ХМЕЛЬЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
МІНІСТЕРСТВА ОСВІТИ І НАУКИ УКРАЇНИ

Кваліфікаційна наукова праця
на правах рукопису

ДРОЗД АНДРІЙ ІГОРОВИЧ

УДК 004.3:004.75:004.49:004.52

ДИСЕРТАЦІЯ

МЕТОДИ ТА СИСТЕМИ ВИЯВЛЕННЯ КОМП'ЮТЕРНИХ АТАК В КОРПОРАТИВНИХ МЕРЕЖАХ НА ОСНОВІ ПОПУЛЯЦІЙНИХ АЛГОРИТМІВ

123 Комп'ютерна інженерія

12 Інформаційні технології

Подається на здобуття наукового ступеня доктора філософії
Дисертація містить результати власних досліджень. Подані до захисту наукові положення є власним напрацюванням. Всі використані ідеї, наукові результати, цитати супроводжуються належними посиланнями на їх авторів та джерела опублікування. Всі частини тексту дисертації, під час написання яких використовувались технології штучного інтелекту, перевірені та відредаговані особисто.

_____ А.І. Дрозд
(підпис)

Наукові керівники:
Савенко Олег Станіславович, доктор технічних наук, професор
Коробчинський Максим Володимирович, доктор технічних наук, професор

Хмельницький – 2026

АНОТАЦІЯ

Дрозд А.І. Методи та системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів. – Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії з галузі знань 12 Інформаційні технології за спеціальністю 123 Комп'ютерна інженерія. – Хмельницький національний університет, Хмельницький, 2026.

У роботі подано результати дослідження, спрямованого на підвищення ефективності протидії зловмисним діям у корпоративних мережах шляхом удосконалення архітектури та методів функціонування обманних систем з приманками і пастками (ОСПП). Запропоновано оновлену архітектуру таких систем, у якій синтезовано популяційні алгоритми, зокрема алгоритм молі й полум'я, що забезпечує оптимізацію формування послідовностей кроків у процесі реалізації комп'ютерних атак і дій зловмисного програмного забезпечення. Завдяки цьому досягається уникнення повного перебору можливих варіантів, прискорення збіжності рішень під час динамічних змін у середовищі корпоративної мережі та врахування потенційної здатності зловмисників здійснювати двоцільові атаки.

У дисертації здійснено аналіз архітектури сучасних обманних систем, методів їх організації функціонування, методів виявлення комп'ютерних атак та типів популяційних алгоритмів, які можуть бути синтезовані в архітектурі ОСПП. В роботі представлено архітектуру ОСПП, в якій синтезовано алгоритм молі і полум'я для покращення їх функціонування під час атак, моделі двоцільових комп'ютерних атак, метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі ОСПП, метод організації функціонування ОСПП в корпоративних мережах, метод виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників, а також розроблено відповідну обманну систему, здійснено постановку експериментів та проведені дослідження із розробленою системою.

Об'єктом дослідження є процес організації обманних систем з приманками і пастками для виявлення комп'ютерних атак та зловмисного програмного забезпечення в корпоративних мережах.

Предметом дослідження є методи організації обманних систем з приманками і пастками для виявлення комп'ютерних атак та зловмисного програмного забезпечення в корпоративних мережах.

Метою дисертаційного дослідження є покращення протидії комп'ютерним атакам та зловмисному програмному забезпеченню в корпоративних мережах шляхом оптимізації кроків обманних систем з приманками і пастками за рахунок синтезу популяційних алгоритмів в центрах прийняття рішень.

Наукова новизна отриманих результатів полягає в наступному:

1) удосконалено архітектуру обманних систем з приманками і пастками, в якій на відміну від відомих варіантів архітектури, здійснено синтез популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміни послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових КА;

2) розроблено новий метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, який, на відміну від відомих, характеризується формуванням дискретного простору пошуку з координатним поданням об'єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі й полум'я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале й адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

3) розроблено новий метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах, в якому на відміну від відомих, в архітектурі обманних систем синтезовано популяційні алгоритми, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак, що дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих

впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж та ускладнює дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп'ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах;

4) розроблено новий метод виявлення атак відмови в обслуговуванні у мережах на основі статистичних показників, який на відміну від відомих, базується на обчисленні статистичних ознак мережного IP-трафіку при розбитті потоку пакетів на часові вікна, і встановлює динамічні зміни трафіку на рівні всього аналізованого періоду, що дозволяє підвищити достовірність виявлення атак відмова в обслуговуванні.

Практичне значення отриманих результатів. Розроблено обманну систему з приманками і пастками для виявлення КА та ЗПЗ в корпоративних мережах, особливістю якої є прийняття в ній рішень щодо наступних кроків та їх коригування з використанням алгоритму дискретної оптимізації молі і полум'я, а також імплементацією в її компонентах методу виявлення комп'ютерних атак на основі аналізу їх статичних показників.

Синтез популяційних алгоритмів в архітектурі оманних систем для прийняття ними рішень дав змогу формувати послідовності кроків систем так, щоб заплувати зловмисників при проведенні КА. Також, в процесі синтезу алгоритму молі і полум'я в архітектуру обманних систем було здійснено розроблення його кроків та адаптації для реалізації саме для задач дискретної оптимізації, що є основою для здійснення аналогічних кроків в процесі деталізації інших популяційних алгоритмів натхненних живою природою.

За результатами проведених експериментальних досліджень встановлено, що розроблена ОСПП забезпечує коректне функціонування в умовах динамічної зміни оточуючого середовища корпоративних мереж, ефективне залучення приманок і пасток для виконання задач виявлення інфікованих програм, а також вибір наступних кроків для виконання.

Теоретичні та практичні результати дослідження впроваджені в ТОВ «Nolt technologies» (м. Хмельницький, Акт від 16.02.2026), ТОВ «ІТТ» (м. Хмельницький Акт від 16.02.2026), а також, в освітньому процесі Хмельницького національного

університету (Акт від 25.02.2026) при викладанні дисциплін на кафедрі комп'ютерної інженерії та інформаційних систем для здобувачів спеціальності F7 Комп'ютерна інженерія, зокрема в курсах «Безпека та захист комп'ютерних систем», «Моделювання та методи оптимізації в наукових та експериментальних дослідженнях», «Методології забезпечення якості, надійності, гарантоздатності та безпеки комп'ютерних систем та мереж» та в освітній процес у блоці військово-спеціальних дисциплін другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка (Акт від 18.12.2025), які використані при удосконалені навчально-лабораторного комплексу.

У вступі представлено обґрунтування актуальності наукової задачі, що полягає у підвищення ефективності протидії зловмисним діям у корпоративних мережах шляхом удосконалення архітектури та методів функціонування ОСПП. Важливим напрямком досліджень визначено обманні системи, рішення в яких формуються на основі популяційних алгоритмів. Продемонстровано зв'язок тематики дослідження з напрямками наукових досліджень науковців з цієї тематики, представлено основні наукові результати роботи, їх практичне використання, перелік підприємств та установ, в яких впроваджено результат роботи.

У першому розділі проведено аналіз предметної області дослідження, існуючих обманних систем, приманок і пасток, методів, що застосовуються для виявлення комп'ютерних атак та зловмисного програмного забезпечення. Проаналізовано типи популяційних алгоритмів та їх особливості.

У другому розділі розроблено моделі двоцільових атак на корпоративні мержі та запропоновано удосконалену архітектуру ОСПП, в якій на відміну від відомих варіантів архітектури, здійснено синтез популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових КА;

У третьому розділі представлено розроблений метод синтезу дискретного алгоритму молі й полум'я, який відрізняється формуванням дискретного простору пошуку, координатним поданням об'єктів, використанням секторного оцінювання

потенційних кроків, побудовою спірального сліду та урахуванням часу як параметра зміни кроків. Запропоноване динамічне переміщення молі та полум'я дозволяє уникати передчасної збіжності до локальних оптимумів і забезпечує довготривале адаптивне функціонування обманних систем за рахунок раціонального вибору послідовностей кроків під час опрацювання подій. Також, представлено розроблений метод організації функціонування обманних систем у корпоративних мережах, у межах якого популяційні алгоритми забезпечують автономний вибір подальших дій обманних компонентів з урахуванням змін у мережному середовищі. Це дає змогу ускладнювати діяльність зловмисників, запобігати реалізації двоцільових атак, формувати рішення без повного перебору варіантів та автоматично активувати або блокувати сервери чи станції залежно від характеру виявлених впливів.

У четвертому розділі представлено розроблений метод виявлення атак відмова в обслуговуванні, який ґрунтується на аналізі статистичних характеристик мережного трафіку, що формується при поділі потоку пакетів на часові вікна. Метод забезпечує відстеження динаміки змін трафіку на рівні всього досліджуваного періоду та підвищує достовірність і оперативність виявлення атак типу Denial of Service. Також, описано розроблені програмні частини обманних систем, особливості їх реалізації та використані бібліотеки для їх реалізації, що забезпечують працездатність реалізованої системи. Представлено проведені експерименти, оцінювання ефективності обманної системи, проведено аналіз з отриманих результатів.

У висновках представлено отримані наукові та практичні результати проведеного дослідження.

У додатках представлено наукові публікації, в яких відображено наукові результати роботи, акти впровадження результатів роботи, лістинг розробленого програмного забезпечення.

Ключові слова: комп'ютерні системи; корпоративні мережі; комп'ютерні станції; обманні системи; популяційні алгоритми; алгоритм молі і полум'я; пастка; приманка; зловмисне програмне забезпечення; комп'ютерні атаки; системи виявлення вторгнень; архітектура систем.

ANNOTATION

Drozd A.I. Methods and systems for detecting computer attacks in corporate networks based on population algorithms. – Qualifying scientific work on manuscript rights.

Dissertation for obtaining the scientific degree of Doctor of Philosophy in the field of knowledge 12 Information technologies in the specialty 123 Computer engineering. – Khmelnytskyi National University, Khmelnytskyi, 2026.

The paper presents the results of a study aimed at increasing the effectiveness of countering malicious actions in corporate networks by improving the architecture and methods of operation of deceptive systems with baits and traps. An updated architecture of such systems is proposed, in which population algorithms are synthesized, in particular, the moth and flame algorithm, which ensures optimization of the formation of sequences of steps in the process of implementing computer attacks and actions of malicious software. Thanks to this, it is possible to avoid a complete search of possible options, to accelerate the convergence of decisions during dynamic changes in the corporate network environment, and to take into account the potential ability of attackers to carry out dual-purpose attacks.

The dissertation analyzes the architecture of modern deception systems, methods of organizing their operation, methods of detecting computer attacks, and types of population algorithms that can be synthesized in the architecture of deception systems with decoys and traps. The work presents the architecture of deception systems with baits and traps, in which the algorithm of moth and fire is synthesized to improve their functioning during attacks, models of dual-target computer attacks, the method of synthesis of the algorithm of discrete optimization of moths and flames in the architecture of deception systems with baits and traps, the method of organizing the functioning of deception systems with baits and traps in corporate networks, the method of detecting denial of service attacks in networks based on statistical indicators, as well as a suitable cheating system was developed, experiments were set up and research was conducted with the developed system.

The object of the study is the process of organizing deceptive systems with baits and traps to detect computer attacks and malicious software in corporate networks.

The subject of the study is methods of organizing deceptive systems with baits and traps for detecting computer attacks and malicious software in corporate networks.

The aim of the dissertation research is to improve countermeasures against computer attacks and malicious software in corporate networks by optimizing the steps of deception

systems with decoys and traps due to the synthesis of population algorithms in decision-making centers.

The scientific novelty of the obtained results is as follows:

1) the architecture of deception systems with decoys and traps has been improved, in which, in contrast to known architectural options, the synthesis of population algorithms, in particular the moth and fire algorithm, has been carried out to optimize the formation of the sequence of the next steps in the implementation of KA and malicious software actions, avoiding a complete search of options, rapid convergence of the selected steps with ongoing influences and changing the sequence of steps taking into account current changes in the surrounding environment of corporate networks, as well as taking into account the potential ability of attackers to carry out dual-purpose KA;

2) a new method of synthesis of the discrete moth and flame algorithm in the architecture of decoy systems with decoys and traps has been developed, which, unlike the known ones, is characterized by the formation of a discrete search space with a coordinate representation of objects, the synthesis of a spiral trail based on the sectoral evaluation of potential steps and angular characteristics, taking into account time as a parameter of step change and dynamic movement of the moth and flame to avoid premature convergence to local optima, which made it possible to develop deceptive systems that ensure long-term and adaptive functioning in the process of countering intruders in corporate networks by changing the steps for processing events;

3) a new method of organizing the functioning of deception systems with baits and traps in corporate networks has been developed, in which, unlike the known, the architecture of deception systems synthesizes population algorithms, in particular the moth and fire algorithm, for their selection of the next steps to avoid the implementation of dual-purpose attacks by criminals, which makes it possible to avoid a complete selection of options from possible steps, rapid convergence of the selected steps with ongoing influences and a change in the sequence of steps taking into account current changes in the surrounding environment of corporate networks and complicates actions due to decision-making based on population algorithms with the ability to independently block or activate servers or computer stations, baits or traps during the establishment of potentially malicious influences in corporate networks;

4) a new method of detecting denial-of-service attacks in networks based on statistical indicators has been developed, which, unlike the known ones, is based on the calculation of statistical characteristics of network IP traffic when dividing the flow of packets into time windows, and establishes dynamic changes traffic at the level of the entire analyzed period, which allows to increase the reliability of detection of denial of service attacks.

Practical significance of the obtained results. A deceptive system with decoys and traps has been developed for the detection of KA and malicious software actions in corporate networks, the feature of which is decision-making in it regarding the next steps and their adjustment using the algorithm of discrete optimization of moths and flames, as well as the implementation in its components of the method of detecting computer attacks based on the analysis of their static indicators.

The synthesis of population algorithms in the architecture of deceptive systems for their decision-making made it possible to form a sequence of steps of the systems in such a way as to overwhelm the attackers during the execution of the KA. Also, in the process of synthesizing the moth and flame algorithm into the architecture of deceptive systems, its steps were developed and adapted for implementation specifically for discrete optimization problems, which is the basis for implementing similar steps in the process of detailing other population algorithms inspired by living nature.

According to the results of experimental studies, it was established that the developed deception system with baits and traps ensures correct functioning in the conditions of dynamic changes in the surrounding environment of corporate networks, effective involvement of baits and traps to perform the tasks of detecting infected programs, as well as the selection of the next steps for implementation.

The theoretical and practical results of the research are implemented in "Nolt technologies" LLC (Khmelnytskyi, act dated 18.02.2026), ITT LLC (Khmelnytskyi, act dated 16.02.2026), as well as in the educational process of the Khmelnytskyi National University (Act dated 25.02.2026) in the teaching of disciplines at the Department of Computer Engineering and Information Systems for students of the F7 Computer Engineering specialty, in particular in the courses "Security and protection of computer systems", "Modeling and optimization methods in scientific and experimental research", "Methodologies for ensuring the quality, reliability, warranty and security of computer systems and networks" and in the educational process in the block of military special

disciplines of the second department of the Second Educational and Scientific Institute of the Yevgeny Bereznyak Military Academy (Act dated 18.12.2025), which were used in the improvement of the educational and laboratory complex.

The introduction presents the justification of the relevance of the scientific task, which consists in increasing the effectiveness of countering malicious actions in corporate networks by improving the architecture and methods of functioning of deceptive systems with baits and traps. Deception systems, in which solutions are formed on the basis of population algorithms, are defined as an important area of research. The connection of the research topic with the directions of scientific research of scientists on this topic is demonstrated, the main scientific results of the work, their practical use, the list of enterprises and institutions in which the results of the work are implemented are presented.

In the first chapter, an analysis of the subject area of research, existing deception systems, decoys and traps, methods used to detect computer attacks and malicious software is carried out. The types of population algorithms and their features are analyzed.

In the second chapter, models of dual-target attacks on corporate targets are developed and an improved architecture of deception systems with baits and traps is proposed, in which, in contrast to known architecture options, a synthesis of population algorithms, in particular the moth and fire algorithm, is carried out to optimize the formation of the sequence of the next steps in the implementation of KA and malicious software actions, avoiding a complete search of options, rapid convergence of the selected steps with ongoing influences and changing the sequence of steps taking into account current changes in the environment of corporate networks, as well as taking into account the potential ability of attackers to carry out dual-purpose KA.

The third chapter presents the developed method of synthesis of the discrete moth and flame algorithm, which is distinguished by the formation of a discrete search space, coordinate representation of objects, the use of sectorial evaluation of potential steps, the construction of a spiral trace and taking into account time as a parameter of step change. The proposed dynamic movement of moths and flames avoids premature convergence to local optima and ensures long-term adaptive functioning of deception systems due to the rational selection of sequences of steps during event processing. Also, a developed method of organizing the functioning of deception systems in corporate networks is presented, within which population algorithms provide autonomous selection of further actions of

deception components, taking into account changes in the network environment. This makes it possible to complicate the activities of attackers, prevent the implementation of dual-target attacks, form solutions without exhaustively sorting out options, and automatically activate or block servers or stations depending on the nature of the detected impacts.

The fourth chapter presents the developed method of detecting denial of service attacks, which is based on the analysis of statistical characteristics of network traffic, which is formed when the packet flow is divided into time windows. The method provides tracking of the dynamics of traffic changes at the level of the entire investigated period and increases the reliability and efficiency of detection of Denial of Service attacks. Also, the developed software parts of cheating systems, the peculiarities of their implementation and the libraries used for their implementation, which ensure the functionality of the implemented system, are described. The conducted experiments, evaluation of the effectiveness of the cheating system, and analysis of the obtained results are presented.

The scientific and practical results of the conducted research are presented in the conclusions.

The appendices present scientific publications that reflect the scientific results of the work, acts of implementation of the results of the work, and a listing of the developed software.

Keywords: computer systems; corporate networks; computer stations; deception systems; population algorithms; moth and flame algorithm; trap; lure; malicious software; computer attacks; intrusion detection systems; systems architecture.

СПИСОК ПУБЛІКАЦІЙ ЗДОБУВАЧА ЗА ТЕМОЮ ДИСЕРТАЦІЇ

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Савенко О.С., Дрозд А.І., Медзатий Д.М. Концептуальна архітектура обманних систем з приманками і пастками на основі популяційних алгоритмів. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №84(4). С. 127-151. DOI: https://doi.org/10.31891/2219-9365-2025-84-15

2. Дрозд А. Метод виявлення комп’ютерних атак типу відмови в обслуговуванні на основі статистичних показників мережного трафіку. *Information Technology:*

Computer Science, Software Engineering and Cyber Security. 2025. № 4, С. 79–89. DOI: https://doi.org/10.32782/IT/2025-4-10

3. Савенко О.С., Дрозд А.І., Коробчинський М.В. Метод синтезу популяційних алгоритмів в архітектурі обманних систем з приманками і пастками. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №82(2). С. 459–474. DOI: https://doi.org/10.31891/2219-9365-2025-82-64

4. RAMSKYI I., DROZD A., LYHUN O., PONOCHOVNA O. SYSTEM FOR CYBERSECURITY EVALUATION OF CORPORATE NETWORKS. *Computer Systems and Information Technologies*. 2025. № 2. С. 123–131. DOI: https://doi.org/10.31891/csit-2025-2-14

5. Дрозд А.І. Метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах. *Вісник Хмельницького національного університету. Технічні науки*. 2025. № 359 (6.2). С. 445–457. DOI: https://doi.org/10.31891/2307-5732-2025-359-135

6. Savenko O., Rusyn B., Lysenko S., Ciszewski T., Savenko B., Drozd A., Nicheporuk A., Sachenko A. Synthesis of a Moth and Flame Algorithm for Incorporation into the Architecture of Deceptive Systems with Baits and Traps. *Applied Sciences*. 2026. 16(5). 2415. DOI: https://doi.org/10.3390/app16052415

Праці, які засвідчують апробацію матеріалів дисертації

7. Rehida, P., Savenko, O., Sachenko, A., Drozd, A., Vizhevski, P. A trust model that ensures the correctness of computing in grid computing system. (2024) *CEUR Workshop Proceedings*, 3675, pp. 388-401. *The 5th International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS-2024)*: CEUR-Workshop Proceedings. Vol. 3675. (Khmelnytskyi, March 2024). Khmelnytskyi, 2024. Pp. 388-401. URL: https://ceur-ws.org/Vol-3675/paper28.pdf *(Scopus)*

8. Denysiuk D., Sochor T., Kapustian M., Kashtalian A., Drozd A. A method for detecting botnets in IT infrastructure using a neural network. (2024) *CEUR Workshop Proceedings*, 3736, pp. 282-292. *The 1th Proceedings of the 1st International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS 2024)*. Khmelnytskyi, Ukraine, June 28,

2024 : CEUR-Workshop Proceedings. Vol. 3736. (Khmelnytskyi, Ukraine, June 28, 2024). Khmelnytskyi, 2024. Pp. 282-292. URL: https://ceur-ws.org/Vol-3736/paper21.pdf *(Scopus)*

9. Sierhieiev Y., Savenko O., Paiuk V., Drozd A. Effectiveness and improvement of Static Application Security Testing (SAST) in the context of SQL Injection vulnerabilities // *Proceedings of 2024 IEEE 14th International Conference on Dependable Systems, Services and Technologies* (DeSSerT-2024, Athens, Greece, October 11-13, 2024) DOI: 10.1109/DESSERT65323.2024.11122171 *(Scopus)*

10. Semeniuk B., Kashtalian A., Martiniuk D., Drozd A., Abdel-Badeeh M. Salem. Detection of computer attacks based on sonification of network traffic. *Intelitsis'25: The 6th International Workshop on Intelligent Information Technologies & Systems of Information Security*, April 04, 2025, Khmelnytskyi, Ukraine. URL: https://ceur-ws.org/Vol-3963/paper21.pdf *(Scopus)*

11. Kozelskyi O., Drozd A., Savenko B., Gaj P. A model for probabilistic monitoring and proactive restart of real-time operating systems under intensive state changes in cyber-physical systems. *Proceedings of the 2nd International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS 2025)* Khmelnytskyi, Ukraine on July 4, 2025. Pp. 198-210. URL: https://ceur-ws.org/Vol-4013/paper16.pdf *(Scopus)*

Публікації, які додатково відображають наукові результати дисертації

12. Свідоцтво про реєстрацію авторського права на твір № 142624 Україна. Комп'ютерна програма «Програмне забезпечення функціонування обманних систем в корпоративних мережах з прийняттям рішень на основі популяційних алгоритмів» / Дрозд А. І., Савенко О. С., Нічепорук А. О., Регіда П. Г. 13.02.2026.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

ЗПЗ – зловмисне програмне забезпечення

ІС – інформаційна (автоматизована) система

ІТ – інформаційні технології

КА – комп’ютерна атака

КС – комп'ютерна система

ОС – операційна система

ОСПП – обманні системи з приманками і пастками

ПЗ – програмне забезпечення

ACO (Ant Colony) – мурашинні алгоритми

CSMS (Cyber Security Management System) –система управління кібербезпекою

DE (Differential Evolution) – метод багатовимірної математичної оптимізації

IDS (Intrusion Detection System) – система виявлення вторгнень

IP (Internet Protocol) – міжмережний протокол

GA (Genetic Algorithm) – генетичний алгоритм

MFO – алгоритм молі і полум’я

PSO (Particle Swarm Optimization) – метод рою частинок

Random / Rule-based – машинне навчання на основі правил

SDN (Software-Defined Network) – програмно-конфігурована мережа

SIEM (Security Information and Event Management) – система управління інформацією та подіями безпеки

TCP/IP (Transmission Control Protocol / Internet Protocol) – протокол керування передаванням / міжмережний протокол

ВСТУП

Обґрунтування вибору теми дослідження. Комп’ютерні атаки (КА) на корпоративні мережі здійснюються зловмисниками постійно і вони постійно удосконалюються. Для протидії КА в корпоративних мережах застосовуються різноманітні засоби забезпечення безпеки та захисту в них. Але зловмисники володіють інформацією про різноманітні комерційні засоби та системи забезпечення безпеки та захисту корпоративних мереж, оскільки вони пропонуються розробниками і їх можна досліджувати. При тривалій експлуатації чи дослідженнях ці засоби та системи стають зрозумілими зловмисникам в частині їх особливостей і, тому, після удосконалення етапів проведення КА на корпоративні мережі з такими засобами чи системами безпека та захист їх суттєво знижується. Для уникнення таких сценаріїв розробники систем та засобів забезпечення безпеки та захисту корпоративних мереж постійно їх удосконалюють, щоб досягти ними непередбачуваної поведінки для зловмисників, тобто щоб зловмисники при здійсненні КА, зокрема і з використанням зловмисного програмного забезпечення (ЗПЗ), не могли швидко зрозуміти особливості функціонування таких систем та засобів. Таким чином, крім сучасних методів, які спрямовані на виявлення або протидію КА чи ЗПЗ, необхідним є постійне розроблення та удосконалення систем та засобів, в які імплементовано такі методи, в частині заплутування ними зловмисників, створення в процесі їх експлуатації недетермінованої поведінки таких засобів і систем для зловмисників, але із збереженням їх ефективності та можливості до тривалої протидії щодо КА і ЗПЗ.

