Інформаційні технології та системи управління

UDC 004.457

DOI: 10.30748/soi.2020.161.08

V. Martovitsky¹, I. Ruban¹, N. Lukova-Chuiko², E. Kortyak¹, Y. Kruglikov¹

¹ Kharkiv National University of Radio Electronics, Kharkiv

² Taras Shevchenko National University of Kyiv, Kyiv

MODEL MONITORING NETWORK INFRASTRUCTURE BASED ON STANDARD FIPA

The article focuses on the problem of creating systems for monitoring the functioning of the network infrastructure of distributed computer systems. This publication represents a model of multi-agent functioning for monitoring anomalies of distributed computer systems functioning components, which details the process of monitoring anomalies in the functioning of distributed computer systems and allows to integrate the monitoring system with other components of the cyber security systems. The model of a monitoring system designed to solve the problems of collecting and storing parameters received from sensors is characterized not only by their target functions, but also by functional capabilities that ensure the implementation of target functions, hierarchy and degree of parallelism in solving problems, uniformity or heterogeneity of the modular structure, organization of information collection in real-time data processing and network exchange of information with subscribers.

Keywords: multi-agent system, agent, distributed computer systems, network infrastructure, monitoring system, cyber-security.

Introduction

Radical changes are occurring in network infrastructure monitoring systems. They are caused by growing competition in the market, the growth of requirements for the quality of security, technical re-equipment of communication networks, changing in the nature of traffic distribution. All this leads to the need of controlling a large number of various technologies networks functioning parameters [1].

Restructuring monitoring data requires harmonization of information that can be reduced to a single view, or expanding data structures by adding unique attributes, which allows to map data from different sources. One way or another, this task is very time consuming, especially for remote repositories, as it requires both technical and organizational interaction from sources.

In addition, after the re-structuring of the data, a qualitative modification of the means of access to information will be required with new data structures.

In this regard, a model of a data collection and storage system is proposed, that provides work with many heterogeneous sources by integrating them in order to obtain a more complete collection of related information.

The system is based on a multi-agent approach and allows you to not stop processing requests when performing modifications to the set and structures, that are used in databases.

The purpose of this article is to describe the structure of a multi-agent monitoring system and its agents, that are considered in the works [2–4].

Statement of basic materials

Analysis of the principles of building modern monitoring systems for distributed computer systems

Today, there are many developed monitoring systems, for example, Nagios system - a system designed to monitor computer systems and networks. It scans the specified nodes and services, and notifies the administrator in case some of the services stop (or resume) their work [5]. Fig. 1 shows the system architecture.

Fig. 1. Nagios system architecture [6]

The disadvantages of such a system are:

- poor scalability;
- large interval between parameter measurements;
- data averaging (it is impossible to determine the exact value of the parameters, for example, a month ago);
- lack of automated expert data analysis tools;

Another example of such systems is the Zabbix product, designed to monitor and track the status of a variety of computer network services, servers and network equipment.

Zabbix is a stable and reliable monitoring system with a stable development speed. The server works with a single database and regardless of actions, with any other resources (memory, network, CPU), at some point in time, you may encounter an I / O restriction on the disk used by the database. Zabbix is poorly prepared for a diverse environment that is managed by a configuration management system. It has built-in capabilities for low-level identification of hosts and services, but they have their own limitations and are not tied to the configuration system. The only possibility for such integration is a proprietary solution that uses the API. Zabbix logs user actions well, except for one blind spot: changes made through the API are mostly not logged. UI Zabbix is convenient and includes many features. Fig. 2 shows the overall system architecture.

Disadvantages are poor scalability, low fault tolerance and lack of data analysis tools.

Fig. 2. Architecture of the Zabbix system [5]

The monitoring system proposed by the author of [7], using a multi-agent approach, solves the problems of scalability and fault tolerance. The author suggests creating agents that are dispersed across computing nodes and collect data on system performance.

Fig. 3 shows the proposed system architecture, which consists of 3 key components:

- 1) work manager;
- 2) work monitor;
- 3) task logs.

The disadvantage of such monitoring is that only

computing nodes are considered. As a result of data analysis, it is carried out only for them and this allows attackers to conduct an attack not on the system nodes, but on its network.

