of the j-th component of the system with the preservation of the characteristics of the profile of the computer station in the network, ie notify all other active components of the system to change the existing system architecture associated with removal j-th system components;

- step 15: execution $(c_{SDS,j}, c_{SDS,i}, 16)$ to transmit a message from the centers of all levels to the rest of the centers of all levels in order to inform about the architecture of the distributed system, which includes all initially specified components and their willingness to perform assigned functions or continue working.

The scheme of the method of maintaining the integrity of the architecture of a self-organized distributed system in local computer networks is that its steps take into account the state of system components, transitions between components and due to its steps the whole distributed system can determine its next steps. The main steps (1-15) of the method are not performed sequentially, but correspond to the iterative scheme with the simultaneous parallel execution of certain steps in different components simultaneously. That is, a certain number of steps can be performed in parallel. Some steps may not be performed at some point. The transition of the system to certain subsequent states is based on certain steps of the method and depends on the set of possible given states, but in fact the system and its components go to certain steps of the method, ie transitions to subsequent states of the system and components method. This allows such an approach to implement such a characteristic of the system as self-organization, which, unlike other methods that take into account the discrete states of the system for transitions or intervals to determine the states of the system or its components, considers transitions as a goal for next steps method. Completion of the steps of the method under the conditions of correct shutdown of computer stations and software components of the system will be the completion of the self-organized distributed system.

Thus, a method has been developed to maintain the integrity of the architecture of a self-organized distributed system in local computer networks, which takes into account the state of system components, transitions between components and determines the next steps of the system. This has allowed them to build systems that are centralized and self-organizing and can decide on their next steps depending on the effects of malware and computer attacks.

4. Detection Of Anomalies In Computer Systems Based On The Main Component Method

Developed self-organized distributed system to detect anomalies in computer systems requires filling it with appropriate methods. Because it is implemented in a local computer network and is a distributed system, the subject area for research is network detection of anomalies. In addition, the processes that will be studied, in order to remain relevant to the results of their course, will require a prompt decision over time, as well as taking into account the clearly limited number of nodes in the network, which is a local computer network. As information about the data collected for decision-making will change rapidly over time, it is necessary to take into account adjustments to it. And taking into account these features (time, adjustment of collected data, limited number of nodes in the network, the use of a distributed system for data collection and decision making) can provide a method of main components. The peculiarity of this method is that when detecting anomalies in the network, the projection of data on the residual subspace is continuously monitored. Using a distributed system that will collect data for analysis from all network nodes in which system components are located will require information from detectors in specific network nodes. The accumulation of data from a large number of nodes in the network, as well as their periodic addition, will affect the speed of processing and the need to optimize them. In addition, the data received from the nodes in the network will have many different representations and not always all of them will have the same weight and significance. Some of the collected data will need to be optimized and reduced in size with minimal loss of information. These requirements are taken into account in the method of principal components (developed by K. Pearson, 1901).

The principal components method may have certain optimizations and improvements depending on the field of application and the possibility of combined application with other methods or within the framework of implementation in the architecture of certain systems. Consider the classical formulation, essence and steps of the principal components method. The initial data collected from the nodes in the computer network is collected in the center of the distributed system, where we present them as a matrix. Let k be the number of nodes in the network in which the components of the distributed system are installed and from which data is collected for analysis. Let us represent a finite set of different data from a node in a network in an ordered sequence by a vector:

$$V_f = (v_1, v_2, \dots, v_f), \tag{2}$$

where v_i is the value of data from the studied component of the node in the network.

Some of the values of data v_i from the studied component of the node in the network are typical components and therefore they can be grouped into classes. For example, such typical test components may be operating system files, individual sets of directories, or executable programs in general. Also, individual classes can be entered with RAM processes and data from all ports.

Thus, as data from nodes in the network we will consider operating system files, executable files in directories, processes in internal memory, data from computer ports, user profile characteristics, network activity from the computer. Running programs can have several characteristics. In particular, for example, creation time and size. Certain classes may be absent or the number of their elements may not be taken all. All data will be presented in numerical form. The number of indicators from the node in the network is large. Not all indicators in the class will have significant values that will affect the result when deciding on the presence of an anomaly. Therefore, it is necessary to involve the appropriate mathematical apparatus, in particular the method of principal components, to reduce the dimensionality of the data.

Obtained from a node in the network matrix of initial data has dimension $k \times f$, where k is the number of nodes in the network in which the observation is performed, f is the number of elementary indicators. To apply the principal components method, you need to perform the following steps: obtain the initial data matrix Q_1; transition from the matrix of initial data to the matrix of centered and normalized values of the features Q_2; transition from the matrix of centered and normalized values of features to the matrix of paired correlations Q_3; transition from the matrix of paired correlations to the diagonal matrix of eigenvalues; transition from the diagonal matrix of eigenvalues V_z to the factor mapping matrix Q_4; transition from the matrix of factor mapping to the matrix of values of the main components of smaller dimension Q_5 than the matrix of the original data. The scheme of steps of application of the method of principal components is shown in fig. 6, where V_v is a matrix of normalized eigenvectors.