Окремими класами систем та засобів забезпечення безпеки та захисту корпоративних мереж є обманні системи, приманки та пастки. Оскільки вони є доступними для користувачів і, також, зловмисників, бо є комерційними розробками, то вони теж потребують постійного удосконалення архітектури та особливостей функціонування для уникнення розуміння їх функціонування. Крім того, перспективним напрямом їх застосування є їх поєднання в цілісні ОСПП. Тоді, виникають завдання щодо забезпечення їх недетермінованого функціонування окремо в обманних системах, окремо в приманках і пастках та керуванні приманками і пастками обманними системами в корпоративних мережах.

Ученими, які проводять наукові дослідження з розроблення нових теорій, принципів, моделей, методів, систем та засобів виявлення КА та ЗПЗ, систем

спрямованих на забезпечення безпеки та захисту інформації, а також розподілених систем стійких до зловмисних впливів, є Анвар А. [1; 2], Бісвас Д. [3], Ванг К. [4], Віола В. [5], Гнатюк С. [6; 7], Дудикевич В. [8], Занг Ю. [9], Золфахарі Ш. [10], Ефенді М. [11], Карпінський М. [12], Кларк А. [13], Кемпвел Р. [14], Коваленко А. [15; 16], Корченко А. [17; 18], Корченко О. [19], Летичевський О. [20; 21], Мілов О. [17], Мухін В. [22; 23], Парк К. [24], Ров Н. [25], Сардана А. [26], Саченко А. [27-31], Сохор Т. [32-34], Харченко В. [35-37], Хофер В. [38], Хохлачова Ю. [39; 40], Фенг М. [41], Фергюсен-Вальтер К. [42; 43], Франко Д. [44], Євсєєв С. [17], Юдін О.[6] та ін.

Реалізовані в обманних системах оманні технології частково відомі. Їх можна встановити, також, за результатами аналізу комерційної інформації від розробників. Тому, актуальною науковою задачею є розроблення нових обманних технологій для їх синтезу в архітектурі обманних систем таким чином, щоб їхні наступні кроки були складно детермінованими. Ці обманні системи повинні, також, підтримувати функціонування та керування приманками і пастками в корпоративних мережах. Перспективною стратегією для розв’язування такої задачі є синтез популяційних алгоритмів в архітектурі обманних систем з метою забезпечення вибору наступних кроків систем так, щоб зловмисники при здійсненні КА не змогли зрозуміти принципи їх функціонування. Вибір популяційних алгоритмів для вибору наступних кроків обманних систем повинен забезпечити, також, підтримування тривалої за часом протидії та виявлення КА в корпоративних мережах. Частина популяційних алгоритмів дає змогу вирішувати задачу оптимізації, яка відповідає вибору наступних кроків систем. Серед популяційних алгоритмів наявні такі, що дають змогу уникати швидкої збіжності в межах локального оптимуму і, таким чином, через певний час дають змогу досягти глобального оптимуму.

Ученими, які проводять наукові дослідження в сфері популяційних алгоритмів, зокрема еволюційних алгоритмів, їх застосування, удосконалення та розроблення в фундаментальних і прикладних наукових задачах та проблемах оптимізації, є Бодянський Є. [45-47], Гуляницький Л. [48], Мірджалілі С. [49-51], Мулеса О. [48; 52; 53], Сайберфілдт А. [54], Сахо С. [55], Ротштейн О. [56; 58], Штовба С. [56-59] та ін.

Для того, щоб забезпечити недетерміновану поведінку ОСПП в їх архітектурі доцільним є реалізація підсистеми прийняття рішень щодо подальших кроків системи із можливістю їх коригування в процесі виконання визначених кроків на основі популяційних алгоритмів. Тоді, поведінка обманних систем буде визначатись

відповідно до характеристики конкретного популяційного алгоритму. Оскільки зловмисники при здійсненні КА можуть теж змінювати послідовність кроків, то найбільш відповідними їм за поведінкою можуть бути популяційні алгоритми натхненні живою природою. В їх основі реалізована поведінка отримана з спостережень за живою природою. Цей клас популяційних алгоритмів активно розвивається. Одним з таких алгоритмів є алгоритм молі і полум'я. Він характеризується забезпеченням поперечної орієнтації та тривалого руху від локального оптимуму до пошуку розв'язків в глобальному оптимумі. Тому, його можна використати в підсистемі прийняття рішень для пошуку наступних кроків ОСПП для заплування зловмисників і підтримування тривалого функціонування під час КА чи дій ЗПЗ. Але переважна більшість пропонованих дослідниками реалізацій цього алгоритму молі і полум'я стосується неперервного простору пошуку і, відповідно, неперервних функцій. Для задачі з вибору наступних кроків, зокрема і вибору приманок та пасток для певних типів КА в корпоративних мережах, простір пошуку є дискретним і, тому, реалізація алгоритму в підсистемі прийняття рішень потребує розроблення простору пошуку, кроків алгоритму та дискретної функції вибору кроків.

Таким чином, виникають протиріччя щодо удосконалення наявної архітектури чи розроблення принципово нової архітектури обманних систем, які функціонують сумісно з приманками і пастками в корпоративних мережах, в якій визначення наступних кроків системи було б з однієї сторони адекватним подіям в середовищах корпоративних мереж, що виникають під впливом КА чи ЗПЗ, а з іншої сторони вибір кроків та їх коригування повинні були б бути складно зрозумілими, тобто заплутаними, для зловмисників. А для формування наступних кроків систем з наявних кроків ускладнено відсутністю ефективних алгоритмів, виконання яких би було і адекватним щодо завдань систем, враховуючи їх специфіку, а також недетермінованим для дослідників, зокрема і зловмисників, одночасно.

Тому, важливою та актуальною науковою задачею є покращення протидії КА та ЗПЗ в корпоративних мережах шляхом оптимізації кроків ОСПП за рахунок синтезу популяційних алгоритмів в центрах прийняття рішень.

Зазначена науково-прикладна задача відповідає предметній області Стандарту вищої освіти України зі спеціальності 123 – Комп'ютерна інженерія для третього (освітньо-наукового) рівня вищої освіти, зокрема, такому об'єкту вивчення та

діяльності, як « - … цифрові комп’ютери та комп’ютерні системи, локальні, глобальні комп'ютерні мережі та мережа Інтернет, …, ІТ-інфраструктури, методи та способи подання, отримання, зберігання, передавання, опрацювання та захисту в них інформації, …, архітектура та організація їх функціонування, інтерфейси та протоколи взаємодії їх компонентів, …; - інформаційні процеси, технології, методи, способи, інструментальні засоби та системи для дослідження, … й експлуатації комп’ютерів та комп’ютерних систем і мереж, …, ІТ-інфраструктур, розроблення, верифікації та розгортання програмного забезпечення та систем у хмарних та інших середовищах, а також процедури та засоби підтримки та керування життєвим циклом, забезпечення якості, надійності та безпеки.».

Зв’язок роботи з науковими програмами, планами, темами. Дисертаційне дослідження виконувалось у рамках науково-дослідної тематики Хмельницького національного університету: держбюджетної науково-дослідної теми №2Б-2024 «Система виявлення ЗПЗ та комп’ютерних атак в корпоративних мережах з використанням хибних об’єктів атак та пасток» (номер держреєстрації 0124U000980); держбюджетної науково-дослідної теми №1Б-2026 «Система забезпечення стійкості до витоку конфіденційної інформації в корпоративних мережах в умовах впливів комп’ютерних атак» (номер держреєстрації 0126U002082), в яких автор дисертації був виконавцем.

Мета і завдання дослідження.

Об’єктом дослідження є процес організації обманних систем з приманками і пастками для виявлення комп’ютерних атак та зловмисного програмного забезпечення в корпоративних мережах.

Предметом дослідження є методи організації обманних систем з приманками і пастками для виявлення комп’ютерних атак та зловмисного програмного забезпечення в корпоративних мережах.

Метою дисертаційного дослідження є покращення протидії комп’ютерним атакам та зловмисному програмному забезпеченню в корпоративних мережах шляхом оптимізації кроків обманних систем з приманками і пастками за рахунок синтезу популяційних алгоритмів в центрах прийняття рішень.

Завдання дослідження сформульовані наступним чином:

1) провести аналіз типів КА та відповідного ЗПЗ, методів та систем їх виявлення в корпоративних мережах, зокрема обманних систем, приманок та пасток, а також

популяційних алгоритмів натхненних живою природою, які можуть бути основою для їх застосування в архітектурі обманних систем для підвищення ефективності вибору ними наступних кроків та їх коригування;

2) розробити моделі КА в корпоративних мережах, в яких врахована особливість щодо наявних в корпоративних мережах реальних та хибних об'єктів для атак та відповідних дій зловмисників, які спрямовані на класифікацію таких об'єктів, що дозволить врахувати їх поведінку в моделі функціонування ОСПП;

3) розробити формальний опис архітектури ОСПП з підсистемою прийняття рішень на основі синтезу в архітектурі популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків та їх коригування при здійсненні КА та дій ЗПЗ;

4) розробити метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі ОСПП для забезпечення їх довготривалого й адаптивного функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

5) розробити метод організації функціонування ОСПП в корпоративних мережах на основі синтезу в їх архітектурі популяційних алгоритмів, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак;

6) розробити метод виявлення атак типу відмова в обслуговуванні у мережах на основі статистичних показників з урахуванням його імплементації в компоненти ОСПП для підвищення достовірності виявлення атак відмова в обслуговуванні;

7) розробити обманну систему з приманками і пастками для виявлення двоцільових КА, зокрема КА типу відмова в обслуговуванні у мережах на основі статистичних показників, для експериментального дослідження таких систем з метою покращення її характеристик та подальшого впровадження для застосування.

Методи дослідження. Для вирішення поставлених задач використовуються основні положення абстрактної алгебри, теорії розподілених систем, теорії множин, теорії графів, методи оптимізації та популяційних алгоритмів.

1. Теорії абстрактної алгебри та теорії розподілених систем для визначення архітектури ОСПП в розподілених середовищах та деталізованого представлення їх основних елементів та компонентів.

2. Теорії множин і теорії графів для визначення компонентів ОСПП, а також процесу синтезу популяційних алгоритмів, зокрема алгоритму молі і полум'я, в архітектуру обманних систем.

3. Теорії розподілених систем для організації функціонування ОСПП та їх компонентів для виявлення комп'ютерних атак і зловмисного програмного забезпечення.

4. Методи оптимізації для розроблення алгоритму дискретної оптимізації молі і полум'я.

Обґрунтованість і достовірність наукових положень, висновків і рекомендацій. Наукові положення, висновки і рекомендації дисертації обґрунтовані коректним та доцільним використанням математичного апарату у вигляді моделей ОСПП та їх компонентів і моделей комп'ютерних атак, що є цілеспрямованими на реальні та хибні об'єкти в корпоративних мережах, реалізацією розробленої обманної системи, що складається із серверного і клієнтського програмних застосунків, які забезпечують процес організації функціонування, керування приманками і пастками та опрацювання штатних і позаштатних подій в корпоративних мережах, включно з імплементацією в їх архітектуру методу виявлення комп'ютерних атак, що базується на аналізі статистичних показників, практичним та наочним впровадженням результатів дисертаційного дослідження на підприємствах, які використовують системи для захисту корпоративних мереж і які демонструють відповідність отриманих результатів теоретичних досліджень із практичними результатами застосування.

Наукова новизна отриманих результатів полягає в наступному:

1) удосконалено архітектуру обманних систем з приманками і пастками, в якій на відміну від відомих варіантів архітектури, здійснено синтез популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміни послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових КА;

2) розроблено новий метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, який, на відміну від

відомих, характеризується формуванням дискретного простору пошуку з координатним поданням об’єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі й полум’я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале й адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

3) розроблено новий метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах, в якому на відміну від відомих, в архітектурі обманних систем синтезовано популяційні алгоритми, зокрема алгоритм молі і полум’я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак, що дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж та ускладнює дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп’ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах;

4) розроблено новий метод виявлення атак відмови в обслуговуванні у мережах на основі статистичних показників, який на відміну від відомих, базується на обчисленні статистичних ознак мережного IP-трафіку при розбитті потоку пакетів на часові вікна, і встановлює динамічні зміни трафіку на рівні всього аналізованого періоду, що дозволяє підвищити достовірність виявлення атак відмова в обслуговуванні.

Практичне значення отриманих результатів. Розроблено обманну систему з приманками і пастками для виявлення КА та ЗПЗ в корпоративних мережах, особливістю якої є прийняття в ній рішень щодо наступних кроків та їх коригування з використанням алгоритму дискретної оптимізації молі і полум’я, а також імплементацією в її компонентах методу виявлення комп’ютерних атак на основі аналізу їх статичних показників.

Синтез популяційних алгоритмів в архітектурі оманних систем для прийняття ними рішень дав змогу формувати послідовності кроків систем так, щоб заплувати

зловмисників при проведенні КА. Також, в процесі синтезу алгоритму молі і полум'я в архітектуру обманних систем було здійснено розроблення його кроків та адаптації для реалізації саме для задач дискретної оптимізації, що є основою для здійснення аналогічних кроків в процесі деталізації інших популяційних алгоритмів натхненних живою природою.

За результатами проведених експериментальних досліджень встановлено, що розроблена ОСПП забезпечує коректне функціонування в умовах динамічної зміни оточуючого середовища корпоративних мереж, ефективне залучення приманок і пасток для виконання задач виявлення інфікованих програм, а також вибір наступних кроків для виконання.

Теоретичні та практичні результати дослідження впроваджені в ТОВ «Nolt technologies» (м. Хмельницький, Акт від 16.02.2026), ТОВ «ІТТ» (м. Хмельницький, Акт від 16.02.2026), а також, в освітньому процесі Хмельницького національного університету (Акт від 25.02.2026) при викладанні дисциплін на кафедрі комп'ютерної інженерії та інформаційних систем для здобувачів спеціальності F7 Комп'ютерна інженерія, зокрема в курсах «Безпека та захист комп'ютерних систем», «Моделювання та методи оптимізації в наукових та експериментальних дослідженнях», «Методології забезпечення якості, надійності, гарантоздатності та безпеки комп'ютерних систем та мереж» та в освітній процес у блоці військово-спеціальних дисциплін другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка (Акт від 18.12.2025), які використані при удосконалені навчально-лабораторного комплексу.

Особистий внесок здобувача. Всі основні результати дисертаційного дослідження, які представлені до захисту, отримані автором особисто.

У роботах, опублікованих одноосібно автором, отримано наступні результати: [168] – розроблено метод виявлення комп'ютерних атак типу відмови в обслуговуванні на основі статистичних показників мережного трафіку; [171] – розроблено метод організації функціонування ОСПП в корпоративних мережах.

У роботах, які опубліковані у співавторстві, автору належать основні ідеї, теоретична та практична розробка положень, відображених у характеристиці наукової новизни отриманих результатів, а саме: [103] – розроблено архітектуру обманних систем з приманками на основі використання популяційних алгоритмів в підсистемі прийняття рішень для вибору наступних кроків систем із множини

наявних кроків; [134, 169] – розроблено метод синтезу популяційних алгоритмів в архітектурі ОСПП, зокрема алгоритм дискретної оптимізації молі і полум’я; [170] – визначено стратегію та основні кроки методу для здійснення оцінювання рівнів кібербезпеки у вузлах корпоративних мереж; [135] – проведено аналіз готових засобів для розгортання бінарного класифікатора; [113] – визначено стратегію застосування штучних нейромереж в кроки методу виявлення бот-мереж в корпоративну ІТ-інфраструктуру; [108] – визначено стратегію тестування безпеки застосунків в корпоративних мережах; [110] – визначено статистичні показники комп'ютерних атак на основі мережного трафіку; [136] – визначено показники операційних систем реального часу, які впливають на зміну стану операційного середовища систем; [181] – розроблено архітектуру програмного забезпечення та програмний код обманних систем в корпоративних мережах з прийняттям рішень на основі популяційних алгоритмів.

У роботах, які опубліковані у співавторстві, співавторам належать такі результати: [103] – запропоновано концепцію синтезу популяційних алгоритмів в архітектурі обманних систем з приманками для вибору наступних кроків систем із множини наявних кроків (Савенко О.), здійснено постановку експерименту щодо функціонування обманних систем для виявлення КА (Медзатий Д.); [169] – запропоновано концепцію синтезу алгоритму молі і полум’я в архітектурі ОСПП (Савенко О.), здійснено визначення умов дискретної оптимізації та постановку експерименту (Коробчинський М.); [134] – розроблено концепцію для синтезу алгоритмів дискретної оптимізації в архітектурі обманних систем з приманками і пастками (Савенко О.), розроблено методологічний підхід до здійснення оптимізації функціонування обманних систем з приманками і пастками під час КА (Савенко О., Саченко А., Савенко Б.), розроблено програмне забезпечення для реалізації алгоритму молі і полум’я в архітектурі обманних систем з приманками і пастками (Савенко Б., Нічепорук А.), здійснено валідацію розробленого методу та програмного забезпечення (Лисенко С., Нічепорук А.), здійснено дослідження та формальний аналіз запропонованого рішення (Савенко О., Саченко А., Ciszewski T., Русин Б.); [170] – розроблено метод для оцінювання рівнів кібербезпеки в корпоративних мережах (Рамський І.), розроблено методику оцінювання рівнів безпеки вузлів в корпоративних мережах (Поночовна О.), здійснено постановку експерименту та опрацьовано його результати (Лигун О.); [135] – запропоновано визначення ролі

обчислювального елемента на основі його поведінки під час функціонування системи (Савенко О.), запропоновано використання рейтингу обчислювального елементу та бінарного класифікатора для його коригування (Саченко А.), розроблено модель із ролями обчислювальних елементів розподіленої системи та її реалізація (Регіда П.), проведено аналіз готових засобів для розгортання бінарного класифікатора (Дрозд А.), сформовано список інформації який можна збирати на обчислювальному елементі, шляхом розширення його функціональності (Віжевський П.); [113] – розроблено метод виявлення бот-мереж в ІТ-інфраструктурі (Денисюк Д..), розроблено принципи застосування нейронних мереж до задачі виявлення бот-мереж (Сохор Т.), розроблено концепцію і метод виявлення ЗПЗ і КА на основі бот-мереж в ІТ-інфраструктурі (Каштальян А.), здійснено постановку експерименту та опрацьовано його результати (Капустян М.); [108] – здійснено вдосконалення методу статичного тестування безпеки застосунків у контексті вразливостей SQL Injection (Сєргєєв Є.), визначено концепцію здійснення тестування застосунків в корпоративних системах (Савенко О.), здійснено постановку експерименту щодо вразливостей застосунків (Паюк В.); [110] – здійснено вдосконалення кроків методу виявлення комп'ютерних атак на основі озвучення мережного трафіку (Семенюк Б.), розроблено концепцію та метод виявлення комп'ютерних атак згідно показників мережного трафіку (Каштальян А.), здійснено постановку концепції застосування штучних нейронних мереж для виявлення комп’ютерних атак та експерименту щодо їх виявлення (Abdel-Badeeh M. Salem); [136] – розроблено модель імовірнісного моніторингу та превентивного перезапуску ОСРЧ в умовах інтенсивних змін станів у КФС (Козельський О.), розроблено концепцію імовірнісного моніторингу та проактивного перезапуску операційних систем реального часу за інтенсивних змін стану кіберфізичних систем (Савенко Б.), здійснено постановку експерименту та визначено параметри операційного середовища (Piotr Gaj); [181] – розроблено вимоги до програмного забезпечення та здійснено його верифікацію (Савенко О.), розроблено алгоритми, які включають формування кроків обманних систем (Нічепорук А., Регіда П.).