Fig. 3. System architecture [7]

The article [8] proposes a dynamic reconfigurable distributed modular monitoring system. Together with the agent approach is proposed to use the statistical data collection of those RCC systems where, for some reason, it is not possible to create an agent. The structure of this system is shown in Fig. 4.

Fig. 4. System architecture [8]

The disadvantage of such system is the non-standardized data obtained from different sources, which leads to the complexity of their intellectual analysis.

In [9], a tool complex was proposed for the meta-monitoring distributed computing environments.

This monitoring system implements an architecture that includes the following main components:

- user access tools that allow you to interact with

the meta-monitoring system in both batch and interactive modes;

- access level subsystems that control the access to the data that the user requests and implement the server side of the graphical user interface;
- op-level agent, operates in the central site of the CSW and performs the main task of managing the meta-monitoring system;
- middle-level agents, operate in intermediate nodes and solve tasks to reduce the load on upper-level agents;
- lower-level agents, operate in CSN nodes and collect and process data on the state of nodes;
- decentralized data storage subsystem, provides functions for working with data for agents of various levels.

Its disadvantage is that an expert system developed using the CLIPS shell is used for data mining. There is no self-learning mechanism in this expert system. This leads to the fact that the system cannot detect new anomalies. The author of [10], together with Nagios, uses neural networks to determine anomalies, solves the learning problem, but does not solve other shortcomings of this system. Each of these deficiencies can lead to problems with operational safety. The main goal of the work is the creation of such a monitoring subsystem, which, based on the data on the operation of the RCC, could assess the integrity of the functioning of the RCC computing process. This is the reason for the need for research in this area.

The structure of a multi-agent system for collecting and storing information

It is based on the transition from passive entities, which are described as classes of objects, to active, which are described as agents or models of an individual with activity, autonomous behavior, which also can make decisions in accordance with a certain set of rules, can interact with the environment and others agents, is able to act in the interests of achieving his goals, and may also change (evolve).

When designing the MAS, the FIPA standard was used, which regulates its general architecture, agent interaction methods and agent life cycle.

AMS (Agent Management System) – the agent who manages the entire platform. This agent is necessary for the interaction of platform agents. It provides agents access to the platform's white pages service and manages their life cycle. AMS provides various platform operations, such as: creating and removing agents, removing containers, shutting down the platform. Each agent must register with AMS to receive a personal AID. The start of the AMS agent occurs inside the main container of the platform, but can exist in any container. Agent performing activities related to the functioning of the platform, at first has to ask for confirmation from the AMS agent [5].

DF (Directory Facilitator) – the agent who provides registration of services and the search for a service agent in the yellow pages. Platform agents can subscribe to the agent DF to receive information on registration of the required service [6].

Under this standard, request agents (Ag_{select}) will perform the functions of AMS and DF, allowing agents to interact with each other. This service is obliged to provide agents with at least the “white pages” service, where agents can find out the addresses of other agents.

To describe a multi-agent system for collecting and storing data, many abstract agents are defined. So, definition of a multi-agent system is as follows:

$$MAC = (Ag_{abstr}, Ag_{select}), \quad (1)$$

where $Ag_{abstr} \in \{Ag\}$ formula (1) of article [2] – is the set of abstract agents $\{Ag_1, \dots, Ag_4\}$, instances of switch agents, network agents, session agents, and application agents that can be dynamically included in the system.

$Ag_{select} \in \{Ag\}$ formula (1) of [2] – is a specialized agent that implements registration of services and agent search, also manages the interaction of platform agents and provides agents with access to the platform's white page service and manages their life cycle.

The objects of the finite set (1) form the MAC. The solution to any problem within the framework of the MAC begins with the initialization of one of the agents of the system:

$$MAC_{init} = (Ag_{init}, Ag_{select_{init}}), \text{ where } Ag_{init} = \{Ag_1^1, \dots, Ag_{h_1}^1, Ag_1^4, \dots, Ag_{h_n}^4\}, h_1, \dots, h_n \in \mathbb{N} \quad (2)$$

$$\text{and } \forall Ag_i^j \in Ag_{init} : Ag^i \Leftrightarrow Ag_i^j \wedge Ag^i \in Ag_{abstr}.$$

The agent Ag_i^j inherits the behavior and all the initial knowledge of the prototype agent, and may also have some additional properties or knowledge (for example, a unique identifier that allows other agents of the multi-agent system to be distinguished from others and use it as an address for communication).