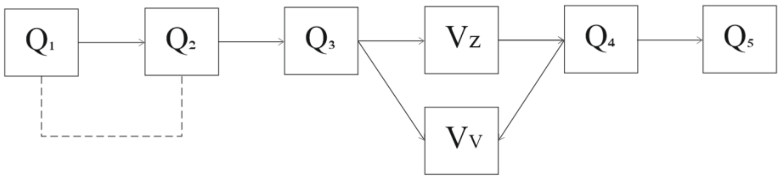


Figure 6: Scheme of connection of steps in the method of principal components

The elements of the matrix of factor mapping Q_4 are weights. First, the matrix Q_4 has dimension $k \times f$ - by the number of elementary features. In the process of transformation, only the most important features remain and the dimension becomes smaller.

The principal components method belongs to the statistical methods of factor analysis, which analyzes the influence of individual factors on the resulting indicator. The principal components are calculated as eigenvectors and eigenvalues of the covariance matrix of the initial data. The task of the analysis of the main components is to approximate the data by linear combinations of smaller dimension or to find a subspace of smaller dimension.

In the developed self-organized distributed system there are components that are located in the nodes in the network. Each of the components has detectors of different types, which collect certain information for further processing in the center. This information is obtained by receiving data streams of a certain time series. At each node in the network at certain intervals, data is collected simultaneously.

The information collected may be as follows: all executable files on the computer with information about the time of their creation or last change, as well as their size; the number of TCP connection requests per second; number of transactions per minute; the amount of traffic in ports per second; number of running processes; the amount of free internal memory; the amount of hard disk space and free space in it. It is also important to take into account the possibility of comparing the results with previous results according to time slices. A self-organizing distributed system stores collected and processed data. Then, when performing the analysis, it takes into account the last obtained values of the features and the previous ones, ie processes them with a certain time window. The decision-making center in the self-organized distributed system monitors the total set of values of the time series features and makes decisions on security issues of individual nodes and the entire computer network. To do this, the system needs to determine the volume anomalies. Abnormal manifestations in the network are unusual load levels caused by worm viruses, distributed attacks, denial of service, device failure, incorrect configurations, the spread of malicious software in nodes and the network as a whole, and so on. Each node collects information according to the specified characteristics and at a certain time step, these collection results are sent to the processing center of the distributed system. The decision center of the distributed system monitors in a sliding time window of size r for each time series from each node in the network. The number r indicates the amount of the most recent data among all data received from the node in the network and stored in the center of the self-organized distributed system. These data are presented in the appropriate time series. Let us denote these data taking into account formula (1) and the fact that such vectors will be obtained at different time intervals, as well as taking into account the fact that the number of different nodes in the network is k. Let $V_{f,j}$ be a vector of signs from the j -th node in the network. Then, a finite set of different data from the j - th node in the network is represented by an ordered sequence:

$$\left(v_{1,j}, v_{2,j}, \dots, v_{f_j,j}\right), \tag{3}$$

where $v_{i,j}$ is the value of the data from the studied i-th component from the j-th node in the network; $i = 1, 2, \dots f_j$; f_j - the number of values of these features from the j - node in the network; $j = 1, 2, \dots, k$; k is the number of nodes in the network.

The obtained vectors from individual nodes in the network are represented in the matrices W_q, where q is the node number in the network from 1 to k. Matrices W_q have dimension $t_i \times f_j$, where t_i is the time at which the results of data collection from all nodes in the network were recorded simultaneously, i is the number of time slices during the entire observation time; f_j - the number of values of these features from the j - node in the network; $j = 1, 2, \dots, k$; k is the number of nodes in the network. Not all computers connected to the network can be turned on at the same time. Similarly, not all of them can be turned off at the same time. Therefore, the countdown is taken from the main computer where the system center operates. In fact, the center of the system is divided into time intervals and the formation of the time series. If certain computers are turned off, then the data in the system for all their indicators in the feature vectors will be equal to -1. Zeros as numerical values cannot be used because they can be significant results of signs. It is also important to reflect in the matrices W_q the relationships that can be between the components of the vectors. For example, the time and file size are different components of a vector, but they belong to the same object. Therefore, to display such information, it is necessary to maintain relevant data, for example, in the vector of initial (initialized) data features, which may have the same dimension, but the value of the components can be determined by the formula at t_0:

$$V_{f_j} = \left(v_{1,j}, v_{2,j}, \dots, v_{f_j,j}\right); \tag{4}$$
$$v_{i,j} = v_{i+1,j}, v_{i,j} = i, if\ adjacent\ features\ relate\ to\ a\ single\ object,$$
$$if\ each\ subsequent\ feature\ also\ applies\ to\ the\ same\ object,$$
$$then\ its\ value\ is\ set\ too\ i;$$
$$v_{i,j} = 0,, if\ the\ adjacent\ (right\ and\ left)\ features\ relate\ to\ different\ objects;$$

where $v_{i,j}$ is the value of the data from the studied i-th component from the j-th node in the network; $i = 1, 2, \dots f_j$; f_j - the number of values of these features from the j - node in the network; $j = 1, 2, \dots, k$; k is the number of nodes in the network.