Апробація результатів дисертації. Апробацію основних положень, ідей, висновків дисертаційної роботи проведено на науковому семінарі кафедри комп’ютерної інженерії та інформаційних систем у Хмельницькому національному університеті. Наукові результати роботи доповідались на таких конференціях: 5th

International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS, Khmelnytskyi, Ukraine, March 28, 2024); 6th International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS, Khmelnytskyi, Ukraine, April 04, 2025,); 14th International Conference on Dependable Systems, Services and Technologies (IEEE, DeSSerT, Athens, Greece, October 11-13, 2024); 1st International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS, Khmelnytskyi, Ukraine, June 28, 2024); 2nd International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS, Khmelnytskyi, Ukraine, July 4, 2025).

Публікації. За результатами проведених досліджень основні наукові результати опубліковано у шести наукових статтях в чотирьох фахових наукових журналах України [103; 134; 168-171] та одному міжнародному науковому журналі [134], що індексується в наукометричній базі Scopus. Апробація засвідчена публікаціями п’яти праць в матеріалах міжнародних конференцій [108; 110; 113; 135; 136], які проіндексовано у наукометричній базі Scopus. Опубліковано та отримано одне свідоцтво про реєстрацію авторського права на твір (програму) [181].

Структура та обсяг дисертації. Дисертаційна робота складається з анотації, змісту, переліку умовних скорочень, вступу, чотирьох розділів, висновку, списку використаних джерел та п’яти додатків. Повний обсяг роботи містить 267 сторінок друкованого тексту, з них анотація – на 12 с., зміст – на 2 с., перелік скорочень – на 1 с., основний текст – на 150 с., список із 181 використаного джерела – на 19 с., додатки – на 68 с. Дисертація містить 29 рисунків та 14 таблиць.

РОЗДІЛ 1
АНАЛІЗ МЕТОДІВ ФУНКЦІОНУВАННЯ СИСТЕМ ВИЯВЛЕННЯ ЗПЗ ТА КА В КОРПОРАТИВНИХ МЕРЕЖАХ

1.1 Аналіз відомих систем виявлення КА та ЗПЗ

Поняття КА та ЗПЗ, а також системи та засоби їх виявлення регламентуються відповідними нормативними документами [60; 61]. В них [62; 63], також, визначені регламенти, яких повинні дотримуватись власники та адміністратори корпоративних мереж. Для забезпечення кіберзахисту автоматизованих систем [61], зокрема і корпоративних інформаційних систем та мереж, можуть бути застосовані різні системи і засоби, включно з обманними системами, приманками і пастками. Розглянемо поширені системи та засоби виявлення КА та ЗПЗ.

Bitdefender Total Security [64] надає рішення з кібербезпеки для малого та середнього бізнесу, середніх підприємств та споживачів. Продукти Bitdefender згруповані у три основні категорії: для дому, для бізнесу та для партнерів. Продукти Bitdefender призначені для використання на різних платформах: настільних, серверних, мобільних та хмарних, підтримуючи більшість операційних систем, що працюють на цих платформах.

AhnLab V3 Endpoint Security [65] — комплексний асортимент продуктів безпеки, який включає антивірусні продукти для комп'ютерів та серверів, продукти для мобільних пристроїв, продукти для безпеки онлайн-транзакцій, пристрої для мережної безпеки та консалтингові послуги. Продукти AhnLab V3 досягли найвищих показників виявлення в глобальних оцінюваннях, таких як AV-TEST [66]. Пакет антивірусного програмного забезпечення динамічно виявляє загрози кінцевим точкам, використовуючи платформу багатовимірного аналізу, оснащену поведінковим виявленням, хмарним машинним навчанням, виявленням пам'яті процесів тощо. Організації можуть використовувати продукти AhnLab V3 окремо або разом з платформою захисту кінцевих точок (AhnLab EPP) та платформою захисту хмарних ресурсів (AhnLab CPP) відповідно до своїх вимог безпеки.

Sophos Home Premium [67] — рішення для захисту кінцевих точок. Здійснює захист усіх пристроїв: кінцеві точки; мобільні пристрої; хмарні; локальні сервери;

віртуальні сервери. Все керується через хмарну платформу безпеки Sophos, яка також доступна як цілодобова служба керованого виявлення та реагування. Портфоліо мережної безпеки Sophos надає все необхідне для підключення розподіленої мережі як на периферії, так і в межах основної мережі.

AVG Internet Security [68] – це проактивна технологія штучного інтелекту, яка постійно виявляє частини вірусів та зловмисних програм. Засіб може сканувати комп'ютери на наявність вірусів, програм-вимагачів, шпигунських програм, прихованих цифрових загроз у програмах тощо, а також здійснює захист особистих файлів від потенційної крадіжки, допомагає тримати кіберзлочинців віддалено від особистих файлів за допомогою вдосконаленого брандмауера, блокує ненадійні програми від використання веб-камери.

TotalAV Pro Antivirus [69] захищає до трьох пристроїв за допомогою простих у використанні програм на всіх пристроях Windows, Mac, iOS та Android. TotalAV Pro Antivirus блокує доступ до веб-сайтів, які можуть зашкодити пристрою користувача або поставити під загрозу користувача ПК.

Surfshark One Antivirus [70] - пакет кібербезпеки для повного захисту. Сповіщення повідомляє коистувача про момент витоку електронної пошти, посвідчення особи, кредитної картки чи інших особистих даних в Інтернет. Surfshark пропонує всі основні функції VPN.

PC Protect [71] – це захист, який включає численні розширені функції безпеки для безпеки в Інтернеті. Вбудований повнофункціональний VPN-сервіс дозволяє приховувати IP-адресу, зберігаючи конфіденційність особи та історії переглядів. Програми доступні на всіх пристроях Windows, Mac, iOS та Android. SafeBrowsing приховує IP-адресу користувача та шифрує мережні дані.

Scanguard [72] – антивірусний засіб, в якому наявний набір функцій повноцінного пакету безпеки. Повний захист Scanguard складається з усіх функцій кібербезпеки та захисту інформації. Scanguard має додаткові функції, такі як безпечний перегляд (VPN) для повної онлайн-безпеки, Система дає змогу здійснювати запуск планового антивірусного сканування та сканування для прискорення роботи системию Захист у режимі реального часу працює непомітно у фоновому режимі та виявить і помістить у карантин будь-які небезпечні файли, перш ніж вони зможуть інфікувати систему.

Guardio [73] захищає комп’ютер і і користувачів від різних зловмисних онлайн-шахрайств та кібератак. Він фактично запобігає потраплянню на комп’ютери користувачів. Guardio – це розширення для браузера, яке працює на хмарному рівні, а не знаходиться на комп’ютері.

Malwarebytes Premium [74] – це антивірус та додаткові засоби. Malwarebytes Premium Security запобігає загрозам у режимі реального часу, зупиняє програми-вимагачі, захищає від зловмисних сайтів, очищає та видаляє ЗПЗ.

Panda Dome [75] – це Panda Security, бренд WatchGuard Technologies. Система пропонує найсучасніший захист. Її лінійка Panda Dome забезпечує максимальний захист від вірусів, програм-вимагачів та комп’ютерного шпигунства, і сумісна з Windows, Mac, Android та iOS.

Comodo Internet Security [76] включає антивірус, брандмауер, пісочницю, запобігання вторгненням хоста та інші функції захисту. Comodo Antivirus включає розширений захист обладнання завдяки партнерству з технологією виявлення загроз Intel, швидше сканування та повністю оновлений інтерфейс користувача для оптимізованої роботи.

ZoneAlarm Extreme Security [77] – є персональним брандмауером, який створено для домашніх персональних комп’ютерів. Захист здійснюється від нещодавно виявлених вірусів, шпигунських програм, троянських програм і worm-вірусів за допомогою антивірусу та антишпигунського ПЗ в реальному часі. Виявлення та видалення загроз здійснюється їх скануванням за допомогою актуальної хмарної бази даних антивірусних сигнатур.

Emsisoft Anti-Malware [78] – система кібербезпеки для малого та середнього бізнесу. Об’єднано два антивірусні механізми в один. Швидкий сканер знаходить та очищає зловмисні та небажані програми, блокує підключення до зловмисних веб-сайтів, забезпечує комплексний захист у режимі реального часу від усіх типів зловмисних програм, постійно контролює поведінку всіх активних процесів.

SentinelOne Singularity [79] охоплює запобігання, виявлення, реагування та пошук загроз на базі штучного інтелекту для кінцевих точок користувачів, контейнерів, хмарних робочих навантажень та пристроїв Інтернету речей.

Seqrite Endpoint Security [80] охоплює інструменти та методи для виявлення, запобігання та реагування на складні кіберзагрози, спрямовані на кінцеві точки.

Завдяки передовим функціям, таким як сканування виявлення та управління активами, це рішення забезпечує цілісну безпеку цифрових активів підприємств, захищаючи критично важливу інформацію та інфраструктуру бізнесу.

Symantec Endpoint Security Complete [81] захищає всі традиційні та мобільні кінцеві пристрої за допомогою інноваційних технологій для зменшення поверхні атаки, запобігання атакам, запобігання порушенням, а також виявлення та реагування на них. Весь цей захист забезпечується глобальною розвідувальною мережею, однією з найбільших у світі. Одноагентне рішення Symantec пропонує гнучкі варіанти управління/розгортання, включаючи повністю хмарні, локальні та гібридні рішення. Завдяки рішенням для безпеки кінцевих точок можна захистити скомпрометовані кінцеві точки, які можуть бути дуже руйнівними для бізнесу. Symantec забезпечує захист від прихованого ЗПЗ, програм-вимагачів, крадіжки облікових даних, крадіжки облікових даних тощо. Symantec пропонує виявлення та усунення загроз за допомогою складної аналітики атак та автоматизованого реагування.

WithSecure Elements Endpoint Protection [82] забезпечує безперервний захист кінцевих пристроїв, незалежно від типу операційних систем, наприклад, Windows, Mac, Linux, чи для мобільних пристроїв. Проактивні можливості запобігання допомагають блокувати або відкатувати програми-вимагачі, інші зловмисні програми або навіть безфайлові атаки.

Intego Antivirus [83] використовується для захисту комп’ютерів від вірусів, шпигунських програм та інших програм. Система блокує зловмисні програми, шпигунські програми, рекламне ПЗ, програми-вимагачі та інші загрози ще до того, як вони атакують. Вдосконалений механізм запобігання Intego блокує багато нових загроз, з якими традиційне антивірусне програмне забезпечення не може впоратися.

Norton Antivirus [84] допомагає захиститися від онлайн-загроз, таких як шахрайство тощо. Система здійснює моніторинг та блокування несанкціонованих повідомлень, щоб запобігти проникненню хакерів на комп’ютер.

McAfee [85] пропонує сканування в режимі реального часу, на вимогу та за розкладом, що дозволяє автоматично або вручну сканувати пристрій на наявність ЗПЗ та інших загроз, зберігаючи конфіденційність та захист ваших даних. АПЗ, також, включає захист особистих даних та VPN. Технологія штучного інтелекту допомагає виявляти та видаляти найновіші загрози.

Fortinet Network Firewall [86] - це найпоширеніший мережний брандмауер, який займає понад 50% світового ринку. Брандмауери наступного покоління FortiGate захищають дані, активи та користувачів у сучасних комбінованих середовищах.

Avira Antivirus [87] поєднує захист у режимі реального часу з технологією безпеки Protection Cloud, яка перевіряє невідомі файли на наявність зловмисних програм та експлойтів, перш ніж вони зможуть потрапити на комп'ютер. Завдяки штучному інтелекту антивірусний захист постійно навчається та автоматично адаптується до нових кіберзагроз.

Avast Antivirus [88] антивірусне програмне забезпечення, яке оснащене однією з найсучасніших мереж виявлення загроз, захистом від вірусів на основі машинного навчання та безпекою домашньої мережі, що не уповільнить роботу ПК з Windows, Mac, Android або iPhone. Сканує файли та програми на наявність потенційних вразливостей ЗПЗ. Крім того, надсилає підозрілі файли на аналіз у хмару, надає сповіщення про загрози тощо. Сканує пристрій на наявність важкодоступних вразливостей у прихованих місцях. Автоматично надсилає підозрілі файли на аналіз у хмару, а потім, за потреби, пропонує користувачам Avast рішення.

Zeek [89] (попередня назва - «Bro») був розроблений для надання глибокого розуміння мережної активності в університетських та національних лабораторних мережах. Zeek залишається дуже затребуваним рішенням з відкритим кодом завдяки розробці та фінансовій підтримці Corelight. На відміну від традиційних інструментів безпеки, таких як брандмауери або системи запобігання вторгненням, Zeek не є активним захисним механізмом. Натомість він непомітно працює на апаратному, програмному, віртуальному чи хмарному датчиках, аналізуючи мережний трафік у режимі реального часу. Zeek фіксує високоточні журнали транзакцій, вміст файлів та налаштовувані вихідні дані, які ідеально підходять для ручного перегляду або інтеграції в системи SIEM для аналітиків безпеки. Поглиблений аналіз Zeek постачається з аналізаторами для багатьох протоколів, що дозволяє проводити високорівневий семантичний аналіз на рівні програми. Адаптивна та гнучка мова сценаріїв Zeek, орієнтована на певний домен, дозволяє використовувати політики моніторингу для конкретних сайтів і означає, що вона не обмежується жодним конкретним підходом до виявлення. Ефективний Zeek орієнтований на високопродуктивні мережі та використовується в операційній діяльності на різних

великих сайтах. Zeek з високим рівнем стану зберігає розширений стан мережі на рівні застосунків, яку він контролює, та надає високорівневий архів активності мережі.

OSSEC [90] це система виявлення та реагування на основі відкритого коду. Додає тисячі покращених правил, часті оновлення та інтеграції ПЗ, вбудоване активне реагування, графічний інтерфейс користувача, інструменти відповідності та професійну підтримку експертів.

Security Onion [91] - це безкоштовна та відкрита платформа, яка включає видимість мережі, видимість хостів, приманки для виявлення вторгнень, керування журналами та керування справами. Приманки для виявлення вторгнень можна додати до розгортання для ще більшої видимості підприємства.

Wazuh [92] - об'єднує історично окремі функції в єдину агентську та платформенну архітектуру. Захист забезпечується для публічних хмар, приватних хмар та локальних центрів обробки даних. Платформа розширеного виявлення та реагування надає комплексне рішення безпеки, яке виявляє, аналізує та реагує на загрози на кількох рівнях IT-інфраструктури. Wazuh збирає дані з кінцевих точок, мережних пристроїв, хмарних робочих навантажень, сторонніх API та інших джерел для єдиного моніторингу та захисту безпеки.

MISP [93] складається з кількох ініціатив, від ПЗ для полегшення аналізу та обміну загрозами до вільно використовуваної структурованої інформації про кіберзагрози та таксономій. Доступ до великої кількості інформації про загрози через спільноти MISP Threat Sharing надає можливості для агрегації інформації та виведення процесу.

Osquery [94] - це фреймворк для інструментації операційних систем Windows, OS X (macOS) та Linux. Ці інструменти роблять низькорівневу аналітику та моніторинг операційної системи одночасно продуктивними та інтуїтивно зрозумілими. Osquery надає операційну систему як високопродуктивну реляційну базу даних.

Платформа Labyrinth Deception Platform [95] створює штучне мережне середовище, що імітує реальні сервіси, контент і пристрої. Інтелектуальні комопненти працюють як фальшиві хости, які виявляють підозрілу активність і спрямовують зловмисника в контрольовану зону, дозволяючи фіксувати та відстежувати його дії.

Attivo ThreatDefend Deception and Response Platform [86] стала однією з перших систем, що поєднала обман з активним реагуванням. Платформа підтримує роботу в локальних, хмарних і гібридних інфраструктурах. Її приманки не лише фіксують спроби доступу, а й взаємодіють зі зловмисником, імітуючи роботу справжніх систем для глибшого аналізу його тактик.

CounterCraft Cyber Deception Platform [97] застосовує гнучкі активні приманки, які можуть розгортатися на кінцевих точках, серверах або онлайн. Система забезпечує інтерактивну взаємодію зі зловмисником і швидке розгортання завдяки агентам, що збирають дані в реальному часі.

Fidelis Deception Platform [98] дає змогу швидко створювати реалістичні фальшиві середовища з приманками операційних систем, сервісів, мережних з'єднань та облікових даних. Усі дії в обманному сегменті фіксуються для подальшої реакції. Система вирізняється точністю, відсутністю хибних спрацювань та постійним оновленням обманних сценаріїв.

CommVault [99] пропонує комплексний захист даних: оцінку ризиків; резервування; автоматичне відновлення; виявлення загроз; захист критичних сервісів. Платформа орієнтована на раннє виявлення та блокування атак, включно з атаками «нульового дня» та прихованими загрозами, що поширюються в мережі.

Proofpoint Identity Threat Defense [100] формує невидиме для зловмисника обманне середовище. Безагентний підхід унеможливлює розпізнання приманок. Система відстежує зміни в інфраструктурі та активує обманні механізми для зупинки атак на ранніх етапах, захищаючи поштові, мобільні, соціальні та робочі платформи.

Таким чином, наявні системи є досить складними, багатофункціональними і можуть ефективно вирішувати завдання забезпечення безпеки та захисту корпоративних мереж. Перспективним класом серед розглядуваних систем і взагалі систем забезпечення безпеки та захисту корпоративних мереж від КА чи дій ЗПЗ є клас обманних систем [33; 101 - 103], до яких можуть бути долучені приманки і пастки. Системи цього класу, крім виявлення КА чи ЗПЗ, можуть заплутувати зловмисників, через що ті втрачають час та ресурси, а також для них ускладнюється здійснення КА.

1.2 Методи виявлення комп’ютерних атак в корпоративних мережах

Для виявлення КА і ЗПЗ активно застосовуються методи штучного інтелекту [104; 105] в системах забезпечення безпеки та захисту корпоративних мереж. Загрози інформації та ресурсам корпоративних мереж і, відповідно, протидії їм залежать від типів операційних систем [106], вразливостей ПЗ та систем захисту [107; 108], методів пошуку вразливостей [109], методів виявлення КА [110; 111], застосування комплексних систем захисту IT-інфраструктури [112; 113], моніторингу стану та подій в корпоративних мережах [114], застосування додаткових засобів захисту типу приманок і пасток [115; 116] тощо. Розглянемо методи виявлення КА в корпоративних мережах та проблеми, які наявні в процесі виявлення.

У роботі [117] кібербезпека визначена як нова проблема для управління інформаційними технологіями в бізнесі та суспільстві завдяки швидкому розвитку телекомунікаційних та бездротових технологій. Безпека кіберпростору мала вплив на численні ключові інфраструктури. Поряд із даними про поточний стан безпеки, система повинна отримувати історичні дані для впровадження новітнього захисту та захисту в галузі кібербезпеки. Вона також приймає інтелектуальні рішення, які можуть забезпечити адаптивне управління безпекою та контроль. У роботі розроблено інтелектуальну систему кібербезпеки з використанням налаштування гіперпараметрів на основі методу регуляризованої пам'яті для підвищення рівня безпеки основних системних активів.

Кіберстійкість [118] у хмарних обчисленнях є однією з найважливіших можливостей корпоративних мереж, яка забезпечує постійну здатність протистояти несприятливим умовам та швидко відновлюватися після них. Цю здатність можна виміряти за допомогою методів оцінки ризиків кібербезпеки. Однак дослідження управління ризиками кібербезпеки в підходах до стійкості хмарних обчислень є недостатніми. Запропонована робота намагається виявити вразливості хмари, оцінити загрози та виявити компоненти високого ризику, щоб запропонувати відповідні заходи безпеки.

У роботі [119] розглянуто клас мереж, які можуть бути створені в паралельному часі за допомогою конструкторів мереж. Починаючи з порожньої мережі, метою роботи була побудова стабільної мережі, яка належить до заданої підмножини. У

роботі [120] проаналізовано чотири парадигми паралельних обчислень – гетерогенні обчислення, квантові обчислення, нейроморфні обчислення та оптичні обчислення. Було розглянуто нові розподілені системи, такі як блокчейн, безсерверні обчислення та хмарні архітектури.

У роботі [121] представлено дослідження комплексного, динамічного та багаторівневого підходу до оцінки та моделювання кіберризиків у розподілених інформаційних системах на основі метрик безпеки та методів їх розрахунку, який демонструє здатність вирішувати задачі інтелектуальної класифікації та моделювання оцінки ризиків для великих масивів розподілених даних.

У роботі [122] розглянуто проблеми інтелектуальних мереж щодо кіберфізичних систем та систем кібербезпеки, стандартних протоколів, комунікаційних та сенсорних технологій. Існуючі методи машинного навчання на основі навчання з учителем для виявлення кібератак в інтелектуальних мережах здебільшого спираються на випадки як нормальних, так і атакуючих подій для навчання. Крім того, щоб навчання з учителем було ефективним, навчальний набір даних повинен містити репрезентативні приклади різних сценаріїв атаки з різними закономірностями, що є складним завданням. Тому, в роботі запропоновано новий підхід до інтелектуального аналізу даних, який базовано на правилах без учителя для виявлення кібератак з введенням помилкових даних в інтелектуальних мережах.

У роботі [123] досліджуються фундаментальні вимоги до зв'язку, структури та протоколи, необхідні для встановлення безпечного з'єднання в мікромережах. Крім того, оцінюється інтеграція різноманітних методів безпеки. У роботі представлено тематичне дослідження, яке ілюструє впровадження розподіленої системи кібербезпеки в умовах мікромережі. У роботі [124] розглянуто проблеми мережного ризику, з якими стикаються кіберсистеми, Враховуючи можливі вразливості кібератак пропонується динамічний байєсівський мережний підхід для кількісної оцінки ризику безпеки розподіленої мережі. У роботі [125] розглянуто нові обчислювальні парадигми, такі як безсерверні обчислення, периферійні обчислення та обчислення на основі блокчейну.

У роботі [126] представлено інноваційну стратегію захисту для активних розподілених мереж, що використовує принципи розподіленої координації та багатоагентних систем. Запропонована стратегія складається з двох етапів. Перший

етап включає алгоритм виявлення несправностей, який спирається виключно на локальні вимірювання, тоді як другий етап використовує класифікацію агентів для обчислення оптимального часу роботи на основі динамічного матричного представлення шляху несправності в поєднанні зі спрощеною задачею розподіленої оптимізації. Процес координації сформульовано як набір задач лінійної оптимізації, що спрощує рішення.

У роботі [127] розглянуто розподілений закон адаптивного керування на основі моделі для керування набором агентів з динамікою подвійного інтегратора за схемою "лідер-слідуваний" за наявності системних аномалій, таких як невизначеності, пов'язані з агентами, невідома ефективність керування та динаміка виконавчих механізмів. Показано стійкість загальної замкнутої багатоагентної системи.