The MAC environment is open and there are agents in it that are already initialized while the first agent was running. So, agents from the set Ag_{init} can be supplemented by agents that already existed in the open environment considered above.

From here, the model can be complemented by many similar existing agents:

$$Ag_{existent} = \{Ag_1^1, \dots, Ag_{h_1}^1, Ag_1^m, \dots, Ag_{h_n}^m\}, \quad (3)$$

$$Ag^m \notin \{Ag\} \wedge Ag^i \in Ag_{abstr}.$$

The programmer who develops Ag_{abstr} will need information about $Ag_{existent}$, since Ag_{abstr} will use the services of these agents.

MAC work is based on agent interactions. New agents can be introduced into the system at any time, and old ones removed from it.

Each agent has a unique address that can be registered by the *Agselect* agent and become available to all MAC agents. *Agselect* agents address is known to all agents right after it was created.

The software implementation of a multi-agent data collection and storage system is a system that is usually presented as a single environment, consisting of autonomous modules with decentralized control. The main goal of such model is to get the benefits which give decentralized management: flexibility, resistance to negative environmental impact and efficient use of available resources.

Agent Interaction Protocol

The agent has the ability to provide one or more services within the framework of a unified and integrated executive model, which may include access to external software, users, and communication capabilities of the system.

The messaging service is one of the main elements of the multi-agent system architecture. The service is based on asynchronous messaging. Each agent has its own “mailbox” – an incoming message queue, where all messages sent to the agent are placed. At the moment when the message is placed in the incoming message queue, the agent is notified about it.

Consider the purpose of the components of a multi-agent data acquisition system. The request agent is the interface between the operator \ administrator, the data warehouse and network nodes with running instances of information collection agents. Instances of collection agents of various levels are the interface between the request agent and sensors for monitoring parameters. This is a clear architecture with a client-server model of interaction. In this system, the roles of the client and server are somewhat blurred.

All agents in the system operate at the 7th level of the OSI model, that is, the application level service. The interaction between request agents and collection agents is organized using Protocol Data Messages (PDM) packets that are encapsulated in a transport protocol. At the same time, each PDM message contains a specific command (to read data from the intermediate storage, change the configuration of the collection agent, or the respond with a generated collection agent). In general, the interaction of system agents can be represented by the following sequence: request agent -> PDM -UDP / TCP-IP-Ethernet-IP-UDP / TCP-PDM -> collection agent. PDM data can be encrypted to ensure reliability.

For the exchange between the agent and the manager, a certain set of PDM command messages is used:

- Trap – a one-way message from the collection agent to the request agent about an event. For example, deviations of a monitoring parameter from a certain range.
- GetReponse – a response from the collection agent to the request agent, which returns information about the data that was requested.
- GetRequest – a collection agent request from to a request agent, which is used to obtain a value from the intermediate data warehouse.
- SetRequest – a request to the agent to configure the parameters of the agent itself.
- GetBulkRequest – a request to the agent for synchronization of data of the intermediate data warehouse with the main database.
- InformRequest – the one-way communication between request agents. For example, it can be used to exchange information about collection agents that are controlled by the request agent. In response, the request agent generates a packet of confirmation that the output is received.

All PDMs (except for Trap) consist of a certain set of fields, into which the necessary information is written, it is shown in Fig. 5.

Fig. 5. Structure PDM package

The fields of the PDM packet have the following meaning:

- PDM type has a digital request identifier (Get, Set, Response, Trap);
- request identifier is the same character set that connects the request and response into a single whole;
- query string - sql similar string which helps to select or establish the corresponding data.

Some common PDM protocol features to consider. Primarily, the receiving side is trying to decode the messages. If the receiving side fails to decode the PDM, then the packet is discarded without any action. Next, the message is processed. If necessary, a JSON response is generated with the specified request identifier.

After analyzing the structure of the multiagent subsystem for collecting and processing data and agent functions, the agent architecture was developed Fig. 6.

Fig. 6. Agent Architecture

The control module configures the parameters of the agent and data transfer from the intermediate database to the central database. It also authenticates the interaction subject. The data processing and standardization module receives parameters from a specific data source that characterizes the state of the RCC component, converts them into the form necessary for storage in an intermediate database. The received data is transmitted at the input to the analysis module and recorded in an intermediate database.