We define a matrix of values of indicators of signs W_j from j - that node in a network so:

$$W_j = \begin{pmatrix} v_{1,j,t_0} & v_{2,j,t_0} & \cdots & v_{f_j,j,t_0} \\ v_{1,j,t_1} & v_{2,j,t_1} & \cdots & v_{f_j,j,t_1} \\ \cdots & \cdots & \cdots & \cdots \\ v_{1,j,t_g} & v_{2,j,t_g} & \cdots & v_{f_j,j,t_g} \end{pmatrix}, \tag{5}$$

where v_{i,j,t_s} - the value of the data from the studied i-th component from the j-th node in the network at time t_s; $i = 1, 2, \dots f_j$; f_j - the number of values of these features from the j - node in the network; $j = 1, 2, \dots, k$; k is the number of nodes in the network; s is the number that corresponds to the iteration of obtaining the values of the features; $s = 0, 1, 2, \dots, g$; g is the last iteration of obtaining feature values from the j-th node in the network.

Matrices W_j for certain j - those nodes in the network in the system provide an opportunity to represent the collected values of the features. In fig. 3 shows them as part of a self-organized distributed system.

The matrices W_j for certain j nodes in the network may have a different number of columns, ie the numbers f_j in all matrices will be mostly different, because it is related to the number of objects that are included for monitoring in the node in the network. You can choose when setting up a self-organized distributed system, which will implement the method of principal components, so that the number of features taken from different nodes in the network was the same, ie choose common for research. But such an approach will make it difficult to improve the detection of anomalies, because it may lose a significant part of the objects in which there will be manifestations of abnormalities. Therefore, the number of monitoring features will be selected differently at different nodes in the network. Thus, all matrices will have a different number of columns, and the number of columns will be finite, but can be very large, because it will contain information about many features and their values.

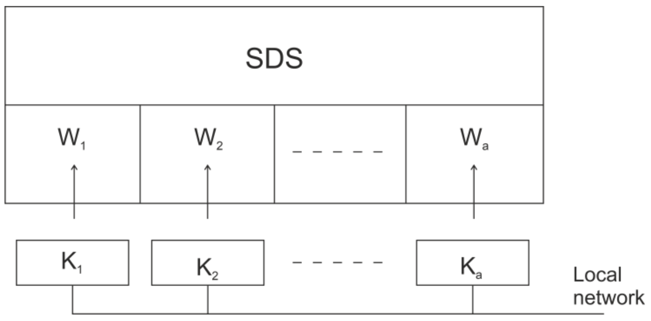


Figure 7: W_j matrices in the architecture of a self-organized distributed system

The number of rows of the matrix W_j for different j-nodes in the network will be the same, because the values of them will be obtained by command at specific points in time for all the same. Another problem that will arise in this process will be the constant accumulation of information. In this case, the matrices W_j for different j - nodes in the network will increase dynamically by the number of rows, which will need to be taken into account in the implementation.

Thus, the accumulated information in the matrices W_j for different j - nodes in the network is necessary for the application of the principal components method in order to detect anomalous manifestations. Time series are stored in the matrices W_j for each j - node in the network. The sliding window will be the interval $]t_i;\ t_l[$, where i and l are numbers of time indicators to obtain the values of features from different j - nodes in the network. In the future, each of the matrices W_j must pass into the matrix Q_1 from Fig. 2 to start applying the method of principal components.

The information in the matrices W_j for each j - node in the network is heterogeneous and does not allow without appropriate processing to draw a conclusion about the presence or absence of abnormal manifestations. The self-organized distributed system anomaly detection system measures the total amount of traffic (in bytes) for each network connection and periodically collects data at the center. Then, the method of principal components is applied in the automatic mode implemented in it, taking into account the data collected in the matrix (5). Similarly, the system processes data attributes about files, processes, and computer resource usage. The prompt processing of data requires that the periods of their renewal be relatively short. Although small periods lead to an increase in additional costs for attracting resources and carrying out communication work.