Сфера [128] самоадаптивного програмного забезпечення стає дедалі важливішою, оскільки програмне забезпечення повинно адаптувати свою поведінку під час виконання, щоб встигати за динамічними та постійно мінливими середовищами. Здатність програмного забезпечення до самомодифікації та налаштування відома як адаптивність, що визнано важливим атрибутом якості. Ступінь, до якої архітектура може адаптуватися до змін, буде ключовим фактором у визначенні адаптивності програмного забезпечення.

У роботі [129] розглянуто проблеми безпеки, такі як несанкціонований доступ, а також проблеми конфіденційності, які створюють значні перешкоди. Їх можна подолати, впроваджуючи методи контролю доступу та динамічну політику безпеки та конфіденційності, яка регулює ці проблеми там, де вони виникають. У роботі запропоновано алгоритм навчання з учителем на основі прийняття рішень щодо політик та контролю доступу. Безпека контролю доступу покращується завдяки тому, що він стає динамічним, адаптивним, гнучким та розподіленим.

У роботі [130] розглянуто самоорганізацію у складних системах. Це процес, який пов'язаний зі зниженням внутрішньої ентропії та виникненням структур, які можуть дозволити системі функціонувати ефективніше та стійкіше у своєму середовищі та більш конкурентно з іншими станами системи або з іншими системами.

У роботі [131] розглянуто системи виявлення вторгнень, які зазвичай поділяються на системи виявлення вторгнень, що засновані на неправильному використанні, та аномаліях. Системи виявлення вторгнень на основі аномалій можуть

виявляти нові атаки, але вони не часто використовуються в промисловості, бо мають високий рівень хибнопозитивних результатів та відсутність інтерпретації навчених моделей. Системи виявлення вторгнень на основі специфікацій, особливо з використанням навчання за правилами, може виявитися перспективнішим. Правила виявлення вторгнень вивчаються за допомогою методів навчання за правилами та періодично автоматично оновлюються для врахування динамічної поведінки системи.

У роботі [132; 133] розглянуто різні загрози, які мають дві основні сфери: внутрішня вразливість системи; зовнішні кібератаки.

Таким чином, крім безпосередньо розроблення нових методів або удосконалення відомих методів для покращення безпеки і захисту корпоративних мереж від КА та дій ЗПЗ необхідно розробляти ефективні системи, в яких реалізуються методи вичвлення.

1.3 Застосування розподілених систем для виявлення КА в корпоративних мережах

Перспективними системами для виявлення КА та ЗПЗ в корпоративних мережах є розподілені системи порівняно з хостовими [134; 135]. Вони можуть містити приманки та пастки [136; 137]. Особливо важливою є їх внутрішня архітектура [138; 139] та її розподілені елементи і компоненти [140], що впливають на ефективність функціонування таких систем та процес виявлення КА і ЗПЗ.

У роботі [141] наведено цілеспрямовану оцінку поточних кіберзагроз, з якими стикаються університети на основі всебічного огляду доступної інформації. Складено хронологічний графік відомих кібератак на університети, при цьому інциденти класифікуються відповідно до тріади (конфіденційність, цілісність, доступність) та типу інциденту.

У роботі [142] розглянуто тенденції безпеки в хмарних моделях. З врахуванням тенденцій безпеки було проаналізовано проблеми безпеки, включаючи порушення даних, конфіденційність даних, контрольованість доступу до даних, автентифікацію, неадекватну перевірку, фішинг, розкриття ключів, аудит, збереження конфіденційності та хмарні IoT-застосунки.

У роботі [4] розглянуто модель поведінки зловмисників, при якій вони почали використовувати КА через автоматизацію атак та посилення їхнього впливу. Зловмисники використовують вразливості, що існують в апаратному, програмному та комунікаційному рівнях. Різні типи кібератак включають розподілену відмову в обслуговуванні, фішинг, атаки типу "людина посередині", атаки за паролем, віддалені атаки, ескалацію привілеїв та ЗПЗ. У роботі розглянуто найновіші КА, моделі атак та методи виявлення. Встановлено, що використання трендових технологій, таких як машинне навчання, глибоке навчання, хмарні платформи, великі дані та блокчейн, є перспективним рішенням для виявлення поточних та майбутніх кібератак.

У роботі [144] запропоновано метод на основі комп'ютерного зору для виявлення зловмисних програм, що знаходяться в основній пам'яті комп'ютера. Він працює, беручи виконувані файли у віртуальному середовищі для вилучення файлів дампа пам'яті з енергозалежної пам'яті та перетворення їх у певний формат зображення.

У роботі [145] запропоновано та досліджено сім гібридних моделей машинного навчання для виявлення шахрайської діяльності з використанням набору даних реальних текстів. У роботі [146] розглянуто методи штучного інтелекту, які використовуються для опису характеристик інформації, оскільки вони допомагають у процесі аналізу даних аналізувати дані та виявляти правила та закономірності. Виявлення аномалій є важливою областю, яка допомагає виявляти приховану поведінку в даних, яка є найбільш вразливою до атак. Це також допомагає виявляти вторгнення в мережу. Такі алгоритми, як гібридний масив K-середніх та послідовна мінімальна оптимізація, можуть бути використані для підвищення точності коефіцієнта виявлення аномалій. У роботі представлено модель виявлення аномалій на основі методу машинного навчання.

У роботі [147] розглянуто одну з найважливіших проблем, з якими стикаються користувачі мережі Інтернету. Такою проблемою є ЗПЗ. Поліморфне ЗПЗ – це новий тип ЗПЗ, яке є більш адаптивним, ніж попередні покоління вірусів. Поліморфне ЗПЗ постійно змінює свої сигнатурні риси, щоб уникнути ідентифікації традиційними моделями виявлення ЗПЗ на основі сигнатур. Для виявлення зловмисних загроз або ЗПЗ використано методи машинного навчання.

У роботі [148] розглянуто програмно-визначені мережі. Це нова мережна парадигма, яка забезпечує централізоване керування, програмованість та глобальний огляд топології в контролері. Розподілена атака відмови в обслуговуванні – це мережна атака, яка переповнює мережні з'єднання незаконними даними з використанням високошвидкісної передачі пакетів. Нелегітимний трафік даних може перевантажувати мережні з'єднання, що призводить до втрати законних даних та недоступності мережних служб. Низькошвидкісна розподілена відмова в обслуговуванні– це нещодавня еволюція DDoS-атаки, яка стала однією з найсерйозніших вразливостей для Інтернету, платформ хмарних обчислень, Інтернету речей та великих центрів обробки даних.

У роботі [149] застосовано підхід машинного навчання для виявлення мережних аномалій та створено різні моделі на основі даних для виявлення DDoS-атак. У роботі [150] розглянуто зростання використання комп'ютерних мереж та відповідне збільшення кількості кібератак. У роботі [151] розглянуто виявлення вторгнень у комп'ютерні мережі, яке має велике значення через його вплив на різні сфери зв'язку та безпеки. Виявлення вторгнень у мережу є складним завданням. Більше того, виявлення вторгнень у мережу залишається складним завданням, оскільки для навчання сучасних моделей машинного навчання для виявлення загроз вторгнення в мережу потрібна велика кількість даних.

У роботі [152] розглянуто кілька видів атак та можливі наслідки для розуміння ландшафту безпеки в галузі IIoT. У роботі [153] представлено комплексну структуру, спрямовану на формування майбутнього кібербезпеки. Ця структура реагує на складність сучасних кіберзагроз і надає організаціям рекомендації щодо підвищення їхньої стійкості. Основна увага приділяється інтеграції можливостей зі стійкістю. У роботі [153] розглянуто моделі атак, реальні випадки кібервторгнення, що виникли, та превентивні рішення безпеки. У роботі [154] проводиться порівняльний аналіз серед систем виявлення вторгнень на основі машинного навчання для розробки орієнтиру різних запропонованих стратегій. У роботі [155] розглянуто виявлення кібератак у системі дискретних подій і запропоновано нову модель. Модель використовує графові згорткові мережі для вилучення просторових ознак із послідовностей подій. Модель на основі глибокого навчання обходить проблеми вибуху станів. Метод сприяє виявленню кібератак у режимі реального часу, усуваючи

необхідність спеціальної ідентифікації станів системи або розрізнення типів атак. Крім того, встановлено регульований поріг ймовірності, щоб визначити скомпрометованість послідовності подій.

У роботі [158] розроблено метод аналізу мережного трафіку на основі виявлення зловмисних атак за допомогою методів штучного інтелекту. Аналіз трафіку було проведено за допомогою квадратично-векторної дискримінантної машини ядра, яка покращує передачу даних за рахунок зменшення мережного трафіку. Потім виявлення зловмисних атак здійснюється за допомогою змагальних баєсівських мереж довіри.

У роботі [157] розглянуто поширені кібератак, що впливають на КФС, такі як відмова в обслуговуванні, введення неправдивих даних та атаки повторного відтворення. Пояснено їхній вплив на роботу та цілісність КФС.

У роботі [158] представлено комплексне дослідження інтеграції компонентів машинного навчання в розподіленому сценарії для забезпечення безпечного наскрізного захисту від кіберзагроз, що виникають на рівні пакетів служб віртуальної приватної мережі.

У роботі [159] розглянуто віруси, які є однією з головних загроз безпеці сучасного кіберпростору. Розроблено новий тип підтверджуючого вірусу з точки зору зловмисника, який може інтелектуально заражати програмні фреймворки. Завдяки включенню моделі розпізнавання функціональної структури програмного фреймворку у вірус, вірус отримує можливість інтелектуально розпізнавати функції програмного фреймворку у виконуваних файлах.

У роботі [160] представлено комплексну модель оцінки ризиків кібербезпеки. У роботі [161] розглянуто теоретичний та практичний прогрес, що відбувся в галузі комп'ютерних мереж протягом останніх п'ятнадцяти років, підвищив економічну ефективність та соціальну значущість пов'язаних з ними реальних випадків використання. Проте, це повсюдне використання також принесло численні ризики для безпеки. Тому, моніторинг апаратних та програмних ресурсів є одним з основних інструментів, що використовуються для запобігання потенційним атакам та забезпечення безпеки та надійності мережі.

У роботі [162] розглянуто попит на великомасштабну розподілену систему, таку як інтелектуальна мережа, яка включає взаємозв'язок у режимі реального часу,

швидко зростає. У роботі [163] розглянуто розподілені обчислення з кодуванням. Це потужний підхід до зменшення додаткових витрат на зв'язок у розподілених обчислювальних системах шляхом використання методів кодування.

Дослідники [164] та аналітики кібербезпеки значною мірою покладаються на дані для навчання та тестування моделей виявлення мережних загроз. Комплексні дані, включаючи мережні трасування, телеметрію хоста та контекстуальну інформацію, є критично важливими для цих завдань. Однак широко використовувані публічні набори даних часто страждають від застарілого мережного трафіку та функцій, статистичних аномалій та артефактів моделювання. Крім того, існуючі системи збору даних часто стикаються з архітектурними та обчислювальними обмеженнями, що вимагає обхідних шляхів, які призводять до неповних або розрізнених даних. Наразі жодна система збору даних не забезпечує комплексного збору даних з усіх сегментів мережі без необхідності спеціалізованих або власних апаратних чи програмних агентів.

У роботі [165] представлена емуляція мережі і пропонується гнучке рішення для розгортання та експлуатації мережі, використовуючи програмне забезпечення для консолідації всіх вузлів у топології та використовуючи ресурси одного хост-сервера. У роботі досліджувався стан кібербезпеки у віртуалізованих системах, охоплюючи вразливості, методи експлуатації, методи виправлення та стратегії розгортання. Крім того, представлено розподілену систему, яка інтегрує мережну архітектуру та можливості емуляції.

У роботі [166] розглянуто розподілену бездротову сенсорну мережу, в якій давачі безперервно сприймають навколишнє середовище, збирають дані та передають їх віддаленим користувачам через мережу, щоб здійснювати моніторинг середовища або конкретних цілей у режимі реального часу. Однак, враховуючи відкритість бездротових каналів та чутливість збору даних, розробка надійного протоколу автентифікації користувача для забезпечення легітимності користувача та давачіів у таких середовищах стикається з серйозними проблемами.

У роботі [167] запропоновано обчислювальний механізм для автентифікації достовірності даних подій у застосунках товарного обміну. Події обробляються запропонованою синергетичною обчислювальною системою, яка складається з периферійних пристроїв.

Таким чином, розподілені системи є перспективними в контексті виявлення КА та ЗПЗ в корпоративних мережах, але потребують наповнення не тільки сучасними методами виявлення, але й ефективними методами організації функціонування [168-171], що повинно поєднуватись з методами виявлення та давачами в середовищі корпоративних мереж.

1.4 Популяційні алгоритми в архітектурі систем виявлення комп'ютерних атак

Вибір наступних кроків в обманних системах, які також включають керування приманками і пастками, та їх зміни чи коригування протягом виконання певних послідовностей кроків відповідає постановці задачі оптимізації, суть якої полягає у виборі найкращого (оптимального) варіанта з великої кількості можливих [172]. Оптимальним значенням буде найкращий із можливих варіантів, який максимально відповідає певним умовам, вимогам чи завданню, забезпечуючи найвигідніший, найефективніший чи найбільш підходящий результат.

Задачу оптимізації з вибору наступних кроків ОСПП в корпоративних мережах сформулюємо згідно досягнення підвищення ефективності виявлення КА і ЗПЗ як критерію оптимальності, змінюваності кроків обманних систем як змінюваних параметрів систем, моделей процесів та архітектури обманних систем в корпоративних мережах, а також обмежень щодо операційного середовища корпоративних мереж і особливостей ОСПП. Для її розв'язування потрібен вибір методу, яким би можна було забезпечити знаходження оптимального значення, тобто послідовності кроків систем в змінюваному середовищі корпоративних мереж під час КА чи дій ЗПЗ та із можливістю їх поточної зміни, пов'язаної із потребою коригування відповіді систем на впливи.

Оптимізація відноситься до процесу пошуку найкращого можливого рішення чи рішень для певної проблеми [51]. У міру зростання складності проблем, за останні кілька десятиліть, потреба в нових методах оптимізації стає більш очевидною, ніж раніше. До появи евристичних методів оптимізації застосовували математичні методи оптимізації, які в основному детерміновані і, тому, страждають проблемою, суть якої в збіжності пошуку в локальних оптимумумах. З появою методів, які базуються на

використанні популяційних алгоритмів, задачі оптимізації можна розв’язувати з досягненням оптимального або субоптимального розв’язків.

Популяційні алгоритми є класом матаевристичних методів оптимізації, які спрямовані на сукупність (популяцію) розв’язків. Вони імітують природні процеси. Популяційні алгоритми переважно класифікують [51] на три основні категорії в залежності від джерел натхнення: еволюція; фізика; рій. Еволюційні алгоритми імітують еволюційні процеси в природі (наприклад, генетичні алгоритми, алгоритм еволюційної мембрани). Кількість алгоритмів на основі рою (наприклад, алгоритм бджіл, оптимізатор сірого вовка) більша, ніж еволюційних. Третій клас алгоритмів натхненний фізичними явищами в природі (наприклад, алгоритм гравітаційного пошуку, оптимізація хімічних реакцій). Відомі, також, популяційні алгоритми [51] з різними джерелами натхнення: неживою природою; живою природою. Обмежитись декількома популяційними алгоритмами при розв’язуванні задач оптимізації складно, оскільки не існує одного чи декількох популяційних алгоритмів оптимізації для розв’язування всіх оптимізаційних завдань [51].

Розглянемо поширені популяційні алгоритми, які використовують для розв’язування оптимізаційних задач.

В роботі [58] запропоновано підхід до постановки та розв’язування задачі оптимального вибору показників якості продукції в системі «виробник–споживач» на основі нечітких когнітивних карт і генетичного алгоритму з урахуванням інтересів як виробника, так і споживача. Критерієм оптимізації є максимальна наближеність між привабливістю товару та бажанням його придбати. Контрольованими змінними є рівні показників виробника та споживача. Обмеженнями є домовленості щодо необхідних рівнів показників, які спільні для виробника і споживача. Нечіткі когнітивні карти використовуються для побудови залежності, зокрема з відображенням в цільовій функції, а оптимальні рішення знаходяться за допомогою генетичного алгоритму.

В роботі [59] запропоновано генетичний алгоритм пошуку набору правил для формування нечіткої бази знань, яка збалансована за критеріями точності та компактності. Відмінністю цього генетичного алгоритму від відомих реалізацій є введення в постановку задачі оптимізації лінійного обмеження, яке задає рівень

компенсації точності моделі її компактністю, що дає змогу наближувати область допустимих розв’язків до парето-фронту.

В роботі [45] здійснено впровадження процедури кластеризації масивів даних на основі вдосконаленого алгоритму сірого вовка. Перевагою запропонованого підходу є скорочення часу вирішення задач оптимізації в умовах перекриття кластерів. Особливістю запропонованого методу є велика швидкість, завдяки тому, що увесь масив обробляється лише один раз, тобто усувається необхідність багатоепохового самонавчання, реалізованого в традиційних алгоритмах нечіткої кластеризації. В роботі [46] запропоновано метод кластеризації масивів даних на основі комбінованої оптимізації функцій щільності розподілу та еволюційного методу котячих роїв. Перевагою запропонованого підходу є скорочення часу вирішення оптимізаційних задач в умовах перекриття кластерів. В роботі [47] розглянуто проблему нечіткої кластеризації великих масивів даних, які надсилаються на обробку як у пакетному, так і в онлайн режимах, на основі достовірного підходу. Щоб знайти глобальний екстремум цільової функції достовірної нечіткої кластеризації, була введена модифікація ройового алгоритму зграй божевільних кішок, яка поєднує в собі переваги еволюційних алгоритмів і глобального випадкового пошуку. Показано, що різні режими пошуку генеруються єдиною математичною процедурою, деякі випадки якої є відомими алгоритмами як локальної, так і глобальної оптимізації. Запропонований характеризується великою швидкістю та надійністю в задачах багатоекстремальної нечіткої кластеризації. Таким чином, для однієї типової задачі щодо кластеризації в масивах даних застосовано різні популяційні алгротми, які дали змону ефективно розв’язати задачі оптимізації.

В роботі [48] подано різні прикладні методи комбінаторної оптимізації, а також розглянуто питання формалізації, класифікації та розв’язування задач метаевристичними методами.

В роботі [49] запропонлвано новий природний алгоритм, в якому імітується механізм полювання мурашиних левів у природі. Реалізовано п’ять основних кроків полювання на здобич, таких як випадкова хода мурах, створення пасток, захоплення мурах у пастки, ловля здобичі та повторне будівництво пасток.

В роботі [50] запропонована нова метаевристика натхненна сірими вовками. Алгоритм імітує ієрархію лідерства та механізм полювання сірих вовків у природі.

Чотири типи сірих вовків, такі як альфа, бета, дельта та омега, використовуються для імітації ієрархії лідерства. Крім того, реалізовано три основні етапи полювання, пошуку здобичі, оточення здобичі та нападу на здобич. Потім алгоритм перевіряється на основі 29 відомих тестових функцій, а результати перевіряються шляхом порівняльного дослідження за допомогою оптимізації роєм частинок, алгоритму гравітаційного пошуку, диференціальної еволюції, еволюційного програмування і стратегії еволюції. Алгоритм здатний забезпечити конкурентоспроможні результати порівняно з відомими метаевристичними методами.

В роботі [51] запропоновано нову парадигму оптимізації, яка натхненна живою природою, і називається алгоритмом оптимізації молі і полум'я. Основним джерелом натхнення для цього оптимізатора є метод навігації метеликів молі у природі, який називається поперечною орієнтацією. Метелики літають вночі, зберігаючи фіксований кут відносно місяця, що є дуже ефективним механізмом для подорожей по прямій лінії на великі відстані. Однак ці комахи потрапляють в пастку на смертоносну спіраль навколо штучного світла. В роботі математично змодельовано цю поведінку для виконання оптимізації. Алгоритм порівнюється з іншими відомими природними алгоритмами на 29 тестових і 7 реальних інженерних задачах. Статистичні результати порівняльних функцій показують, що цей алгоритм здатний забезпечити конкурентоспроможні результати. Крім того, результати реальних задач демонструють переваги цього алгоритму у вирішенні складних задач із обмеженими та невідомими просторами пошуку. В роботі [52] розглянуто проблеми автоматизації побудови класифікаційних дерев на основі схеми відбору розгалужених ознак. Об'єктом дослідження є класифікаційні дерева. Предметом дослідження є методи, алгоритми та схеми побудови класифікаційних дерев. Метою роботи є побудова ефективного методу для синтезу моделей дерева класифікації на основі групової оцінки важливості дискретних ознак у межах розгалуженого вибору ознак. Запропоновано метод побудови дерев класифікації, який для заданої навчальної вибірки визначає індивідуальну інформативність груп ознак та їх комбінацій по відношенню до початкового значення функції класифікації. Розроблений метод логічного дерева при побудові наступного вузла дерева класифікації намагається виділити групу найбільш тісно пов'язаних між собою дискретних ознак, що зменшує загальну структурну складність моделі, прискорює обчислення при розпізнаванні

об’єктів на основі моделі, а також підвищує узагальнюючі властивості моделі. Таким чином, запропонований підхід в роботах [51; 52] дає змогу коригувати наступні групи показників. В роботі [53] об’єктом дослідження є методика оцінки показника інформаційної безпеки. Розглянуто проблему уніфікації та спрощення процесів оцінки ступеня інформаційної безпеки з метою зменшення залучення в них людських і матеріальних ресурсів, використовуючи апарат теорії нечітких множин для врахування висновків компетентних експертів. Розроблено нечітку продукційну модель оцінки ступеня інформаційної безпеки, яка базується на використанні експертних знань та методів нечіткої логіки. Запропоновано поетапний підхід до визначення потенційних ризиків, їх класифікації за категоріями та розрахунку коефіцієнтів впливу. Створено ітераційний метод оцінки, який дозволяє отримати числовий показник ступеня інформаційної безпеки. Розроблено евристичні правила визначення ефективної оцінки ступеня інформаційної безпеки з урахуванням коефіцієнта критичності та коефіцієнтів впливу різних категорій ризику.

В роботі [54] описано приклад оптимізації лінії виробництва двигуна на основі моделювання в реальному світі. Оптимізація спрямована на максимізацію використання машин і водночас мінімізацію зв’язаного капіталу шляхом маніпулювання 56 унікальними змінними рішень. Для цього використано метаевристичний алгоритм молі і полум’я, який отримав успішні результати в різних проблемних областях.