The analysis module, depending on the type of agent, evaluates the functioning state of the corresponding resource and, if necessary, transfers information to the event generation module.

Depending on the type of agent and settings, the event module can submit information about the event, send an ICMP packet that informs the attacking host about the unavailability of the host, network or service.

The message generation module, depending on the request, generates a message containing settings, messages that invite certain information or a message that initiates the process of transferring data from an intermediate database to the central database.

The message processing module, depending on the received message, transfers the received settings of the

control module, transmits a request to the control module to send events or ordered data in response to the request.

The encryption module is used to encrypt messages exchanged between agents.

The authentication module is used to identify and authenticate other agents.

Conclusion

The article presents a model of the process of multi-agent system functioning for monitoring anomalies in the functioning of RIS components, which details the process of monitoring anomalies in the functioning of RIS and allows you to integrate the monitoring system with other components of the cyber security systems.

The model of a monitoring system designed to solve the problems of collecting and storing parameters received from sensors is characterized not only by their target functions, but also by functional capabilities that ensure the implementation of target functions, hierarchy and degree of parallelism in solving problems, uniformity or heterogeneity of the modular structure, organization of information collection in real-time data processing and network exchange of information with subscribers.

References

1. Smelyakov, K., Pribyl'nov, D., Martovytskyi, V. and Chupryna, A. (2018), Investigation of network infrastructure control parameters for effective intellectual analysis, *IEEE 14th International Conference on Advanced Trends in Radio-electronics, Telecommunications and Computer Engineering (TCSET)*, 20-24 Feb. 2018, pp. 983-986. <http://dx.doi.org/10.1109/TCSET.2018.8336359>.

2. Ruban, I., Lukova-Chuyko, N. and Martovitsky, V. (2016), Development of a monitoring model for cluster supercomputers, *East European Journal of Advanced Technology*, Vol. 6, No. 2(84), pp. 32-37. <http://dx.doi.org/10.15587/1729-4061.2016.85433>.
3. Ruban, I., Lukova-Chuyko, N. and Martovitsky, V. (2018), An approach to classifying the network state based on statistical parameters for detecting anomalies in the information structure of a computer system, *Cybernetics and systems analysis*, Vol. 54, No. 2, pp. 142-150. <http://dx.doi.org/10.1007/s10559-018-0032-1>.
4. Martovitsky, V. and Ruban, I. (2017), "Modelj muljtyaghennoji systemy zboru ta zberighannja informacii" [Model of a multi-agent system for collecting and storing information], *Control, navigation and communication systems*, Vol. 6 (46), PNTU, Poltava, pp. 150-153.
5. Kora, A. and Soidridine, M. (2012), Nagios based enhanced IT management system, *International Journal of Engineering Science and Technology (IJEST)*, Vol. 4, No. 4, pp. 1199-1207.
6. Knopová, M. (2014), The Third World War? In The Cyberspace Cyber Warfare in the Middle East, *Acta Informatica Pragensia*, Vol. 3, No. 1, pp. 23-32.
7. Cigala, V. (2011), Job-Oriented Monitoring of Clusters, *International Journal on Computer Science and Engineering*, Vol. 3, No. 3, pp. 56-60.
8. Stefanov, K. (2015), Dynamically Reconfigurable Distributed Modular Monitoring System for Supercomputers (DiMMon), *Procedia Computer Science*, Vol. 66, pp. 625-634.
9. Sidorov, I. Oparin, G. and Skorov, V. (2014), "Instrumental'nyj kompleks metamonitoringa raspredelennyh vychislitel'nyh sred" [Instrumentation complex for metamonitoring distributed computing environments], *Parallel'nye vychislitel'nye tekhnologii*, pp. 159-167.
10. Tarasov, A. (2010), Integration of computing cluster monitoring system, *Proc. of the First Russia and Pacific Conference on Computer Technology and Applications (RPC 2010)*, pp. 221-224.
11. Michael, J. (2009), *Wooldridge. An Introduction to Multi-Agent Systems*, Chichester: John Wiley & Sons Ltd, 461 p.
12. Zinoviev, P. and Moiseev, V. (2000), Functional models of corporate systems with multifaceted intranet architecture, *Studies in Informatics*, Vol. 2, No. 4, pp. 13-24.