Consider the essence of the method of principal components and its features when applied to detect anomalies. The geometric interpretation of this method is as follows; not plane placed points; conducted direct; the distances from points to a straight line are determined, ie projections are made from points to a straight line; the sum of squares of projections of points on a straight line is defined; searching for a new line so that the sum of the squares of the projections of the points on the line is minimal. Similarly, the formulation of this problem is scaled to three-dimensional space and n-dimensional. If you successfully try to find such a line, the number of points that will affect the sum of the squares of the projections will decrease, because some of them will be on this new line. The same points that will be removed from the line will affect the result of the sum of the squares of the projections and will be significant. In the problem of detecting anomalies, the values of the considered features can also be given by points on the plane and finding a straight line to solve the problem of best approximation of a finite set of points allows to determine those points, ie features that significantly affect the sum and therefore are abnormal manifestations.

Let's make a formal statement of the approximation problem and move from it to the method of principal components. For example, let a finite number of vectors $v_{1,j}, v_{2,j}, \dots, v_{f_j,j} \epsilon R^u$, where the vectors $v_{1,j}, v_{2,j}, \dots, v_{f_j,j}$ correspond to the values of signs from j - that node, f_j - the number of signs, ie vectors. For each $p = 1, 2, \dots, u-1$ it is necessary to find $M_p \subset R^u$ from p - dimensional linear combinations in $R^u$and provided that the sum of squares of deviations $v_{i,j}$ from M_p was minimal. In particular, the formal record of such a problem is given by the formula:

$$\sum_{i=1}^{f_j} d^2(v_{i,j}, M_p) \rightarrow min, \quad (6)$$

where $d\,(v_{i,j}, M_p)$ is the distance from the point $v_{i,j}$ to the linear combination M_p.
The distance from a point to a linear combination can be determined by various metrics, including Euclidean. Set the orthonormal set of vectors $\alpha_1, \alpha_2, \dots, \alpha_{f_j} \epsilon R^u$, then the linear combinations M_p for all $p = 1, 2, \dots, u-1$ set by the formula:

$$M_p = \alpha_0 + \sum_{i=1}^{p} \beta_i * \alpha_i, \quad (7)$$

where $\beta_i \epsilon R$ and are the coefficients in the linear expansion M_p.
The approximation problem for each $p = 1, 2, \dots, u-1$ is solved by finding the linear combinations $M_1 \subset M_2 \subset M_3 \subset \dots \subset M_{u-1}$, where M_p is represented by formula (7). According to formula (6) it is necessary to have the values of the applications responsible for the distance. We present the definition of terms from formula (6) taking into account the given representations of linear combinations by the formula:

$$d^2(v_{i,j}, M_p) = \left|v_{i,j} - \alpha_0 - \sum_{s=1}^{p} \alpha_s \cdot (\alpha_s, v_{i,j} - \alpha_0)\right|^2, \quad (8)$$

where the square of the distance $d^2(v_{i,j}, M_p)$is determined by the Euclidean norm; the expression $(\alpha_s, v_{i,j} - \alpha_0)$ is defined as the scalar product of the vectors α_s and $v_{i,j} - \alpha_0$.
All linear combinations M_p are determined by an orthonormal set of vectors $\{\alpha_1, \alpha_2, \dots, \alpha_{f_j-1}\}$, which are vectors of principal components, and a vector α_0. Finding the vector α_0 is carried out by the formula:

$$\alpha_0 = \arg\min_{\alpha_0 \in R^u} \left(\sum_{s=1}^{f_j} d^2(v_{s,j}, M_0)\right). \quad (9)$$

According to formula (9) we obtain that α_0 is calculated by the formula:

$$\alpha_0 = \frac{1}{f_j} \sum_{i=1}^{f_j} v_{i,j} = \overline{v_j}. \quad (10)$$

Thus, α_0 minimizes the sum of the squares of the distances to the data points, ie the values of the features and according to formula (10) is the average value.

To find the vectors of the principal components, you need to perform the following sequence of steps of the optimization problem:

1) reduce all $v_{i,j}$ by the value $\overline{v_j}$, then the sum of all obtained $v_{i,j}$ will be equal to zero;

2) the first main component is calculated by the formula:

$$\alpha_1 = \arg\min_{|\alpha_1|=1} \left(\sum_{s=1}^{f_j} \left|v_{s,j} - \alpha_1 \cdot (\alpha_1, v_{s,j})\right|^2\right).$$

When obtaining several solutions for further calculations, choose one of them;

3) subtract from the data the projection on the first main component by the formula:

$$v_{i,j} = v_{i,j} - \alpha_1 \cdot (\alpha_1, v_{i,j});$$

4) the second main component is found by the formula:

$$\alpha_2 = \arg\min_{|\alpha_2|=1} \left(\sum_{s=1}^{f_j} \left|v_{s,j} - \alpha_2 \cdot (\alpha_2, v_{s,j})\right|^2\right).$$

If we obtain several solutions for further calculations, we choose one of them;

5) similarly to step 3 from the data subtract the projection on the p-1 main component by the formula:

$$v_{i,j} = v_{i,j} - \alpha_{p-1} \cdot (\alpha_{p-1}, v_{i,j});$$

6) the p-th main component is found by the formula:

$$\alpha_p = \arg\min_{|\alpha_p|=1} \left(\sum_{s=1}^{f_j} \left|v_{s,j} - \alpha_p \cdot (\alpha_p, v_{s,j})\right|^2\right).$$

If we obtain several solutions, we choose one of them for further calculations.