В роботі [55] представлено алгоритм оптимізації молі і полум’я. Він застосовується для вирішення складних реальних проблем оптимізації в багатьох областях. У роботі подано широкий огляд варіантів цього алгоритму, включаючи класичну версію, двійкові типи, модифіковані версії, гібридні версії, багатоцільові версії та прикладну частину алгоритму у різних секторах.

В роботі [173] алгоритм молі і полум’я демонструє кращі результати продуктивності порівняно з іншими розглядуваними метаевристичними алгоритмами для вирішення проблем глобальної оптимізації з нелінійними обмеженнями. Однак він все ще страждає від отримання якісного рішення та низької швидкості конвергенції.

В роботі [56] технологічний процес вважається багатоаспектним, якщо має місце ряд дефектів різного типу, які виявляються та усуваються одночасно в процесі

виконання процесу. Якість технологічного процесу оцінюється ймовірністю вихідних нульових дефектів, а також ймовірностями нульових дефектів для кожного з типів дефектів. Завдання оптимізації передбачають вибір такої структури процесу, яка забезпечить необхідний вихідний рівень якості продукції за певних витратних обмежень. Задача оптимізації розв’язується за допомогою генетичних алгоритмів, які дозволяють знаходити наближене до глобального оптимальне рішення.

В роботі [57] розглядається теорія та застосування мурашиних алгоритмів, нові методи дискретної оптимізації на основі моделювання самоорганізованої колонії біологічних мурах. Колонію розглянуто як багатоагентну систему, де кожен агент функціонує незалежно за простими правилами. На відміну від майже примітивної поведінки агентів, поведінка всієї системи виявляється розумною. Мурашинні алгоритми були успішно застосовані для розв’язування багатьох складних задач комбінаторної оптимізації, таких як задача комівояжера, задача маршрутизації транспортного засобу, задача розфарбовування графа, задача квадратичного призначення, задача оптимізації мережного трафіку, задача планування роботи в магазині тощо. Мурашинні алгоритми особливо ефективні для онлайн-оптимізації процесів у розподілених нестаціонарних системах (наприклад, телекомунікаційна мережа). маршрутизація).

В роботі [174] розглянуто гібридний фільтр активної потужності. Щоб отримати його більш точні параметри, запропоновано новий популяційний метод на основі алгоритму молі і полум’я. Його модифікація в тому, що здійснено поділ на групу експлуатації та групу дослідження.

В роботі [175] отримання відповідних параметрів фотоелектричних моделей на основі виміряних даних про струм та напругу фотоелектричної системи є важливим для оцінки, керування та оптимізації фотоелектричних систем. Щоб отримати конкретні параметри фотоелектричних моделей, запропоновано метаевристичний алгоритм оптимізації молі і полум’я та симбіотичної еволюції. Запропонований підхід поділяє популяцію на дві паралельні робочі підгрупи.

В роботі [176] традиційна робота водосховища з виробництва електроенергії в основному передбачає максимізацію виробництва електроенергії та гарантує стабільність енергосистеми. Однак, розглядаючи цілі виробництва електроенергії, можна ігнорувати її вплив на екологічне середовище та навігацію. Створено

багатоцільову модель оптимізації, яка враховує цілі виробництва електроенергії, екології та навігації. Для ефективного розв’язання моделі було запропоновано новий покращений багатоцільовий алгоритм оптимізації молі і полум’я на основі домінування. Щоб підвищити здатність алгоритму оптимізації молі і полум’я подолати потрапляння в локальний оптимум, його було вдосконалено в трьох аспектах: формула оновлення; натхнення лінійної траєкторії польоту молі; стратегія оновлення популяції полум’я. Щоб відрізнити цих індивідів, які не домінують один над одним у домінуванні Парето, було запропоновано домінування в поєднанні з контрольними точками.

В роботі [177] пропонується нова схема навчання для машини екстремального навчання ядра, як базується на використанні стратегії хаотичної оптимізації молі і полум’я. У запропонованій схемі одночасно здійснюється оптимізація параметрів і вибір ознак.

В роботі [178] запропоновано вдосконалений алгоритм оптимізації молі і полум'я, щоб полегшити проблеми передчасної збіжності до локальних мінімумів. З точки зору різноманітності, вага інерції контролю зворотного зв’язку різноманітності введена в оптимізацію полум’я молі, щоб збалансувати використання алгоритму та можливості глобального пошуку. Крім того, додається невелика мутація ймовірності після етапу оновлення позиції для покращення ефективності оптимізації.

В роботі [179] розглянуто алгоритм оптимізації молі і полум'я як новий метаевристичний алгоритм на основі сукупності для вирішення проблем глобальної оптимізації. Генерація полум'я та спіральний пошук є двома ключовими компонентами, які впливають на продуктивність алгоритму. Для покращення різноманітності полум’я та пошукової здатності молі запропоновано вдосконалений алгоритм оптимізації молі і полум’я. Основними особливостями його є такі: полум'я генерується ортогональним опозиційним навчанням; модифіковано механізм оновлення позиції молі з лінійним пошуком і оператором мутації.

В роботі [180] проаналізовано проблему відбору генів, яка пов’язана з великою кількістю генів (релевантних, надлишкових або шумових), що потребують ефективного методу, щоб допомогти у виявленні захворювань. У цій задачі обчислювальна складність зменшується шляхом вибору невеликої кількості генів, але необхідно вибрати відповідні гени, щоб зберегти високий рівень точності. Тому, щоб

знайти оптимальну підмножину генів, представлено алгоритм ройового інтелекту для відбору генів, який називається квантовим алгоритмом оптимізації молі і полум'я, який базується на комбінуванні між квантовими обчисленнями та алгоритмом оптимізації молі і полум'я.

Таким чином, використання популяційних алгоритмів для розв'язування задач оптимізації є перспективним. Серед багатьох популяційних алгоритмів наявний клас таких алгоритмів натхнений живою природою, з якого можуть бути використані алгоритми для синтезу в архітектурі ОСПП, оскільки їх поведінку, тобто підібрані наступні кроки для виконання, буде складно зрозуміти зловмисникам.

1.5 Постановка задачі дослідження

Для розв'язання задачі покращення протидії КА та ЗПЗ в корпоративних мережах шляхом оптимізації кроків ОСПП за рахунок синтезу популяційних алгоритмів в центрах прийняття рішень, потрібно вирішити такі завдання:

1) систематизувати особливості типів КА та ЗПЗ для імплементації їх в ОСПП, методи та системи їх виявлення в корпоративних мережах, зокрема обманні системи, приманки та пастки, а також популяційні алгоритми натхненні живою природою, які можуть бути основою для їх застосування в архітектурі обманних систем для підвищення ефективності вибору ними наступних кроків та їх коригування;

2) розробити моделі КА в корпоративних мережах, в яких врахована особливість щодо наявних в корпоративних мережах реальних та хибних об'єктів для атак та відповідних дій зловмисників, які спрямовані на класифікацію таких об'єктів, що дозволить врахувати їх поведінку в моделі функціонування ОСПП;

3) розробити формальний опис архітектури ОСПП з підсистемою прийняття рішень на основі синтезу в архітектурі популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків та їх коригування при здійсненні КА та дій ЗПЗ;

4) розробити метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі ОСПП для забезпечення їх довготривалого й адаптивного функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

5) розробити метод організації функціонування ОСПП в корпоративних мережах на основі синтезу в їх архітектурі популяційних алгоритмів, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак;

6) розробити метод виявлення атак типу відмова в обслуговуванні у мережах на основі статистичних показників з урахуванням його імплементації в компоненти ОСПП для підвищення достовірності виявлення атак відмова в обслуговуванні;

7) розробити обманну систему з приманками і пастками для виявлення двоцільових КА, зокрема КА типу відмова в обслуговуванні у мережах на основі статистичних показників, для експериментального дослідження таких систем з метою покращення її характеристик та подальшого впровадження для застосування.

1.6 Висновки до першого розділу

Згідно проведеного аналізу існуючих комерційних систем виявлення КА та ЗПЗ, обманних систем, приманок і пасток, типів КА та особливостей популяційних алгоритмів встановлено наявність проблем щодо недостатнього забезпечення безпеки та захисту корпоративних мереж від КА та дій ЗПЗ та за його результатами можна виокремити наступні висновки.

1. Кількість КА та ЗПЗ постійно збільшується та удосконалюється, зокрема найбільш суттєві проблеми з безпекою та захистом інформації та ресурсів залишаються в користувачів, адміністраторів та власників корпоративних мереж.

2. Зловмисники при здійсненні КА на корпоративні мережі прагнуть класифікувати об'єкти, які знаходяться в них, тобто особливістю їх атак стає прагнення досягти розуміння та встановлення типів вузлів, наявність хибних об'єктів для атак, зв'язків між ними тощо.

3. Наявні системи виявлення вторгнень, обманні системи, антивірусні програми, комплексні системи захисту корпоративних мереж є ефективними, особливо коли застосовуються комбіновано на різних рівнях захисту та різних етапах, які можуть проходити КА та ЗПЗ, але з часом особливості їх функціонування стають відомими зловмисникам, що зменшує ефективність їх використання та, відповідно, достовірність виявлення.

4. Методи виявлення та протидії КА і ЗПЗ в корпоративних мережах є ефективними на початкових етапах їх застосування, але з часом зловмисники адаптуються під їх особливості і вони втрачають початкову ефективність.

5. Перспективним напрямом дослідження є розроблення ОСПП для застосування в корпоративних мережах з метою протидії та виявленню КА і ЗПЗ.

6. Для уникнення можливості зі сторони зловмисників до швидкої адаптації щодо ОСПП в корпоративних мережах необхідно забезпечити функціонування таких систем згідно таких методів прийняття рішень, які б унеможливили розуміння принципів їх функціонування та вибору наступних кроків після виникнення певних подій, які викликані впливами КА чи дій ЗПЗ.

7. Формування поведінки ОСПП визначається переліком її наступних кроків, які повинні бути здійснені такими системами самостійно без залучення адміністраторів, а оскільки вони повинні протидіяти зловмисникам, тому перспективним напрямом для забезпечення вибору наступних кроків є використання в архітектурі обманних систем популяційних алгоритмів, що можуть напротязі тривалого часу підтримувати протидію КА обираючи свої кроки та коригуючи їх, тобто здійснюючи пошук рішення переходячи з локального оптимуму до глобального оптимуму.

8. Серед популяційних алгоритмів виділяють клас алгоритмів, які натхненні живою природою і, тому, вони можуть бути синтезовані в архітектурі обманних систем в частині підсистеми прийняття рішень.

9. Найбільш оптимальним для забезпечення функціонування підсистеми прийняття рішень ОСПП серед популяційних алгоритмів натхненних живою природою є алгоритм молі і полум’я, оскільки, крім його відповідності з вибору наступних кроків систем, його особливості відповідають моделям залучення приманок і пасток під час КА та дій ЗПЗ.

Основні результати розділу опубліковані у працях [103; 108; 110; 113; 134-136; 168-171]

РОЗДІЛ 2
МОДЕЛІ ТА АРХІТЕКТУРА СИСТЕМ ВИЯВЛЕННЯ КОМП'ЮТЕРНИХ АТАК У КОРПОРАТИВНИХ МЕРЕЖАХ З ПРИМАНКАМИ ТА ПАСТКАМИ НА ОСНОВІ ПОПУЛЯЦІЙНИХ АЛГОРИТМІВ

Організація систем виявлення комп'ютерних атак та зловмисного програмного забезпечення в корпоративних мережах потребує удосконалення на рівні архітектури, особливо при використанні приманок і пасток для покращення протидії зловмисникам. Корпоративні мережі з великою кількістю вузлів та посиленим захистом за рахунок підтримки великої кількості приманок та пасток потребують швидкого та ефективного управління ними.

Приманки можуть використовуватись не тільки для відволікання зловмисників та неправильного розуміння ними хибних об'єктів, але й для того, щоб при проведенні зловмисниками атак перенаправити або продублювати трафік на приманки, які є невидимими зловмисникам, з подальшим аналізом трафіку та прийняття відповідних дій. Також, при цьому можуть бути активізовані пастки в поєднанні з приманками. Приманки можуть бути різними та мати різну будову, тому система повинна оцінити їх та здійснити вибір найбільш ефективних з них. Для здійснення вибору приманок з пастками можуть бути застосовані різні стратегії. Але в цьому процесі важливим є вибір саме таких варіантів приманок з пастками, які б могли в процесі проведення атак, привести не тільки до однієї певної приманки. Взагалі приманки з пастками можуть бути досліджені зловмисниками і дуже багато з них не мають повноцінних наборів правильних відповідей, які повинні імітувати, і тому, вони можуть бути зрозумілими зловмисникам через їх виявлення, а в частих випадках і використані ними для зловмисних дій. Тому, керування приманками з пастками має бути організовано з врахуванням спроможності досліджувати комп'ютерні атаки сукупністю таких об'єктів за певними стратегіями. Перспективним напрямом для дослідження таких стратегій є стратегії, що базуються на використанні популяційних алгоритмів в архітектурі систем виявлення комп'ютерних атак з приманками і пастками.

2.1 Модель комп’ютерних двоцільових атак в корпоративних мережах з системами приманок і пасток

Зловмисники в процесі організації та проведення атак на корпоративні мережі спрямовуються на використання вразливостей систем, викрадення конфіденційної інформації і порушення штатного функціонування корпоративних інформаційних та комп’ютерних систем і мереж. Для організації атак застосовують типові інструменти з використанням автоматизованих сценаріїв та засобів для здійснення атак на велику кількість об’єктів в глобальній мережі або зловмисник використовує клавіатуру для цілеспрямованої початкової спроби доступу і виконання подальших дій. Розглядатимемо мережні атаки винятково на корпоративні мережі і з врахуванням того, що вони можуть бути пасивними і активними, тобто КА, а також з використанням ЗПЗ і спрямованими на перехоплення трафіку. До пасивних атак віднесемо сканування портів, а до активних атак - DDoS-атаки і атаки на основі паролів. Тому, будемо розглядати КА як мережні атаки і атаки з використанням ЗПЗ.

В корпоративних мережах можуть бути розміщені різні системи та засоби протидії комп’ютерним атакам і зловмисним діям. Корпоративні мережі можуть мати різні стани захищеності ресурсів та вузлів. Розглядатимемо ті з них, які мають високий рівень захищеності, для забезпечення якого в мережах обов’язково встановлені системи з приманками і пастками. Така організація захисту не є типовою та поширеною, але стає все більш актуальною із зростанням та появою нових типів комп’ютерних атак і зловмисних загроз. Застосування окремих приманок і пасток або систем з приманками і пастками вимагає додаткових витрат від власників корпоративних мереж, але такі витрати покращують рівень захисту і безпеку в корпоративних мережах порівняно з корпоративними мережами без них. Для досягнення ефекту з протидії комп’ютерним атакам загальноприйнято встановлювати не менше п’ятнадцяти відсотків приманок від загальної кількості вузлів в комп’ютерних мережах. Така кількість приманок потребує системи з керування їх застосуванням. Самі приманки і пастки можуть бути однаковими, але можуть бути і повністю різними за будовою, призначенням, місцем встановлення, способами та етапами застосування тощо. Приманки можуть містити пастки або застосовуватись разом з пастками чи без них. Складними завданнями із застосування приманок і

пасток є спрямування зловмисника на них. Не завжди приманки можуть бути активізовані під час здійснення комп'ютерних атак. Все це необхідно враховувати в архітектурі систем з приманками і пастками для забезпечення їх застосування.

Приманки мають різну будову в залежності від призначення і застосування. Їх можна поділяти на дві групи: низького рівня взаємодії; високого рівня взаємодії. Або на три такі групи в залежності від рівнів взаємодії: низького; середнього; високого. Приманки з низьким рівнем взаємодії мають обмежену симуляцію обслуговування та невеликі експлуатаційні витрати. Приманки з середнім рівнем взаємодії забезпечують розширене моделювання, збір цільового профілю пристрою. До приманок з середнім рівнем взаємодії відносять ті з них, які забезпечують функції послуг від тих, які надають приманки з високим рівнем, до послуг, які надають приманки з низьким рівнем взаємодії. Приманки з високим рівнем взаємодії забезпечують майже повноцінні послуги і, тому, виступають цілями в корпоративних мережах. Всі приманки забезпечують моделювання послуг або певних систем за рахунок закладеного в них механізму обману. Але до якого б рівня вони не відносились і скільки їх не було б в корпоративних мережах, зловмисники при проведенні атак можуть встановити несправжність цих об'єктів в корпоративних мережах, а також використати це в своїх цілях зі зворотньою до їх призначення метою.

Для дослідження корпоративних мереж та їх ресурсів зловмисники можуть використовувати доступні стандартні засоби і на їх основі, відповідно, реалізовувати активні чи пасивні методи дослідження або їх поєднання. При реалізації активних методів використовують набори команд і засобів, які взаємодіють із системою, для виклику конкретних реакцій, щоб сформувати характеристику системи. При застосуванні пасивних методів дані про систему отримують з мережного трафіку або метадані із зовнішніх баз даних тощо. Таким чином, для вивчення корпоративних мереж зловмисники можуть використовувати різні методи окремо або поєднувати їх. Якщо корпоративні мережі містять системи з приманками та пастками або окремі приманки з пастками, то зловмисники теж можуть досліджувати їх наявність під час здійснення початкових етапів атак. Реалізація приманок як повноцінних об'єктів є складним завданням і, тому, для його спрощення розробники частину функцій або елементів повноцінних об'єктів не реалізовують. Часто реалізовані функції чи елементи приманок можуть бути примітивними. Тому, зловмисники в процесі

дослідження корпоративних мереж або при проведенні активної фази атак досліджують та встановлюють компоненти систем з приманками та пастками. Це понижує ефективність такого підходу до протидії КА та зловмисним проявам і потребує вдосконалення архітектури систем з приманками та пастками для забезпечення їх ефективної організації, враховуючи обмежені спроможності окремих приманок та пасток.

Для КА загально прийнято, що вони реалізуються переважно в три етапи: розвідка; атака; завершення атаки. Кожен етап КА є важливим. На етапі розвідки зловмисники обов'язково досліджують системи і засоби захисту корпоративних мереж, зокрема і оточення визначених для атак об'єктів. Об'єктами комп'ютерних атак є комп'ютерні системи загалом та їхні компоненти, інформаційні ресурси, дані, програмне та технічне забезпечення, а також комп'ютерні пристрої та мережі. Метою атак зловмисників на корпоративні мережі можуть бути спроби порушити конфіденційність інформації для отримання несанкціонованого доступу до неї, порушити цілісність інформації для її зміни чи знищення, а також порушити доступність інформації, здійснюючи несанкціоноване блокування доступу до системи або даних. Всі ці цілі зловмисників є актуальними і можливими для реалізації в контексті корпоративних мереж з різними ступенями захисту.

Задамо модель комп'ютерних атак з урахуванням цілей зловмисників, що стосуються порушення конфіденційності, цілісності та доступності інформації в корпоративних мережах, так:

$$M_{KA,1} = \langle R_{kn}, R_p, G_{kn,p}, K_{a,1}, G_{ka,1} \rangle, \tag{2.1}$$

де R_{kn} – множина об'єктів корпоративних мереж, які можуть піддані комп'ютерним атакам; R_p – множина хибних об'єктів для комп'ютерних атак, тобто приманок і приманок з пастками; $G_{kn,p}$ – граф зв'язків між об'єктами корпоративних мереж та хибними об'єктами для комп'ютерних атак; $K_{a,1}$ – множина векторів комп'ютерних атак на об'єкти корпоративних мереж; $G_{ka,1}$ – граф зв'язків джерел комп'ютерних атак зловмисників та потенційних об'єктів корпоративних мереж з урахуванням векторів комп'ютерних атак.

Комп'ютерні атаки, які задано моделлю $M_{KA,1}$ за формулою (2.1), віднесемо до одноцільових атак за умови, що вони спрямовані винятково на реальні об'єкти

корпоративних мереж і не спрямовані на хибні об’єкти. Таким чином, якщо зловмисники спрямовуватимуть свої атаки тільки на реальні об’єкти корпоративних мереж, серед яких можуть бути і хибні об’єкти, то такі комп’ютерні атаки вважатимемо одноцільовими.

В контексті використання в корпоративних мережах систем з приманками і пастками КА можуть бути спрямовані не тільки на визначені об’єкти, але й на приманки та пастки з метою їх використання в процесі продовження виконання КА. Для цього на етапі розвідки такі об’єкти можуть бути виявлені і за результатами виявлення скориговані цілі та моделі КА. Таким чином, КА може мати дві цілі: запланований об’єкт для атаки; додатковий (хибний) об’єкт для атаки (приманки). Такі КА будемо називати двоцільовими і введемо їх загальну модель так:

$$M_{KA,2} = \langle R_{kn}, R_p, G_{kn,p}, K_{a,2}, G_{ka,2} \rangle, \quad (2.2)$$

де R_{kn} – множина об’єктів корпоративних мереж, які можуть піддані комп’ютерним атакам; R_p – множина хибних об’єктів для комп’ютерних атак, тобто приманок і приманок з пастками; $G_{kn,p}$ – граф зв’язків між об’єктами корпоративних мереж та хибними об’єктами для комп’ютерних атак; $K_{a,2}$ – множина векторів комп’ютерних атак на об’єкти корпоративних мереж та хибні об’єкти для комп’ютерних атак; $G_{ka,2}$ – граф зв’язків джерел комп’ютерних атак зловмисників, потенційних та хибних об’єктів корпоративних мереж з урахуванням векторів комп’ютерних атак.

В формулі (2.1) множина R_p може бути порожньою або не порожньою. А в формулі (2.2) ця множина R_p завжди є непорожньою. Множини R_{kn} та R_p в формулах (2.1) та (2.2) є однаковими. Граф зв’язків $G_{kn,p}$ між об’єктами корпоративних мереж та хибними об’єктами для комп’ютерних атак в формулах (2.1) та (2.2) є однаковим і відображає зв’язки між в корпоративних мережах в контексті взаємодії для забезпечення захисту і безпеки.