Список літератури

1. Smelyakov K. Investigation of network infrastructure control parameters for effective intellectual analysis / K. Smelyakov, D. Pribyl'nov, V. Martovytskyi, A. Chupryna // IEEE 14th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), 20-24 Feb. 2018. – P. 983-986. <http://dx.doi.org/10.1109/TCSET.2018.8336359>.
2. Ruban I. Designing a monitoring model for cluster super-computer / I. Ruban, V. Martovytskyi, N. Lukova-Chuiko // Eastern-European Journal of Enterprise Technologies. – 2016. – № 84. – P. 32-37. DOI: <http://dx.doi.org/10.15587/1729-4061.2016.85433>.
3. Ruban I. Approach to Classifying the State of a Network Based on Statistical Parameters for Detecting Anomalies in the Information Structure of a Computing System / I. Ruban, V. Martovytskyi, N. Lukova-Chuiko // Cybernetics and Systems Analysis. – 2018. – №54. – P. 302-309. DOI: <http://dx.doi.org/10.1007/s10559-018-0032-1>.
4. Мартовицький В.О. Модель мультиагентної системи збору та зберігання інформації / В.О. Мартовицький, І.В. Рубан // Системи управління, навігації та зв'язку. – Полтава: ПНТУ, 2017. – Т. 6(46). – С. 150-153.
5. Kora A.D. Nagios based enhanced IT management system / A.D. Kora, M.M. Soidridine // International Journal of Engineering Science and Technology (IJEST). – 2012. – Т. 4, № 4. – P. 1199-1207.
6. Knopová M. The Third World War? In The Cyberspace. Cyber Warfare in the Middle East / M. Knopová // Acta Informatica Pragensia. – 2014. – Vol. 3, № 1. – С. 23-32.
7. Cigala V. Job-Oriented Monitoring of Clusters / V. Cigala // International Journal on Computer Science and Engineering. – 2011. – Т. 3, № 3. – P. 56-60.
8. Stefanov K. Dynamically Reconfigurable Distributed Modular Monitoring System for Supercomputers (DiMMon) / K. Stefanov // Procedia Computer Science. – 2015. – Т. 66. – P. 625-634.
9. Сидоров И.А. Инструментальный комплекс метамониторинга распределенных вычислительных сред / И.А. Сидоров, Г.А. Опарин, В.В. Скоров // Параллельные вычислительные технологии. – 2014. – С. 159-167.
10. Tarasov, A.G. Integration of computing cluster monitoring system / A.G. Tarasov // Proc. of the First Russia and Pacific Conference on Computer Technology and Applications (RPC 2010). – 2010. – С. 221-224.
11. Michael J. Wooldridge. An Introduction to Multi-Agent Systems / J. Michael. – Chichester: John Wiley & Sons Ltd, 2009. – 461 p.
12. Zinoviev P.A. Functional models of corporate systems with multifaceted intranet architecture / P.A. Zinoviev, V.S. Moiseev // Studies in Informatics. – 2000. – Vol. 2. – № 4. – P. 13-24.

Received by Editorial Board 21.02.2020

Signed for Printing 14.04.2020

Відомості про авторів:

Рубан Ігор Вікторович

доктор технічних наук професор
Перший проректор
Харківського національного університету
радіоелектроніки,
Харків, Україна
<https://orcid.org/0000-0002-4738-3286>

Information about the authors:

Igor Ruban

Doctor of Technical Sciences Professor
the First Vice-Rector
of Kharkiv National University
of Radio Electronics,
Kharkiv, Ukraine
<https://orcid.org/0000-0002-4738-3286>

Мартовицький Віталій Олександрович

кандидат технічних наук
старший викладач
Харківського національного університету
радіоелектроніки,
Харків, Україна
<http://orcid.org/0000-0003-2349-0578>

Лукова-Чуйко Наталія Вікторівна

доктор технічних наук доцент
професор Київського національного університету
ім. Тараса Шевченка,
Київ, Україна
<https://orcid.org/0000-0003-3224-4061>

Кортяк Єлизавета Юрївна

студентка
Харківського національного університету
радіоелектроніки
Харків, Україна
<https://orcid.org/0000-0002-1402-7955>

Кругліков Єгор Олександрович

аспірант
Харківського національного університету
радіоелектроніки,
Харків, Україна
<https://orcid.org/0000-0003-2977-9727>