For each iterative stage of the algorithm application, the projection on the previous main component is subtracted. The vectors thus obtained $\alpha_1, \alpha_2, \dots, \alpha_{u-1}$ will be orthonormal. This is achieved by solving the optimization problem. In order to avoid calculation errors due to the influence of rounding use the inclusion in the conditions of the optimization problem of the following condition:

$$\alpha_p \perp \{\alpha_1, \alpha_2, \dots, \alpha_{p-1}\}. \quad (11)$$

Calculations of α_i can be performed in other ways. In particular, the first main component maximizes the sample variance of the data projection. Therefore, in fact, as for the approximation problem, the solution of which is given in the presented algorithm, at each step of the iteration you need to calculate the first principal component for the data from which the projections on all previously found principal components are removed. Problems on the calculation of principal components are reduced to the problem of diagonalization of the covariance matrix. The basis of eigenvectors is represented in the covariance matrix, so the matrix is diagonal. In this case, the covariance coefficient between the different coordinates is zero. The mathematical content of the principal components method is the spectral decomposition of the covariance matrix, but with certain transformations this problem becomes the problem of singular decomposition of the data matrix. Although formally the problems of singular decomposition of the data matrix and spectral decomposition of the covariance matrix coincide, the algorithms for calculating the singular decomposition directly, without calculating the covariance matrix and its spectrum, are more efficient and stable. This is necessary when using the principal components method to solve application problems of a particular subject area in information technology, where large sets of input data can cause computational errors or increase the duration of calculations, then you need to choose more stable algorithms and high speed.

To convert the data to the principal components, we construct a matrix from the vectors of the principal components so that the orthonormal vectors-columns of the principal components are arranged in descending order of eigenvalues. This allows you to focus most of the data variation in the first coordinates after the conversion. As a result, you can discard the remaining ones and get a space of reduced size.

Thus, the principal components method makes it possible to reduce the dimensionality of data, which is important in detecting malicious software and computer attacks, because it is necessary to process a lot of initial different types of information that is dynamically accumulating.

5. Improving the method of centralized detection of distributed anomalies by the main components search algorithm

The use of a self-organized distributed anomaly detection system in computer systems makes it possible to search directly in one computer station or in several at the same time. In both cases, you can use the principal components method as a step-by-step iterative algorithm to obtain numerical values of the characteristics of the characteristics directly obtained in one computer station and several for some time in a sliding window. Also, a self-organized distributed anomaly detection system can investigate the information received from nodes in the network about the manifestations of anomalies for abnormalities that correspond to either malicious software or computer attacks. Given the difficulty of detecting malicious software or computer attacks due to the limited number of signs that can detect abnormalities, as well as the presence of excessive amounts of different and heterogeneous information collected from nodes in the network, it is necessary to improve the method of centralized detection of distributed anomalies. a component that would reduce the dimensionality of information collected at nodes in the network without losing its value and quickly process in a single center to ensure the relevance of the detection result, which would improve the efficiency of detection.

To ensure the detection of distributed anomalies using the method of centralized detection of distributed anomalies by the main components search algorithm, we will develop a method of detecting anomalies in one of the computer stations in the network.

The increase in activity in the network to its nodes is a sign that these may be malicious manifestations and need to be investigated. In real time, increased activity quickly changes to moderate, so you need effective tools and implemented in them methods that would respond quickly to such events. Otherwise, the relevance of the information received by the system about the activity in the network is directed to its node, as well as the reaction to it will lose the need. The main feature that needs to be explored primarily in the network that is actually responsible for increased activity is the amount of traffic. There are many methods of processing network traffic, taking into account the different topologies of networks and access channels to corporate or local networks. In particular, they also take into account the peculiarities of traffic in the implementation of distributed attacks and the search for identity in its parts.

The concept of network traffic in the formulation of the problem of anomaly research includes the study of the amount of data moving in the network over time. For the proper functioning of computer networks, they need to control, analyze, model and manage the relevant specialized tools. Particularly important in the process of detecting anomalies are the analysis and measurement of network traffic, which includes monitoring traffic, changes in it, trends, measuring the amount and type of traffic. Receiving reports through various specialized means of network traffic provides information on the prevention of malicious activity and allows you to ensure network security. Fragments of the volume of data traffic during a certain time (three different time intervals) in the network of Khmelnytsky National University are shown in Fig. 8. As can be seen from the graphs in time intervals, the amount of traffic varies and can deviate significantly from the average value that can be used to identify abnormalities.