В формулі (2.1) граф зв’язків $G_{ka,1}$ джерел комп’ютерних атак зловмисників та потенційних об’єктів корпоративних мереж з урахуванням множини векторів $K_{a,1}$ комп’ютерних атак на об’єкти корпоративних мереж не враховує спрямування зловмисників на пошук хибних об’єктів атак, тобто приманок і пасток. В формулі (2.2) граф зв’язків $G_{ka,2}$ джерел комп’ютерних атак зловмисників, потенційних та хибних об’єктів корпоративних мереж з урахуванням множини векторів $K_{a,2}$

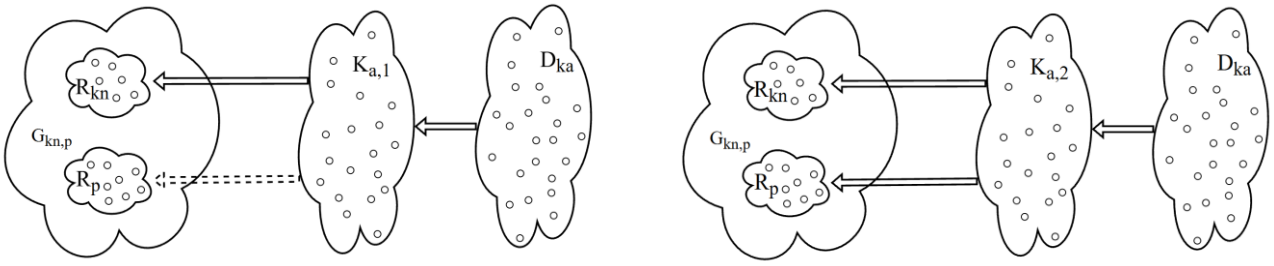
комп’ютерних атак на об’єкти корпоративних мереж враховує спрямування зловмисників на пошук хибних об’єктів атак, тобто приманок і пасток, з метою їх подальшого використання. Таким чином, в другому випадку модель КА враховує дві цілі. Таку потенційну поведінку зловмисників повинні враховувати при проєктуванні захисту корпоративних мереж із використанням приманок і пасток як хибних об’єктів для КА.

Логіка дій зловмисника в першому випадку, який задано за формулою (2.1), полягає в тому, що зловмисник в процесі здійснення атаки не здійснює пошук хибних об’єктів для атак в корпоративних мережах. Він спрямований безпосередньо на досягнення мети і тому він в процесі здійснення атаки може не досягти мети через технічну неспроможність або попавши на хибні об’єкти для атак. Якщо він попадає на хибні об’єкти для атак, то в даному випадку він може знати або не знати про такий розвиток подій. Але при цьому він не розглядав їх наявність як засобів для досягнення мети. В другому випадку зловмисник в своїй моделі атаки закладає можливість поділу об’єктів корпоративних мереж на основні об’єкти та хибні об’єкти для атак. Модель такої атаки зі сторони зловмисника має на меті розроблення додаткового етапу атаки, що полягає в класифікації об’єктів в корпоративних мережах для покращення результативності при здійсненні атаки. Подальші дії зловмисника будуть зосереджені на використанні знайдених ним хибних об’єктів для атак в корпоративних мережах для удосконалення планованої атаки, що потребуватиме від нього розроблення додаткового етапу КА. Таким чином, зловмисник після здійснення власної класифікації об’єктів в корпоративних мережах на два класи повинен здійснювати окремі дії з дослідження хибних об’єктів для атак. З іншої сторони, тобто зі сторони хибних об’єктів для атак, особливо якщо вони інтелектуальні, можуть бути дії, які будуть спонукати зловмисника до витрат ресурсів і часу. Крім того, зловмиснику буде складно провести чітку класифікацію реальних та хибних об’єктів в корпоративних мережах. Це теж повинно бути враховано в моделі захисту корпоративних мереж, а також в моделі зловмисних дій.

Відмінність між обома моделями поведінки зловмисника є суттєвою. Але при цьому складно відокремити особливість, яка характеризує другий випадок. Тому, в першому випадку можна здійснити припущення, що зловмисник цілеспрямований винятково щодо результатів атаки і не поділяє об’єкти корпоративних мереж, бо це

може бути не пов’язано з його кваліфікацією, а саме винятково із метою атаки. Вважатимемо, що кваліфікація зловмисника в обох випадках є однаковою. Тоді, наприклад, метою атаки в першому випадку можуть бути відповідно до властивостей інформації такі загрози: цілісності, що пов’язані із знищенням або модифікацією інформації; загрози доступності, які пов’язані із блокуванням чи знищенням. А в другому випадку до цих загроз для інформації можна додати ще загрози для конфіденційності, які полягають в спробах здійснити несанкціонований доступ, витік або розголошення. Ці загрози для інформації вибілено в другому випадку, бо для їх успішного забезпечення зловмиснику потрібно буде після проведення атаки продовжувати здійснення комунікації з об’єктами корпоративних мереж, тому порівняно з першим випадком обсяг робіт, витрати часу та ресурсів будуть більшими. Крім того, розглядувані загрози для інформації в другому випадку будуть включати всі загрози і першого випадку. В першому випадку теж можуть бути метою КА реалізовані загрози для конфіденційності, але оскільки вони спрямовані на несанкціонований доступ, витік або розголошення, то через наявність хибних об’єктів атак та довшу тривалість для забезпечення досягнення мети щодо порушення конфіденційності вони можуть мати суттєво менший успіх. Таким чином, для забезпечення безпеки та захисту корпоративних мереж з використанням хибних об’єктів атак потрібно враховувати, що модель КА переважно буде двоцільовою.

Графи зв’язків $G_{ka,1}$ та $G_{ka,2}$ зобразимо на рис. 2.1 у відповідності до моделей, які задано формулами (2.1) і (2.2).



а) одна ціль б) дві цілі

Рисунок 2.1 – Графічне подання компонентів моделей $M_{KA,1}$ та $M_{KA,2}$

На рис. 2.1 для КА виділено окремо множину джерел атак D_{KA}, з елементів якої зловмисники проводять свої атаки, задані елементами в множинах $K_{a,1}$ та $K_{a,2}$. Джерел КА може бути декілька, а може бути одне. В контексті моделей $M_{KA,1}$ та $M_{KA,2}$

це не є принциповим, оскільки важливим є тільки наявність мети у зловмисників та відповідних засобів для пошуку і здійснення КА безпосередньо на хибні об'єкти для атак в корпоративних мережах. Отже, відношення джерел КА до об'єктів корпоративних мереж можуть бути «один до одного», «один до всіх», «всі до одного» та «всі до всіх», якщо розглядати елементи множин R_{kn} та R_p як окремі різні об'єкти. Таких об'єктів є небагато в контексті саме різних, тоді варіантів відношень джерел КА до них теж буде небагато. Варіантів доступу до об'єктів може бути багато, а самих об'єктів, які цікавлять зловмисників в корпоративних мережах небагато. Таким чином, при здійснені КА на корпоративні мережі з метою пошуку та взяття під контроль або для нейтралізації хибних об'єктів для КА формується нова друга ціль, реалізація якої відрізняється віл цілі щодо основних об'єктів для КА тим, що може бути здійснена типовими підходами для елементів множини R_p.

Деталізуємо множини в формулах (2.1) та (2.2) їх елементами для задання відповідності саме між елементами. Нехай $R_{kn} = \{r_{kn,1}, r_{kn,2}, \dots, r_{N_{R_{kn}}}\}$, де $N_{R_{kn}}$ – кількість об'єктів корпоративних мереж, на які можуть бути спрямовані КА зловмисників, та $R_p = \{r_{p,1}, r_{p,2}, \dots, r_{N_{R_p}}\}$, де N_{R_p} – кількість хибних об'єктів для КА в корпоративних мережах. Граф зв'язків $G_{kn,p}$ між об'єктами корпоративних мереж та хибними об'єктами для комп'ютерних атак задамо так:

$$G_{kn,p} = \{(a,b) | \; if \; f_{G_{kn,p}}(a,b) = 1, a \neq b, \forall a \in R_{kn} \cup R_p, \forall b \in R_{kn} \cup R_p, R_{kn} \cap R_p = \emptyset\}, \quad (2.3)$$

де $f_{G_{kn,p}}(a,b)$ – бітова функція, що відображає зв'язок між елементами $a \in R_{kn} \cup R_p$ та $b \in R_{kn} \cup R_p$, і її значення визначаються так, що при наявності зв'язку між елементами $f_{G_{kn,p}}(a,b) = 1$, а при відсутності $f_{G_{kn,p}}(a,b) = 0$.

За формулою (2.3) встановлюється топологія між об'єктами в корпоративних мережах, які можуть бути піддані КА, включно з хибними об'єктами для КА. Частина елементів множин R_{kn} та R_p можуть розміщені в одних і тих же комп'ютерних станціях в корпоративних мережах. Але елементи множин R_{kn} та R_p не можуть належати обом множинам одночасно, тобто $R_{kn} \cap R_p = \emptyset$. Елементи a, b не можуть бути однаковими в формулі (2.3), бо тоді такий випадок відповідатиме ізольованій вершині з петлею в графі $G_{kn,p}$ і, відповідно, для такого елементу зв'язку з рештою

об’єктів не буде, включно з підсистемою організації та взаємодії між елементами, що неможливо згідно визначення елементів множин R_{kn} та R_p. Множини R_{kn} та R_p формують два класи об’єктів.

Визначимо елементами множину векторів $K_{a,1}$ комп’ютерних атак на об’єкти корпоративних мереж так: $K_{a,1} = \{k_{a,1,1}, k_{a,1,2}, \dots, k_{a,1,N_{K_{a,1}}}\}$, де $N_{K_{a,1}}$ – кількість векторів КА, які можуть здійснити зловмисники. Також, визначимо елементами множину векторів $K_{a,2}$ комп’ютерних атак на об’єкти корпоративних мереж та хибні об’єкти для комп’ютерних атак так: $K_{a,2} = \{k_{a,2,1}, k_{a,2,2}, \dots, k_{a,2,N_{K_{a,2}}}\}$, де $N_{K_{a,2}}$ – кількість векторів КА, які можуть здійснити зловмисники. Згідно цих визначень та формул (2.1) і (2.2) множина $K_{a,2}$ буде включати множину $K_{a,1}$, тобто множина $K_{a,1}$ є підмножиною множини $K_{a,2}$. Тоді, частина векторів КА з множини $K_{a,2}$ буде стосуватись винятково хибних об’єктів для КА в корпоративних мережах.

Граф зв’язків $G_{ka,1}$ джерел комп’ютерних атак зловмисників та потенційних об’єктів корпоративних мереж з урахуванням множини $K_{a,1}$ векторів комп’ютерних атак у формулі (2.1) задамо так:

$$G_{ka,1} = \{(a,b) | \, if \; f_{G_{ka,1}}(a,b) = 1, a \neq b, \forall a \in K_{a,1}, \forall b \in R_{kn}, K_{a,1} \cap R_{kn} = \emptyset\}, \quad (2.4)$$

де $f_{G_{ka,1}}(a,b)$ – бітова функція, що відображає зв’язок між елементами $a \in K_{a,1}$ та $b \in R_{kn}$, і її значення визначаються так, що при наявності зв’язку між елементами $f_{G_{ka,1}}(a,b) = 1$, а при відсутності $f_{G_{ka,1}}(a,b) = 0$.

Граф зв’язків $G_{ka,2}$ джерел комп’ютерних атак зловмисників, потенційних та хибних об’єктів корпоративних мереж з урахуванням векторів комп’ютерних атак згідно множини $K_{a,2}$ векторів КА у формулі (2.2) задамо так:

$$G_{ka,2} = \{(a,b) | \, if \; f_{G_{ka,2}}(a,b) = 1, a \neq b, \forall a \in K_{a,2}, \forall b \in R_{kn} \cup R_p,$$
$$R_{kn} \cap R_p = \emptyset, K_{a,2} \cap R_p = \emptyset, K_{a,2} \cap R_{kn} = \emptyset\}, \quad (2.5)$$

де $f_{G_{ka,2}}(a,b)$ – бітова функція, що відображає зв’язок між елементами $a \in K_{a,2}$ та $b \in R_{kn} \cup R_p$, і її значення визначаються так, що при наявності зв’язку між елементами $f_{G_{ka,2}}(a,b) = 1$, а при відсутності $f_{G_{ka,2}}(a,b) = 0$.

КА згідно заданого графу $G_{ka,2}$ в формулі (2.5) порівняно із заданням графу $G_{ka,1}$ в формулі (2.4) можуть бути здійснені безпосередньо на елементи множини R_{kn}, на елементи множини R_p та на елементи множини R_{kn} через елементи множини R_p або з їх залученням. Наприклад, на рис. 2.2 зображено ці три варіанти за умови вирішення завдань атаки від джерела до об'єктів. На рис. 2.3 зображено ці ж варіанти за умови встановлення взаємодії джерела атаки з об'єктами корпоративних мереж. Якщо ж розглядати граф $G_{ka,1}$, то можливий лише один варіант здійснення КА, тобто безпосередньо на елементи множини R_{kn}. Якщо серед об'єктів при здійсненні КА в цьому випадку з графом $G_{ka,1}$ будуть траплятись хибні об'єкти з множини R_p, то зловмисниками вони можуть прийняті як важливі об'єкти в корпоративних мережах або будуть відхилені. Якщо вони будуть прийняті як важливі об'єкти, то протягом певного часу КА активує приманки чи пастки, що забезпечить для зловмисників втрату часу та планованого результату КА. В цьому випадку зловмисник не використовує засоби проведення КА безпосередньо на хибні об'єкти для КА в корпоративних мережах.

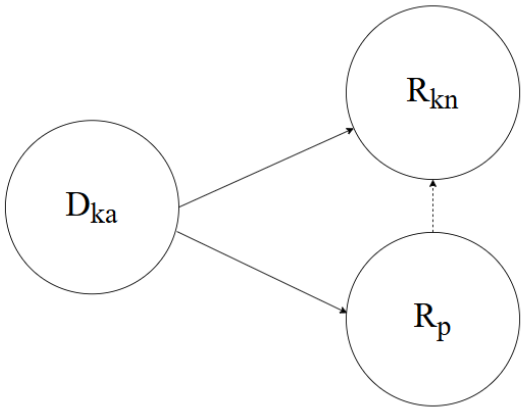


Рисунок 2.2 – Граф трьох варіантів КА згідно формули (2.5) на основі односторонніх зв'язків

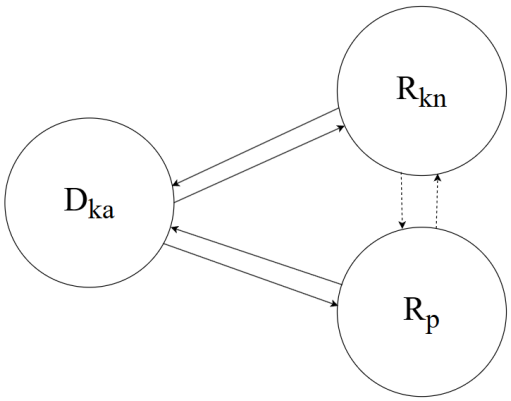


Рисунок 2.3 – Граф трьох варіантів КА згідно формули (2.5) на основі багатосторонніх зв'язків

Елементи множин R_{kn}, R_p, D_{KA} після встановлення відповідних з'єднань, тобто деталізації КА, задамо матрицями інцидентності. В результаті отримаємо базу поведінкових сигнатур КА. Згідно формули (2.2) визначимо місце особливості двоцільових КА серед типових етапів КА.

1. Етап розвідки, на якому зловмисник здійснює збір даних з відкритих джерел, шляхом зондування комп’ютерних мереж або перехопленням мережного трафіку для аналізу.

2. Етап класифікації об’єктів корпоративних мереж з метою виділення реальних та хибних об’єктів корпоративних мереж.

3. Етап доставлення спеціалізованого ЗПЗ із застосуванням різних способів та засобів на реальні та хибні об’єкти корпоративних мереж.

4. Етап приховування КА та доставлення спеціалізованого ЗПЗ.

5. Етап використання виявлених вразливостей в системі корпоративних мереж, зокрема і хибних об’єктів для атак.

6. Етап встановлення спеціалізованого ЗПЗ на реальні та/або хибні об’єкти корпоративних мереж.

7. Етап встановлення зв’язку між джерелом атаки та об’єктами корпоративних мереж.

8. Етап використання зловмисником контрольованих хибних об’єктів корпоративних мереж для розвідки реальних об’єктів.

9. Проведення внутрішньої розвідки в корпоративних мережах.

10. Етап обходу внутрішнього захисту комп’ютерних станцій корпоративних мереж.

11. Етап використання виявлених вразливостей комп’ютерних станцій корпоративних мереж.

12. Етап підвищення привілеїв.

13. Етап доступу до облікових даних.

14. Етап проведення розвідки згідно отриманих внутрішніх даних.

15. Етап обходу внутрішнього захисту на досягнутому об’єкті корпоративних мереж.

16. Етап використання виявлених вразливостей на досягнутому об’єкті корпоративних мереж.

17. Етап здійснення впливу на об’єкті, зокрема хибному об’єкті, корпоративних мереж, що включає також отримання та фіксування контролю над ним.

18. Етап завершення КА, який може мати варіант повного завершення або завершення в контексті утримання комп'ютерних станцій корпоративних мереж під контролем.

Таким чином, поділ зловмисниками на реальні та хибні об'єктів корпоративних мереж збільшує кількість етапів проведення КА, але при цьому дає їм можливість ефективніше їх здійснювати, залучаючи також для атак наявні хибні об'єкти над якими вони отримали контроль.

У випадку неправильної класифікації об'єкту, що відноситься до хибних об'єктів для атак, і за наявності в нього високого ступеня інтеграції зловмисник може отримати проблеми з проведенням КА. Такий хибний об'єкт може містити різні шарі обману, зокрема такі, що імітують мережу, чи систему, чи програмне забезпечення, чи дані. При цьому такий хибний об'єкт може комунікувати зі зловмисником, вводячи його в оману та в результаті спонукає витрачати час та ресурси. Така модель взаємодії повинна бути врахована зловмисником і розробниками/адміністраторами систем безпеки та захисту корпоративних мереж. Наприклад, для зловмисників це можуть бути етапи 3, 7, 8, на яких зловмисник вважає, що він контролює хибний об'єкт для атак чи реальний об'єкт, а реально об'єкт надає йому певний шар обману.

Тому, розроблена модель двоцільових КА включає особливості в захисті корпоративних мереж, які пов'язані із використанням хибних об'єктів для атак та їх включенням в сценарій КА з виділенням певних етапів. Це дає змогу зловмиснику не тільки здійснювати поділ на реальні та хибні об'єкти в корпоративних мережах, але й згідно розроблюваного сценарію КА залучати хибні об'єкти, які взяті під його контроль, для виконання своїх подальших дій. Такі дії зловмисника повинні бути враховані розробниками систем безпеки та захисту корпоративних мереж, в яких використовуються додатково хибні об'єкти для атак. Організація керування і взаємодії таких хибних об'єктів для атак та їх внутрішня архітектура повинні враховувати дії зловмисника, які базуються на розробленій загальній моделі двоцільових КА, що потребує розроблення нової архітектури системи керування та взаємодії хибних об'єктів для атак в корпоративних мережах з врахуванням її таких особливостей.

Розроблена модель двоцільових КА базується на поділі КА на етапи з виділенням етапів, на яких відбувається класифікація реальних та хибних об'єктів в

корпоративних мережах. В загальному підході позиціонування двоцільових КА через поділ на етапи відповідає класичній моделі «Kill Chain» (Lockheed Martin Cyber Kill Chain), в якій виділено вісім етапів проведення КА. Восьмий етап в моделі «Kill Chain» відповідає винагороді зловмисника і може не розглядатись. Тому, кількість етапів або фаз кіберланцюга може містити в цій моделі сім етапів. Але модель «Kill Chain» є лінійною, бо вважається що всі етапи виконуються послідовно, що є недоліком, бо зловмисники можуть її обійти виконуючи етапи не послідовно і розгалужуючи їх та поділяючи на окремі підетапи. Адміністратори корпоративних мереж, які забезпечують захист та безпеку в мережах, при орієнтації захисту саме на відбиття атак, що відповідають моделі «Kill Chain», втрачають переваги над зловмисниками. Також, ця модель не дуже ефективна проти внутрішніх загроз і атак, які не пов’язані із ЗПЗ. Розроблена модель двоцільових КА удосконалена порівняно з моделлю «Kill Chain» щодо відсутності в ній лінійності та враховує цілеспрямованість зловмисників щодо класифікації об’єктів корпоративних мереж.

Багатоетапність в проведенні КА вимагає від адміністраторів корпоративних мереж враховувати поведінку зловмисників. Тому, в розвиток моделі «Kill Chain» було запропоновано модель MITRE ATT&CK (Adversarial Tactics, Techniques, and Common Knowledge), в якій збільшено кількість етапів кіберланцюга до чотирнадцяти. В розробленій моделі двоцільових КА їх поділено на вісімнадцять етапів. Крім того, модель MITRE ATT&CK фокусується на індикаторах компроментації, які класифіковані в піраміді болю зловмисника (David Bianco), та пропонує перелік тактик, методів та процедур зловмисників для ускладнення дій зловмисникам зі сторони користувачів. Запропонована модель двоцільових КА передбачає її застосування в технологіях активного захисту, зокрема в обманних системах з приманками і пастками, що відповідає засобам та системам протидії зловмисникам, які віднесені до найвищого рівня піраміди болю зловмисника.

Таким чином, розроблена модель двоцільових КА в корпоративних мережах поділяє ланцюг кіберзагроз на вісімнадцять етапів, допускає їх нелінійне виконання зловмисниками, враховує особливості поведінки зловмисників щодо здійснення класифікації реальних та хибних об’єктів, спрямована на наявні індикатори компроментації з піраміди болю зловмисника та є основою для розроблення ОСПП.

2.2 Архітектура систем виявлення ЗПЗ та КА в корпоративних мережах з використанням хибних об’єктів для атак

Хибні об’єкти для атак в корпоративних мережах потребують автоматичного організаційного забезпечення. Його може вирішити система виявлення ЗПЗ та КА, яка повинна функціонувати в корпоративних мережах з метою забезпечення їх безпеки та захисту. Певна кількість хибних об’єктів для атак потребує координації роботи зі сторони центру системи. Також, система повинна мати змогу направляти потенційно зловмисні дії безпосередньо на такі хибні об’єкти. Вони повинні мати змогу до здійснення координації виконання завдань і між собою. Основними проблемами у застосуванні систем з хибними об’єктами для атак є спрямування зловмисників саме на такі об’єкти та однакові реакції системи на тривалі повторювані дії зловмисників під час проведення КА.