Vitalii Martovytskyi

Candidate of Technical Sciences
Senior Instructor
of Kharkiv National University
of Radio Electronics,
Kharkiv, Ukraine
<https://orcid.org/0000-0003-2349-0578>

Nataliia Lukova-Chuiko

Doctor of Technical Sciences Associate Professor
Professor of Taras Shevchenko
National University of Kyiv,
Kyiv, Ukraine
<https://orcid.org/0000-0003-3224-4061>

Yelizaveta Kortyak

Student
of Kharkiv National University
of Radio Electronics,
Kharkiv, Ukraine
<https://orcid.org/0000-0002-1402-7955>

Yegor Kruglikov

Doctoral Student
of Kharkiv National University
of Radio Electronics,
Kharkiv, Ukraine
<https://orcid.org/0000-0003-2977-9727>

**МОДЕЛЬ МОНИТОРИНГУ МЕРЕЖНОЇ ІНФРАСТРУКТУРИ
НА ОСНОВІ СТАНДАРТУ FIPA**

В.О. Мартовицький, І.В. Рубан, Н.В. Лукова-Чуйко, Є.Ю. Кортяк, Є.О. Кругліков

Статтю присвячено проблемі створення систем моніторингу процесу функціонування мережної інфраструктури розподілених комп'ютерних систем. Для реструктуризації даних моніторингу потрібно гармонізація інформації, яка може бути приведена до єдиного представлення, або розширення структур даних шляхом додавання унікальних ознак, що дозволяють зіставляти дані з різних джерел. Так чи інакше, ця задача дуже трудомістка, особливо для віддалених один від одного сховищ, так як вимагає і технічної, і організаційної взаємодії від джерел. Крім того, після реструктуризації даних буде потрібна якісна модифікація засобів доступу до інформації з урахуванням нових структур даних. У зв'язку з цим пропонується модель системи збору і зберігання даних, що забезпечує роботу з множиною різномірних джерел, шляхом їх інтегрування з метою отримання більш повного збору інформації, що зв'язана. Система заснована на мультиагентному підході і дозволяє не припиняти обробку запитів при виконанні модифікацій набору і структур, які використовуються в базах даних. Була сформована загальна структура системи моніторингу, яка дозволяє оцінити не тільки стан окремих компонентів розподілених комп'ютерних систем, а і стан функціонування розподілених комп'ютерних систем в цілому. Представлено модель процесу функціонування мультиагентної системи моніторингу аномалій функціонування компонентів розподілених комп'ютерних систем на основі стандарту FIPA, що деталізує процес моніторингу аномалій в функціонуванні розподілених комп'ютерних систем і дозволяє інтегрувати систему моніторингу з іншими компонентами системи кібернетичного захисту. При проектуванні МАС використовувався стандарт FIPA, що регламентує її загальну архітектуру, способи взаємодії агентів, життєвий цикл агента. Модель системи моніторингу призначена для вирішення завдань збору та зберігання параметрів отриманих від датчиків, що характеризуються не тільки їх цільовими функціями, але і функціональними можливостями, що забезпечують реалізацію цільових функцій, ієрархією і ступенем паралелізму вирішення завдань, однорідністю або різномірністю модульної структури, організацією збору інформації в режимі реального часу, обробки даних і мережного обміну інформацією з абонентами.

Ключові слова: мультиагентна система, агент, розподілені комп'ютерні системи, мережна інфраструктура, система моніторингу, кібербезпека.

**МОДЕЛЬ МОНИТОРИНГА СЕТЕВОЙ ИНФРАСТРУКТУРЫ
НА ОСНОВЕ СТАНДАРТА FIPA**

В.А. Мартовицкий, И.В. Рубан, Н.В. Лукова-Чуйко, Е.Ю. Кортяк, Е.А. Кругликов

Статья посвящена проблеме создания систем мониторинга процесса функционирования сетевой инфраструктуры распределенных компьютерных систем. Представленная модель процесса функционирования мультиагентной системы мониторинга аномалий функционирования компонентов распределенных компьютерных систем детализирует процесс мониторинга аномалий в функционировании распределенных компьютерных систем и позволяет интегрировать систему мониторинга с другими компонентами системы кибернетической защиты.

Ключевые слова: мультиагентная система, агент, распределенные компьютерные системы, сетевая инфраструктура, система мониторинга, кибербезопасность.