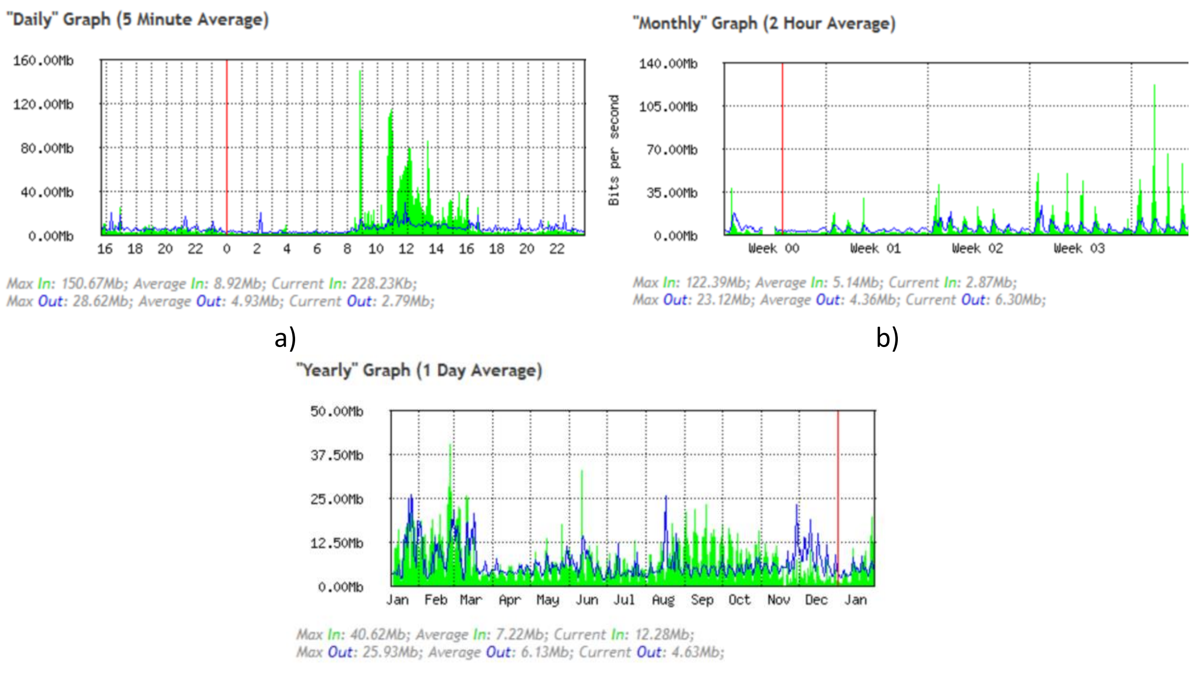


Figure 8: Image of the amount of data traffic over time (three different time intervals) in the network of Khmelnytsky National University

Data transmission in computer networks is carried out mainly in network packets. These packets provide network loads. There can be many packet transmission options and it is carried out according to network protocols. Upon arrival at the destination, depending on the rules and protocols, the packages need to check all, check the integrity and source of receipt. If you consider the traffic in the trunk lines, the anomalies of its volume may go unnoticed due to the enlarged view. Measurement results can be large, depending on the number of lines, but normal traffic models are in a smaller subspace. The allocation of this subspace of network traffic, using the method of principal components in traffic, allows you to identify volume anomalies in the subspace.

To analyze anomalous manifestations in network traffic, we first use the following characteristic parameters: traffic load factor; typical package size; the average number of fragmented packets. To study the load factor of network traffic, consider the following options: network traffic over time decomposes into a time series; network traffic is compared over certain time periods.

If the network traffic is received dynamically over a period of time, we will break it down into a time series. For example, let a finite number of vectors be given for its representation $v_{1,j}, v_{2,j}, \dots, v_{f_j,j} \epsilon R^u$, where the vectors $v_{1,j}, v_{2,j}, \dots, v_{f_j,j}$correspond to the values of the signs of traffic from the j - node in the network, f_j - the number of signs, ie vectors. For the case of traffic representation, an example of which is shown in the graphs in Fig. 8, after the time of its receipt and the volume at a particular time, we obtain that the value of $f_j = 2$. Then, the pair of vectors $v_{1,j}, v_{2,j}$ will represent network traffic during the time specified by the vector $v_{1,j}$. The total amount of traffic at a particular point in time will be represented by the vector $v_{2,j}$ and will be measured in bytes for all connections. Thus, each point of the graph, representing the amount of network traffic, is set by a pair of values. Over a certain time interval, a self-organized distributed anomaly detection system with certain fixed time periods collects these pairs of points. The numbering of points starts from the first pair obtained and continues to the point that is the last of the expected points. After the specified number of pairs is collected, the system centralizes them. The data presented in this way are two-dimensional. Let us represent a pair of vectors for a certain number of q observation points as follows:

$$\begin{pmatrix} v_{1,j.1} & v_{1,j.2} & \cdots & v_{1,j.q} \\ v_{2,j.1} & v_{2,j.2} & \cdots & v_{2,j.q} \end{pmatrix} \tag{12}$$