Хибні об’єкти для атак в корпоративних мережах можуть бути різними. Зокрема: ізольована система, що імітує корпоративну мережу, без зв’язку із захищеною частиною корпоративних мереж і зі зв’язком з її демілітаризованою зоною; підсистема приманок в корпоративних мережах з різними платформами, сервісами, вразливостями без зв’язку із захищеною частиною корпоративних мереж і зі зв’язком з її демілітаризованою зоною; система приманок і пасток, які розміщені перед основними об’єктами захисту; система приманок і пасток, які розміщені поряд з основними об’єктами захисту та імітують їх; зовнішні приманки і пастки, на які йдуть перенаправлення пакетів, що надсилаються в корпоративні мережі; приманки та пастки безпосередньо в комп’ютерних станціях корпоративних мереж, які імітують порти і сервіси тощо. Також, можуть бути інші хибні об’єкти для атак і особливо їх комбінації різні за призначенням та, відповідно, архітектурою. Крім того, хибні об’єкти для атак можуть бути прихованими для зловмисників, в яких система здійснює аналіз спрямованих на них мережних пакетів тощо, тобто тіньові, або такі, що повинні зацікавити зловмисників і це теж є складним завданням для розробників. Така різноманітність та складність хибних об’єктів для атак, а також удосконалення наявних та розроблення нових, вимагають при застосування їх значної кількості забезпечувати ефективну організацію та їх взаємодію. Тому, потребує розроблення

система, що повинна функціонувати в корпоративних мережах та мати змогу підтримувати функціонування хибних об’єктів для атак.

Такі системи, як правило, повинні бути обманними, щоб не тільки підтримувати функціонування множини хибних об’єктів для атак, але й самим бути такими, які заплутують зловмисників. Особливо це актуально, за наявності двоцільової моделі КА. Тоді, крім хибних об’єктів для атак зловмисник може шукати і систему, яка здійснює керування ними. В таких системах повинні бути реалізовані обманні технології для приховування їх особливостей від зловмисників. Таким чином, може бути сформований ланцюг кіберобману, що буде підтримуватись такими системами і включатиме самі системи та керовані ними хибні об’єкти для атак.

Зловмисники постійно розробляють нові моделі КА та типи ЗПЗ. Таким чином, в операційному середовищі корпоративних мереж та ззовні потенційні загрози не тільки зростають, але й модифікуються. Відповідями на такі зловмисні дії повинні бути нові технології обману або модифікації відомих технологій, а також їх поєднання.

Для формування ланцюга кіберобману необхідно визначити мету кіберобману, сформувати базу загроз на основі розроблення моделей загроз і моделей КА для їх реалізації, забезпечити збереження історичних даних про поведінку системи та КА під час атак, сформувати методи протидії з використанням обманних технологій, розробити моделі обманних технологій та архітектуру обманних систем, в якій потрібно реалізувати моніторинг подій в корпоративних мережах та керування хибними об’єктами і системою, координацію дій та контроль подій і змінювану поведінку системи та її компонентів.

Реалізація обманних технологій в системі повинна забезпечити введення в оману зловмисників щодо поверхні атаки та, відповідно, приховати критично важливі ресурси корпоративних мереж від зловмисників. Досягнення таких цілей дасть змогу збільшити ризик виявлення зловмисників при проведенні ними КА, бо змушуватиме направляти або витрачати ресурси на хибні дії та об’єкти в корпоративних мережах. Наприклад, побудувати ланцюг кіберобману можна здійснюючи реконфігурацію мереж, ресурсів та інструментів захисту. При цьому, необхідно змішувати дійсну та фальшиві поверхні атак.

Таким чином, актуальність полягає у синтезі обманних систем протидії та виявлення ЗПЗ та КА в корпоративних мережах, які повинні бути адаптивними, гнучкими, автономними, здатними до колективної взаємодії та прийняття рішень без потреби залучення адмінстарторів. Одним з напрямів розроблення таких обманних систем є реалізація в їх архітектурі популяційних алгоритмів, особливо в частині взаємодіїі з приманками і пастками, а також функціонуванні самих систем. Застосування в архітектурі обманних систем популяційних алгоритмів дає змогу створити динамічне, адаптивне середовище, яке здатне ефективно виявляти нові, невідомі або модифіковані загрози. Основна ідея полягає в тому, що такі алгоритми імітують принципи природного добору, використовуючи популяцію рішень, яка еволюціонує, поступово знаходячи оптимальні або наближені до оптимальних конфігурації системи. У контексті обманних систем для протидії та виявлення ЗПЗ і КА популяційні алгоритми, як клас алгоритмів, можуть бути імплементовані на декількох рівнях архітектури. Це може забезпечити підтримку самоадаптивних можливостей системи. Вони можуть постійно запускатись у фоновому режимі для пошуку нових моделей класифікації або адаптації вже існуючих до змін у мережній поведінці. Такий підхід забезпечує стійкість системи до еволюції атак і дозволяє уникати ситуацій, коли обманні системи втрачають актуальність через статичну модель.

В архітектурі, яка побудована на основі мультиагентних технологій, популяційні алгоритми можуть бути локалізованими в окремих агентах, що дозволяє кожному з них адаптувати власну поведінку, а результати еволюції можуть передаватися іншим агентам у рамках спільного навчання. Це створює умови для децентралізованої, гнучкої та масштабованої обманної системи, яка здатна функціонувати у складних динамічних мережних середовищах.

З точки зору архітектури, особливу цінність становить інтеграція популяційних алгоритмів в середовища, які побудовані із застосуванням мультиагентних технологій. У такій архітектурі кожен агент є окремою автономною одиницею, яка здатна виконувати локальну оптимізацію своєї поведінки залежно від отриманої інформації про стан середовища. Розміщення такого модуля безпосередньо всередині агента забезпечує локальну адаптивність, коли кожен агент може змінювати власні правила виявлення або стратегії взаємодії з іншими елементами обманної системи.

Це важливо у великих або розподілених мережах, де централізоване оновлення моделей є технічно складним або ресурсоємним. Механізм кооперативного навчання між агентами дозволяє ефективно передавати нові сформовані знання від одного агента до інших, сприяючи формуванню узагальненої моделі загроз на основі локальних спостережень. Такий підхід забезпечує децентралізацію, зменшує залежність від центрального контролера та підвищує загальну стійкість системи до відмов чи цілеспрямованих атак на окремі вузли.

Представимо обманну систему з приманками і пастками в корпоративних мережах для виявлення КА та ЗПЗ в частині саме приманок і пасток так:

$$S = \bigcup_{i=1}^{n} S_i\,, \tag{2.6}$$

де n – це кількість приманок і пасток в системі; $n \geq 2$, S_i - i-та приманка чи пастка.

Кількість приманок і пасток в обманній системі з приманками і пастками повинно бути не менше двох, оскільки при невідповідності цій умові система вироджується в одну приманку в комп'ютерній мережі і не зможе виконувати поставлені завдання у решті вузлів в мережі, крім комп'ютерної станції в якій вона розміщена, а також для керування приманками і пастками завдання зводиться до керування одним компонентом обманної системи. Позначення i-тої приманки S_i означає наявність в її складі комп'ютерної станції та програмного забезпечення приманки та/або пастки і системи керування. Тобто кожна приманка чи пастка системи міститься в окремому вузлі в мережі. При цьому допускатимемо, що приманка чи пастка може не мати активного спеціалізованого функціоналу, або активованих засобів, тобто бути нуль-приманкою за спеціалізованим функціоналом. Але вона обов'язково міститиме функційну частину, яка поєднуватиме її з системою або матиме додатково функціонал з керування системою. Особливості функційних можливостей приманки залежать від відповідного наповнення і можуть відрізнятись від решти приманок в системі.

Окремо сформуємо та задамо безпосередньо обманну систему без компонентів, які означені за формулою (2.6) так:

$$S_{o,osn} = \bigcup_{i=1}^{k} S_{o,osn_i}, \tag{2.7}$$

де k – це кількість компонентів та модульних елементів основної частини обманної системи; $k \geq 1$, S_{o,osn_i} - i-тий компонент або модульний елемент; $i = 1,2, \dots, k$.

Тоді, ОСПП задамо згідно формул (2.6) і (2.7) так:

$$S_o = S_{o,osn} \cup S = \bigcup_{j=1}^{k} S_{o,osn_j} \cup \bigcup_{i=1}^{n} S_i, \quad (2.8)$$

де $S_{o,osn}$ – основна частина обманних систем без приманок і пасток; S – частина обманних систем саме з приманками і пастками; n – це кількість приманок і пасток в системі; $n \geq 2$, S_i - i-та приманка чи пастка; k – це кількість компонентів та модульних елементів основної частини обманної системи; $k \geq 1$, S_{o,osn_j} - j-тий компонент або модульний елемент; $j = 1,2, \dots, k$.

Загальна архітектура ОСПП згідно з формулами (2.7) та (2.8) зображена на рис. 2.4.

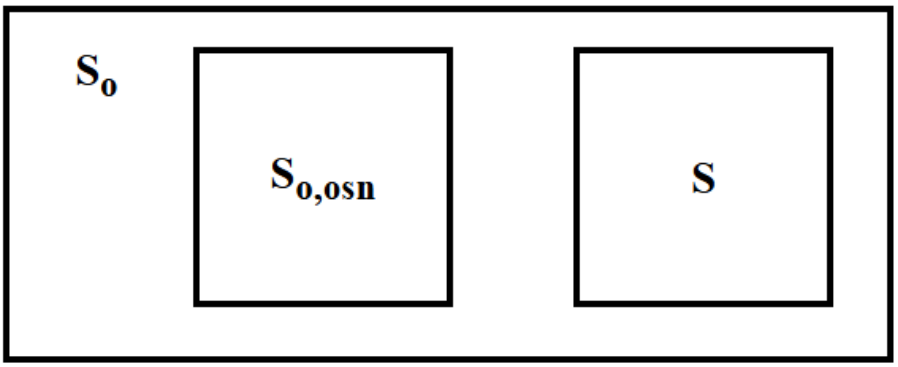


Рисунок 2.4 - Загальна архітектура ОСПП

Обманні системи S_o повинні мати не менше одного компоненту в основній частині, тобто $k \geq 1$ в формулі (2.8), і не менше двох приманок та/або пасток, тобто $n \geq 2$ в формулі (2.8). Таким чином, розглядатимемо обманні системи з мінімум одним компонентом основної частини та мінімум двома приманками чи пастками.

Місце обманної системи в загальній архітектурі корпоративних мереж зображено на рис. 2.5. В ній виділено чотири типи вузлів в залежності від їх функційного призначення та ступеня залучення компонентів і елементів обманних систем:

1) комп'ютерні станції $KC_{1,0} - KC_{n_0,0}$ (n_0 – кількість комп'ютерних станцій), в які встановлені компоненти обманної системи S_o тільки для здійснення керування ними в критичних ситуаціях;

2) комп'ютерні станції $KC_{1,1} - KC_{n_1,1}$ (n_1 – кількість комп'ютерних станцій), в які встановлені компоненти підсистеми S обманної системи S_o і вони виконують функції паралельно із завданнями користувачів;

3) комп'ютерні станції $KC_{1,2} - KC_{n_2,2}$ (n_2 – кількість комп'ютерних станцій з приманок і пасток) з приманками і пастками, в які встановлені компоненти підсистеми S обманної системи S_o і вони виконують винятково функції приманок і пасток, зокрема і імітують роботу серверів в корпоративній мережі;

4) сервери $C_1 - C_{n_3}$ (n_3 – кількість серверів, в які встановлені компоненти обманної системи S_o тільки для здійснення керування ними в критичних ситуаціях.

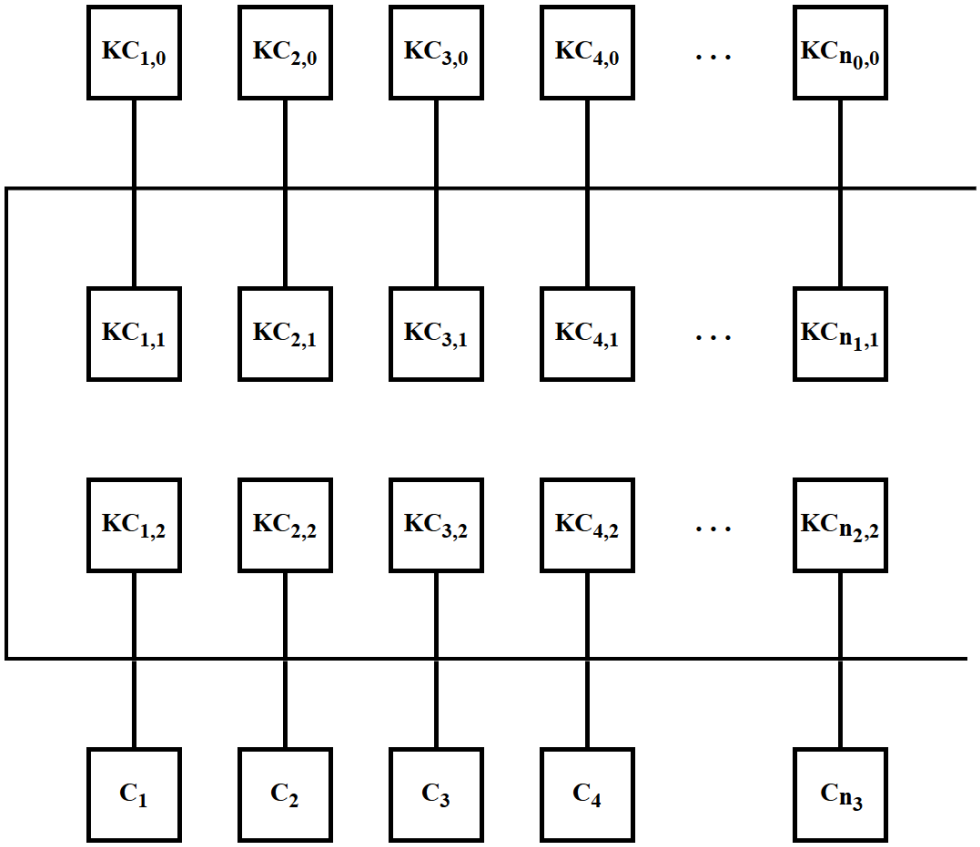


Рисунок 2.5 – Місце ОСПП в загальній архітектурі корпоративних мереж

Таким чином, зловмисник згідно моделі двоцільової атаки повинен класифікувати вузли комп'ютерних мереж для формування чотирьох класів об'єктів в них.

Центр прийняття рішень, який належить основній частині обманних систем $S_{o,osn}$ (формула (2.7)), в системах буде знаходитись в одній з компонент системи і може бути переміщений системою в залежності від поточного стану функціонування

та вибору подальших кроків в функціонуванні всієї системи. ОСПП може мати децентралізовану архітектуру, але тоді зростатимуть витрати часу на обробку подій в ній за рахунок зростання кількості встановлених зв'язків в ній. Тому, розглядатимемо систему з центром, який може переміщуватись системою між різними комп'ютерними станціями в мережі. Для його захисту використовуватимемо як можливість його переміщення так і попередню обробку мережних пакетів в інших вузлах мережі, які надходитимуть до вузла, в якому перебуватиме центр системи.

Синтез ОСПП в корпоративних комп'ютерних мережах для виявлення КА та ЗПЗ, в якому буде поєднання системи, мультиагентних технологій та приманок і пасток для виявлення КА і ЗПЗ дозволить отримати переваги над КА і ЗПЗ за рахунок залучення до виявлення та протидії більшої кількості різнотипних обчислювальних засобів та ресурсів порівняно з можливостями зловмисних проявів.

Здійснення керування обманною системою з приманками і пастками в корпоративних комп'ютерних мережах і її компонентами забезпечуватиметься засобами з використанням компонентів штучного інтелекту, бо потребуватиме оперативного вирішення використання приманок і пасток в поточний момент часу, в якому вузлі міститиметься центр системи і куди його за потреби перемістити, як перебудовувати динамічно систему і які подальші дії вона повинна виконувати. Ці завдання включатимуть необхідність перевірки значної кількості параметрів як в системі так і ззовні неї, що потребуватиме оперативної обробки таких параметрів та подій, які впливатимуть на зміну параметрів ззовні.

Необхідність залучення в обманну систему з приманками і пастками в корпоративних комп'ютерних мережах мультиагентних технологій та реалізації створення та керування колективом агентів пов'язано з особливостями КА, частина з яких орієнтована на суттєве перевантаження обчислювальних ресурсів, що потребуватиме розподілу засобів виявлення системи між зловмисними процесами в мережі, а також для виявлення ЗПЗ, яке може перебувати в різних вузлах в мережі та в різних місцях у певному вузлі. Тому, наявність агентів, які кооперуватимуться для вирішення задач з виявлення, покращить ефективність та достовірність результату з виявлення.

Реалізація в обманних системах з приманками і пастками в корпоративних комп'ютерних мережах приманок і пасток різного типу та призначення дасть змогу

врахувати використання зловмисниками різноманітних технологій здійснення КА та ЗПЗ. Різні функційні можливості в приманках і пастках можуть бути представлені повністю або частково в певних з них і система визначатиме, які з них будуть залучатись в процес виявлення.

ОСПП для заплутування зловмисника на протязі тривалого часу функціонування повинні мати можливості для перебудови своєї архітектури в динамічному режимі без втручання користувача, тобто в архітектурі систем мають бути закладені механізми забезпечення адаптивності та гнучкості.

Після певного часу функціонування системи з приманками і пастками потребуватимуть визначення своїх подальших кроків. Вони можуть продовжувати виконувати поточні завдання щодо моніторингу процесів в комп'ютерних системах і мережах. Але при виявленні підозрілих подій, або після їх опрацювання, або після динамічної зміни їх архітектури їм потрібно буде визначитись щодо своїх подальших кроків. Для виконання цього завдання вони не повинні очікувати команд від адміністратора чи користувача, бо це сповільнить їх роботу і може призвести, також, до втрати зловмисних проявів, які відбуватимуться протягом часу очікування. Тому такі системи повинні мати можливість здійснювати самоорганізацію, тобто бути самоорганізованими. Побудова механізмів самоорганізації повинна базуватись не на вичерпному переліку станів, в які вони могли б переходити в залежності від опрацьованих подій протягом певного часу, а мати перелік напрямків вибору завдань, які їм потрібно виконати. Щодо детальніших кроків та залучення необхідних засобів для виконання таких завдань, то це вже має вибудовуватись системою самостійно, виходячи з результатів оцінювання наявних засобів для досягнення мети. ОСПП повинні самостійно оцінити можливості виконання завдань з визначеного напрямку свого подальшого спрямування і за потреби, наприклад неможливості виконати рух у визначеному напрямку, який було визначено ними, перебудувати себе та визначити інший напрям для досягнення виконання завдань.

В обманних системах з приманками і пастками повинно бути декілька рівнів, на яких реалізується підтримка прийняття рішень. Це може бути реалізовано в складі одного центру, або може бути в одному центрі і для виконання інших завдань можуть бути використанні компоненти системи, в яких теж знаходитимуться ці підсистеми

підтримки прийняття рішень. Ефективність результату виявлення КА і ЗПЗ суттєво залежатиме від архітектури підсистеми підтримки прийняття рішень.

До засобів ОСПП для виявлення КА і ЗПЗ необхідним є залучення агентів та організація їх колективної роботи щодо поставлених завдань із виявлення. Наявність таких агентів збільшуватиме можливості системи за рахунок створення цих агентів в динамічному процесі в залежності від поточних потреба та завдань.

На відміну від засобів виявлення КА і ЗПЗ, які орієнтовані на виявлення та негайну реакцію на результат виявлення, засоби побудовані з використанням приманок і пасток орієнтовані саме на продовження подій та дозвіл на проникнення в розставлені системою приманки і пастки для забезпечення захисту важливих ресурсів користувача та отримання максимальної інформації про зловмисні прояви. Це впливатиме на їх внутрішню архітектуру.

В результаті S_i – приманка чи пастка ($i = 1, \dots, n$, n – кількість приманок чи пасток) міститиме такі складові елементи, які задамо вектором:

$$S_i = \left(s_{i,0}, s_{i,1}, s_{i,2}, \dots, s_{i,k}\right), \tag{2.9}$$

де $s_{i,0}$ – компонента вектора S_i, в якій міститься загальний функціонал компоненти, що відповідає за її функціонування, збір нових даних, комунікацію та функціонування центру системи за потреби; $s_{i,j}$ - j-та компонента вектору S_i, яка показує наявність складової частини компоненти приманки S_i, k – кількість спеціалізованих засобів приманки для ЗПЗ та КА, $j = 1, \dots, k$.

Компоненти вектору S_i ($i = 1, \dots, n$, n – кількість приманок та пасток), які відповідають за складові приманок чи пасток присутні завжди, навіть якщо безпосередні складові компоненти відсутні. В цьому випадку така складова компоненти прирівнюється до нуля: $s_{i,j} = 0$ ($j = 1, \dots, k$).

Таким чином, в обманних системах з приманками та пастками для заплутування зловмисників з метою виявлення КА та ЗПЗ необхідно синтезувати засоби обманних систем, мультиагентних технологій та приманок і пасток. Архітектура таких синтезованих обманних систем повинна надавати можливість будувати засоби, які будуть адаптивними, гнучкими, самоорганізованими, прийматимуть рішення щодо своїх подальших кроків та щодо виявлення, а також організовуватимуть колективну

робота агентів. Така архітектура ОСПП в поєднанні з відомостями в ній про наявні ресурси комп'ютерних мереж дасть перевагу над засобами зловмисників.

2.3 Синтез обманних систем з приманками та пастками виявлення ЗПЗ та КА в корпоративних мережах

Обманні системи з приманками та пастками для забезпечення переваг над засобами зловмисників повинні мати архітектуру, яка надаватиме їм можливості бути адаптивними, гнучкими, самоорганізованими. Для забезпечення таких функційних можливостей застосуємо поділ системи на компоненти і елементи її таким чином, щоб вона досягла певного рівня модульності в залежності від рівнів функціонування та завдань відповідних рівнів. Це важливо для відокремлення функцій з самостійного керування роботою всієї системи і виконання локальних завдань та безпосередньо з реалізації завдань спеціалізованого призначення для організації приманок та пасток і обробки результатів їх роботи. Таким чином, необхідним є введення в модель архітектури системи рівнів та модульності. Зображення архітектури підсистеми S (формули (2.6), (2.9)) системи S_o з врахуванням розглянутих вимог задано на рис. 2.6.

Позначення елементів, які зображено на рис. 2.6, відповідає змінним в формулах (2.6) та (2.9).