After receiving q pairs by the system and processing, synchronously the amount of network traffic continues to be displayed by the system in subsequent pairs. The self-organized distributed system, after processing a certain set of data specified by formula (12), receives part of the data actually updated and leaves from the processed data pairs that came last. The first pairs of points, after processing, are removed from further calculations. The number of such deleted pairs depends on the processing time and the time spent transferring the data. If the time spent is greater than the time spent collecting q new pairs by the system, then these collected new pairs are lost because a new set of subsequent pairs will start. To solve this problem, it is necessary to increase the collection interval of adjacent pairs of vectors.

6. Conclusions

Based on the results of theoretical and practical research, a self-organized distributed system for detecting anomalies in computer systems has been developed according to the main components method to improve the efficiency of detecting malicious software and computer attacks.

The following main results were obtained:

It is established that the detection of malicious software and computer attacks in local computer networks according to the studied methods and means of detection can be implemented by methods of detecting anomalies in computer systems and creating distributed anomaly detection systems.

Improved architecture of self-organized distributed system, in which, unlike known solutions, improved internal organization of interaction of parts of the system center between different levels of the hierarchy and depending on the activity of system components at a time, based on the distribution of decision center components with a division of the center between the upper and lower levels of the hierarchy. The result of such a designed architecture of a self-organized distributed system is the ability to increase its functionality by filling it with implemented methods to detect anomalies in computer systems. The system is designed so that its components can share the results of processing anomalies and their identified sources.

Developed a method to maintain the integrity of the architecture of self-organized distributed system in local computer networks, which takes into account the state of system components, transitions between components and determines further steps of the system, which allowed to build distributed systems with a single decision center decisions about their next steps depending on the effects of malicious software and computer attacks.

Improvement of the anomaly detection method according to the method of main components in computer systems in the network made it possible to apply it not to one computer station, but to a group of stations with self-organized distributed anomaly detection system in computer systems in the network. Its application has made it possible to reduce the amount of data and, accordingly, to speed up their exchange between system components.

Experimental studies with the developed implementation of a self-organized distributed system for detecting anomalies in computer systems according to the obtained coefficients confirmed the effectiveness of the proposed solutions and the developed distributed system for its operation in the computer network.

7. References

[1] A. Mohiuddin, N.M. Abdun, H. Jiankun. A survey of network anomaly detection techniques. Journal of Network and Computer Applications 60 (2016) 19-31.

[2] J.Bernadette Stolz, T. Jared, A. Heather Harrington, N. Vidit, Geometric anomaly detection in data. Proceedings of the National Academy of Sciences (2020), 117(33) 19664-19669. doi: 10.1073/pnas.2001741117

[3] Y. Xiang, L. Hui, Y. Xianfei, Y. Chen, S. Haifeng An adaptive method based on contextual anomaly detection in Internet of Things through wireless sensor networks, International Journal of Distributed Sensor Networks 16(5) (2020). doi: 10.1177/1550147720920478

[4] M. Goldstein, S. Uchida A Comparative Evaluation of Unsupervised Anomaly Detection Algorithms for Multivariate Data.. PLOS ONE 11(4) (2016). Doi:10.1371/journal.pone.0152173

[5] M.A. Hayes, M.A. Capretz, Contextual anomaly detection framework for big sensor data. Journal of Big Data 2(2) (2015). doi:10.1186/s40537-014-0011-y

[6] L. Liu, M. Hu, C. Kang, X. Li, Unsupervised Anomaly Detection for Network Data Streams in Industrial Control Systems. Information 11 (2020). doi:doi.org/10.3390/info11020105

[7] X. Xiaodan, L. Huawen, Y. Minghai, Recent Progress of Anomaly Detection. Complexity, 2019, (2019). doi:10.1155/2019/2686378

[8] H. Jianwen, C. Zhen, Z. Hailong, Detecting anomalies in data center physical infrastructures using statistical approaches. Journal of Physics: Conference Series, 1176(2) (2019).

[9] A. Fisch, D. Grose, I.A. Eckley, P. Fearnhead, L. Bardwell,. anomaly: Detection of Anomalous Structure in Time Series Data, 2020 arXiv: Applications.arXiv:2010.09353

[10] A. Anta, T. Hadjistasi, N. Nicolaou et al. Tractable low-delay atomic memory. Distrib. Comput. 34 (2021) 33–58. doi:10.1007/s00446-020-00379-y

[11] L. Ouyang, Y. Huang, H. Wei, J. Lu Achieving Probabilistic Atomicity With Well-Bounded Staleness and Low Read Latency in Distributed Datastores, Proceeedings of IEEE Transactions on Parallel and Distributed Systems, 32(4) (2021) 815-829. doi: 10.1109/TPDS.2020.3034328.

[12] A. Lakshman, P. Malik Cassandra: A decentralized structured storage system, SIGOPS Operating Syst. 44(2) (2010) 35-40.

[13] C. Ganesh, A. Patra Optimal extension protocols for byzantine broadcast and agreement. Distrib. Comput. 34 (2021) 59–77. doi:10.1007/s00446-020-00384-1

[14] Z. Huang, B. Radunovic, M. Vojnovic, et al. Communication complexity of approximate maximum matching in the message-passing model. Distrib. Comput. 33 (2020) 515–531. doi:10.1007/s00446-020-00371-6

[15] A. Czumaj, C. Konrad, Detecting cliques in CONGEST networks. Distrib. Comput. 33 (2020) 533–543. Doi:10.1007/s00446-019-00368-w

[16] A. Abboud, K. Censor-Hillel, S. Khoury, et al. Fooling views: a new lower bound technique for distributed computations under congestion. Distrib. Comput. 33 (2020) 545–559. Doi:10.1007/s00446-020-00373-4

[17] G.A. Di Luna, P. Flocchini, T. Izumi, et al. Fault-tolerant simulation of population protocols. Distrib. Comput. 33 (2020) 561–578. doi:10.1007/s00446-020-00377-0

[18] F. Ellen, R. Gelashvili, N. Shavit, et al. A complexity-based classification for multiprocessor synchronization. Distrib. Comput. 33 (2020) 125–144. doi:10.1007/s00446-019-00361-3

[19] S. Chatterjee, G. Pandurangan, P. Robinson, The complexity of leader election in diameter-two networks. Distrib. Comput. 33 (2020) 189–205. doi:10.1007/s00446-019-00354-2

[20] C. Busch, M. Herlihy, M. Popovic, et al. Time-communication impossibility results for distributed transactional memory. Distrib. Comput. 31 (2018) 471–487. doi:10.1007/s00446-017-0318-y.

[21] L. Boczkowski, A. Korman, E. Natale, Minimizing message size in stochastic communication patterns: fast self-stabilizing protocols with 3 bits. Distrib. Comput. 32 (2019) 173–191. doi:10.1007/s00446-018-0330-x.

[22] B. Min, V. Varadharajan, Feature-Distributed Malware Attack: Risk and Defence, Computer Security - ESORICS 2014. ESORICS 2014. Lecture Notes in Computer Science, 8713 (2014) 457-474 doi:10.1007/978-3-319-11212-1_26.

[23] H. V. Nath, B.M. Mehtre, Static Malware Analysis Using Machine Learning Methods, Recent Trends in Computer Networks and Distributed Systems Security, Communications in Computer and Information Science, 420 (2014) 440-450. doi:10.1007/978-3-642-54525-2_39

[24] M. G. Merayo, R. M. Hierons, M. Núñez, Passive testing with asynchronous communications and timestamps. Distrib. Comput. 31 (2018) 327–342.

[25] B. Savenko, S. Lysenko, K. Bobrovnikova, O. Savenko, G. Markowsky, Detection DNS Tunneling Botnets, Proceedings of the 2021 IEEE 11th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS), Cracow, Poland, September 22-25 (2021) 64-69

[26] O. Savenko, S. Lysenko, A. Nicheporuk, B. Savenko, Metamorphic Viruses' Detection Technique Based on the Equivalent Functional Block Search, CEUR Workshop Proceedings 1844 (2017) 555–569.

[27] O. Savenko, S. Lysenko, A. Nicheporuk, B. Savenko, Approach for the Unknown Metamorphic Virus Detection, Proceedings of the 8-th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Bucharest Romania, September 21–23, (2017) 71–76.

[28] O. Pomorova, O. Savenko, S. Lysenko, A Kryshchuk, Multi-Agent Based Approach for Botnet Detection in a Corporate Area Network Using Fuzzy Logic, Communications in Computer and Information Science 370 (2013) 243-254.

[29] Oleg Savenko, Sergii Lysenko, Andrii Kryshchuk, Yuriu Klots. Botnet detection technique for corporate area network. Proceedings of the 7-th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, Berlin (Germany), September 12–14, 2013. Berlin, 2013. Pp. 363–368. ISBN 978-1-4799-1426-5.