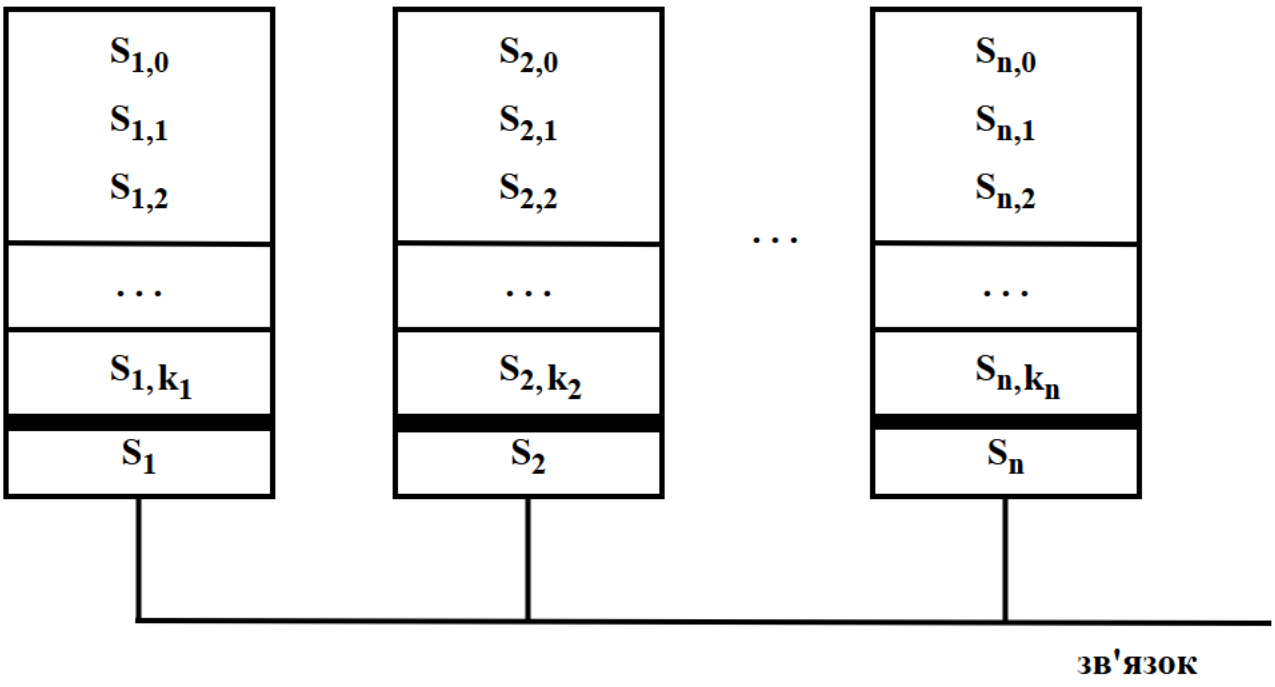


Рисунок 2.6 - Узагальнена архітектура підсистеми S

Обов'язковими підсистемами елементу S_i $(i = 1, \dots, n$, n – кількість приманок і пасток) множини S є підсистема керування компонентою S_i $(i = 1, \dots, n)$, засоби комунікації в системі S, підсистема сенсорів для отримання інформації про зовнішні по відношенню до компоненти події, процеси і стани в комп'ютерній станції, в яку вона встановлена, включаючи також інформацію щодо зовнішніх та внутрішніх комунікацій в середовищі, де експлуатується вся система S, підсистема керування всією системою S за потреби. Таким чином, в узагальненій архітектурі ОСПП виділено рівні, на яких розміщено відповідні функційні засоби систем та, відповідно, досягнуто певної модульності в компонентах системи. Така внутрішня архітектурна організація системи дасть змогу ефективніше змінювати її конфігурацію протягом часу експлуатації.

Розглянемо детальніше організацію функціонування підсистеми керування обманними системами з приманками і пастками з врахуванням сформульованих до них вимог бути адаптивними, гнучкими, самоорганізованими. Підсистема керування обманними системами з приманками і пастками може бути активною в одній з компонент системи, а в решті компонент або неактивною або відсутньою. Визначення наступної компоненти, в якій буде функціонувати підсистема керування системою S, здійснюватиметься в компоненті, в якій знаходиться активна в поточний стан система компонента, або більшість компонент системи визначають наступну таку компоненту при втраті зв'язку з попередньою компонентою, в якій функціонував центр системи і був активним та основним. Крім того, така підсистема повинна здійснювати опрацювання подій, що надходять ззовні системи S від сенсорів, від комп'ютерної станції, в яку вона встановлена як компонент системи S, приймати рішення щодо використання приманок та пасток компоненти і системи в цілому, а також здійснювати планування та визначення своїх подальших кроків без втручання користувача чи адміністратора системи, що є актуальним враховуючи специфіку завдань системи S. Поведінка системи повинна бути незрозумілою як адміністратору системи, так і зловмисникам чи ЗПЗ, інакше результативність її роботи суттєво знизиться. Тобто, вона не повинна бути такою, що на певні початкові набори вхідних даних реагує постійно однаково і передбачувано. Крім того, її подальші дії повинні визначатись нестандартно і бути непрогнозованими. Все це суттєво ускладнює її

внутрішню будову підсистеми керування. Таким чином, підсистема керування має враховувати такі вимоги:

1) керувати опрацюванням подій, які надходять ззовні системи;

2) здійснювати комунікацію між розподіленими компонентами системи;

3) підтримувати цілісність системи та змінювати місце розміщення центру системи;

4) розподіляти завдання між підсистемами керування компонентами;

5) здійснювати аналіз результатів роботи приманок і пасток та залучати різні компоненти системи для збільшення та нарощування обчислювальних ресурсів і можливостей з виконання спеціалізованих завдань, які забезпечуються відповідними приманками;

6) визначати подальші кроки у функціонуванні системи;

7) здійснювати аналіз попередніх рішень і згідно його результатів приймати рішення;

8) володіти засобами для визначення нестандартних подальших дій;

9) оцінювати прийняті рішення з керування системою;

10) гнучко перебудовувати свою архітектуру;

11) здійснювати самонавчання протягом всього часу експлуатації системи;

12) приймати оперативні рішення в нестандартних ситуаціях, обираючи оптимальне рішення з можливих варіантів.

Всі ці вимоги відносяться винятково до організації функціонування підсистеми керування системою S і не відносяться до решти підсистем. Наповнення функційних можливостей решти підсистем здійснюватимемо з врахуванням їх завдань та команд отриманих від підсистеми керування. Узагальнена схема з місцем центру керування системою, компонентами центру та відображенням комунікації із зовнішнім середовищем зображено на рис. 2.7. Узагальнену структуру керування підсистемою S системою S_o з врахуванням функціональних елементів зображено на рис. 2.8.

Кожна компонента системи містить підсистему керування, тому зображені на рис. 2.8 функційні елементи керування підсистемою S системи S_o відносяться до всіх компонентів системи $S_1, S_2, S_3, \ldots, S_n$, які формують підсистему S. Функційні елементи $s_{i,0,1} - s_{i,0,12}$ $(i = 1,2, \ldots, n)$ відносяться до підсистеми керування системою і не відносяться підсистеми керування компонентою. Для керування компонентою

системи S використовуватимуться інші функційні елементи з цієї підмножини для компоненти $s_{i,0}$ $(i = 1,2,\dots,n)$. Виділення функційних елементів до такого рівня деталізації дасть змогу досягти модульності, розподілу за рівнями пріоритетності при виконанні завдань і буде використано при визначенні наступного місця розміщення центру керування за рахунок встановлення активності відповідного елементу. Забезпечення інформації щодо активності / неактивності функційних елементів $s_{i,0,1} - s_{i,0,12}$ $(i = 1,2,\dots,n)$ компонент здійснюватимемо формуванням вектору функційних елементів підсистеми керування компонент системи S з компонентами, які відповідатимуть предикативним функціям. Сформуємо такий вектор, позначивши його $v_{s_{i,0}}$ $(i = 1,2,\dots,n)$, так:

$$v_{s_{i,0}} = \left(p(s_{i,0,1}), p(s_{i,0,2}), \dots, p(s_{i,0,12})\right), \tag{2.10}$$

де $p(s_{i,0,q})$ – предикат, що вказує на активність / неактивність функційного елементу видаючи значення $\{1\}$ або $\{0\}$ відповідно; $i = 1,2,\dots,n$, n – кількість компонент системи S; $q = 1,2,\dots,12$.

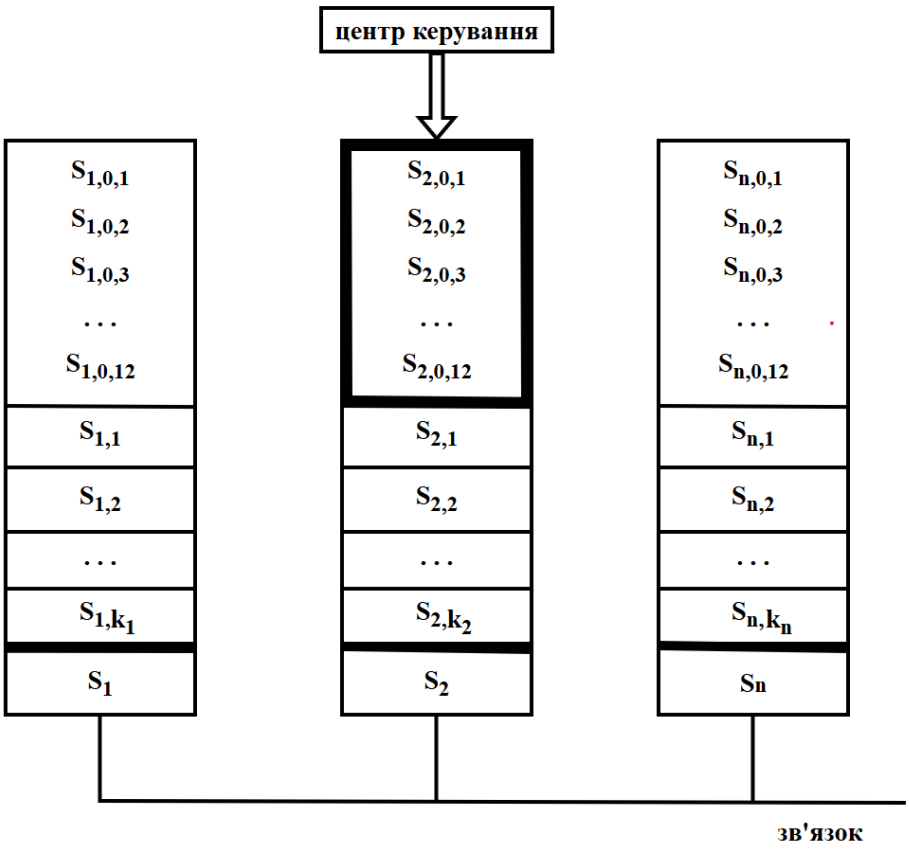


Рисунок 2.7 - Узагальнена структура з відображенням місця керування підсистемою S та комунікації із зовнішнім середовищем

функційний елемент підсистеми керування для обробки подій, які надходять ззовні системи	$s_{i,0,1}$	компоненти підсистеми: $S_1, S_2, S_3, \ldots, S_n$
функційний елемент підсистеми керування для здійснення комунікації між розподіленими компонентами системи	$s_{i,0,2}$	
функційний елемент підсистеми керування для підтримки цілісності системи та зміни місця розміщення центру системи	$s_{i,0,3}$	
функційний елемент підсистеми керування розподілом завдання між підсистемами керування компонентами	$s_{i,0,4}$	
функційний елемент підсистеми керування для здійснювати аналізу результатів роботи приманок та залучення різних компонентів системи для збільшення та нарощування обчислювальних ресурсів і можливостей з виконання спеціалізованих завдань, які забезпечуються відповідними приманками	$s_{i,0,5}$	
функційний елемент підсистеми керування для визначення подальших кроків у функціонуванні системи	$s_{i,0,6}$	
функційний елемент підсистеми керування для здійснення аналізу попередніх рішень і згідно його результатів прийняття рішення	$s_{i,0,7}$	
функційний елемент підсистеми керування для визначення нестандартних подальших дій	$s_{i,0,8}$	
функційний елемент підсистеми керування оцінювання прийнятих рішень щодо керування системою	$s_{i,0,9}$	
функційний елемент підсистеми керування гнучкою перебудовою архітектури системи	$s_{i,0,10}$	
функційний елемент підсистеми керування для здійснення самонавчання протягом всього часу експлуатації системи	$s_{i,0,11}$	
функційний елемент підсистеми керування для прийняття оперативних рішень в позаштатних ситуаціях	$s_{i,0,12}$	

Рисунок 2.8 - Узагальнена структура керування підсистемою S з врахуванням функціональних елементів

Таким чином, сформована в поточний момент часу така інформація щодо активності / неактивності компонентів буде використовуватись для прийняття рішення системою щодо подальших кроків. Крім того, подальша деталізація керування підсистеми S буде враховувати компоненти сформованого згідно формули (2.10) вектору. Для кожної компоненти системи сформовані вектори поєднаємо у вектор $V_{s_{0,1}}$ за формулою так:

$$V_{s_{0,1}} = \left(v_{s_{1,0}}, v_{s_{2,0}}, \ldots, v_{s_{n,0}}\right), \tag{2.11}$$

де n – кількість компонент системи S.

В антивірусних засобах, які використовують приманки і пастки, крім підсистеми визначення зловмисної активності в комп'ютерних системах і мережах, важливою складовою є підсистема керування всією мережею приманок і пасток із залученням різноманітних засобів. Відомостей про підсистему прийняття рішень відомих мереж приманок і пасток недостатньо для їх використання. Крім того, рівень непередбачуваності дій антивірусних засобів при використанні відомих або готових рішень саме в підсистемі керування усією системою повинен бути належним та непрогнозованим. Тому, цей рівень, на якому здійснюється керування всією системою є важливим і потребує розроблення.

Позначимо зовнішні впливи на компоненти системи і систему S_o в цілому елементами множини W. Позначимо ці впливи узагальнено, без деталізації особливостей та їх класифікації. Наявність впливів в загальній моделі функціонуючих процесів в середовищі розміщення і експлуатації підсистеми S є необхідним, бо такі впливи спонукатимуть систему до реакції на них і відповідно формуватимуть множину наслідків. Врахування впливів, що надходять із зовнішнього середовища, може бути пов'язане як із завданнями для встановлення дій ЗПЗ чи КА, так і дій, які не відносяться до зловмисної активності. Але такі впливи із зовнішнього середовища спонукатимуть підсистему S до їх аналізу та обробки. Крім того, впливи можуть бути викликані і подіями, що відбуватимуться безпосередньо в компонентах самої системи S_o, тобто ці впливи будуть внутрішніми і викликаними в результаті функціонування безпосередньо самої системи S_o. До опрацювання подій, що виникатимуть через внутрішні впливи на компоненти системи S_o, не залучатимуться засоби виявлення ЗПЗ та КА. Але такі впливи можуть відбуватись синхронно із зовнішніми впливами і оскільки система S_o має розподілену архітектуру, то вони повинні враховуватись, бо оброблятимуться в компонентах системи. Також, ці внутрішні впливи можуть бути результатами обробки зовнішніх впливів, оскільки система S_o орієнтована на організацію функціонування мережі приманок для ЗПЗ та КА. Тому, враховуючи складність та природу впливів, яка пов'язана із зовнішніми та внутрішніми джерелами їх появи, розглядатимемо їх всі як елементи однієї множини $W = \{w_1, w_2, \dots, w_{k_S}\}$, де k_S – кількість впливів. До елементів множини W відноситимемо тільки ті впливи, які сприймає система S_o і реагує на них. Можуть бути впливи, на які

система S_o не реагує і на які не має засобів сприймати їх. Вважатимемо, що системо S_o матиме засоби для сприйняття максимально можливої кількості впливів. Якщо ж протягом тривалого часу функціонування системи S_o вона спочатку сприймала і обробляла певні конкретні впливи, а в подальшому ні, то такі впливи вважатимемо елементами множини W. Втрата можливості сприйняття частини впливів може бути пов'язана з деградацією системи S_o і підсистеми керування нею повинні обробляти такі події. Також, в середовищі функціонування системи S_o можуть відбуватись процеси, впливи яких не сприйматимуться її засобами, але через певний час її функціонування частина таких впливів буде видима для засобів системи S_o і вони будуть оброблятись нею, тоді їх буде додано до множини W системою S_o.

Елементи множини W можуть бути одночасно ідентифіковані різними компонентами системи S_o. Тому, впливи на систему S_o можуть бути одні і ті ж, але їх спрямування може бути в різних комп'ютерних системах в компонентах системи. В зв'язку з можливістю таких подій, як не одиничних, необхідно враховувати їх при проєктуванні системи S_o. Джерела таких впливів можуть бути різними, а може бути одне. Пошук джерел впливів в комп'ютерних системах в мережі, в які встановлені компоненти системи S_o теж є її завданням, тому аналіз таких впливів і їх джерел є необхідними. Задамо потенційні впливи на компоненти системи S_o з множини W вектором впливів на компоненту так:

$$w_{S_i} = \left(w_{1,S_i}, w_{2,S_i}, \dots, w_{k_S,S_i}\right), \tag{2.12}$$

де w_{l,S_i} – активність / неактивність l-го впливу на S_i компоненту системи, який кодується значеннями {1} або {0} відповідно; $i = 1,2, \dots, n$, n – кількість компонент системи S; $l = 1, 2, \dots, k_S$, k_S – кількість можливих впливів.

Тоді, вектор впливів на всю систему S_o в частині приманок і пасток задаватимемо так:

$$W_S = \left(w_{S_1}, w_{S_2}, \dots, w_{S_n}\right), \tag{2.13}$$

де w_{S_i} – вектор впливів на S_i компоненту системи S; $i = 1,2, \dots, n$; n – кількість компонент підсистеми S.

Згідно формул (2.12) та (2.13) таке задання двох типів векторів W_S та w_{S_i} ($i = 1,2, \dots, n$) дає змогу отримати їх прямий добуток, який формуватиме матрицю впливів

на систему S_o в цілому з деталізацією впливів на кожну компоненту. Задамо її формулою так:

$$M_{W_S} = W_S \times w_{S_{i(i=1,2,\dots,n)}}, \tag{2.14}$$

де w_{S_i} – вектор впливів на S_i компоненту системи (формула 2.5); $i = 1,2,\dots,n$, n – кількість компонент підсистеми S.

Аналогічно за формулами (2.10) і (2.11) отримуємо матрицю $M_{V_{S_{0,1}}}$ впливів на функційні елементи компонентів підсистеми S, які відносяться до підсистеми керування системою S_o. Задамо її так:

$$M_{V_{S_{0,1}}} = \begin{pmatrix} p(s_{1,0,1}) & \cdots & p(s_{1,0,n}) \\ \vdots & \ddots & \vdots \\ p(s_{12,0,1}) & \cdots & p(s_{12,0,n}) \end{pmatrix}, \tag{2.15}$$

де n – кількість компонент підсистеми S.

Таким чином, впливи на систему можуть бути ззовні неї та із середини. Але в сформованій множині впливів W містяться впливи, які можуть бути вторинними по відношенню до початкових впливів, тобто впливи які виникають після впливу на систему S_o. Також, цими вторинними впливами можуть бути і початкові впливи, тобто впливи початкові і вторинні можуть збігатись. Важливими для розгляду є руйнуючі та неруйнуючі впливи, які можуть впливати на систему, зокрема і цілеспрямовано для її деградації. Крім того, впливи можуть зловмисними або такими, що породжені виконанням завдань користувача чи системи S_o. Подальша деталізація компонентів та елементів системи, її архітектури потребуватиме виокремлення підмножин впливів з певними особливостями і їх врахування для забезпечення стійкості системи в цілому та її компонентів.

Впливи в комп’ютерних системах можуть відбуватись без подальших збурень системи S_o, тобто можуть відбуватись невидимо для давачів системи або відбуватись так, що будуть невидимими для неї. Аде більшість впливів буде видима для системи. І завданням з розробки архітектури системи S_o є забезпечення її такими засобами, щоб вона могла бачити максимальну кількість подій в комп’ютерній системі та мережі.

Результатами дій впливів на систему S_o будуть наслідки. Вони можуть стати новими впливами або сформують в системі певний ланцюг дій. Впливи

змінюватимуть стани системи. Наслідки впливів можуть активувати засоби системи S_o з активізації надання приманок чи пасток в комп'ютерній мережі або будуть мати відношення до дій системи для обробки подій. Але наслідки можуть породжувати нові впливи, які переважно впливатимуть на гнучку зміну та динамічну перебудову архітектуру системи S_o. Наслідки впливів можуть поділені на критичні та некритичні. Наслідки для системи S, також, можуть виникнути не в результаті відбування в комп'ютерній системі чи мережі певних процесів, які формуватимуть впливи, але і в результаті певного часу функціонування системи S_o та будуть результатом виконання певних дій в системі. Тобто, з'являтимуться в результаті функціонування системи S_o будуть для системи S_o вторинними. Враховуючи важливість наслідків і їх різноманітність для функціонування системи S_o введемо для них множину $M_{S,N}$, яку задамо її елементами так:

$$M_{S,N} = \left\{m_{S,n_1}, m_{S,n_2}, \dots, m_{S,n_{k_N}}\right\}, \tag{2.16}$$

де k_N – кількість наслідків; m_{S,n_l} – елемент множини, який кодує l – ий наслідок; $l = 1,2, \dots, k_N$.

Якщо врахувати, що наслідки впливів виникають в компонентах системи S_o в комп'ютерних системах, тоді сформуємо матрицю наслідків з врахуванням не системи в цілому, а саме її компонентів. Прояви наслідків задамо предикатами та окремо ймовірнісними функціями. Задання функціями із значеннями з проміжку $[0; 1]$ необхідно для використання в системі S_o для оціночного обчислення на певних кроках її функціонування. Якщо ж використовувати предикативне значення, то воно потрібно для встановлення факту наслідку і не завжди може бути чітким. Задамо матрицю наслідків з врахуванням компонентів системи згідно їх предикативних значень так:

$$M_{S,N,p} = \begin{pmatrix} p\left(m_{S,n_1,1,1}\right) & \cdots & p\left(m_{S,n_1,n,1}\right) \\ \vdots & \ddots & \vdots \\ p\left(m_{S,n_1,1,k_N}\right) & \cdots & p\left(m_{S,n_1,n,k_N}\right) \end{pmatrix}, \tag{2.17}$$

де n – кількість компонент підсистеми S; k_N – кількість наслідків.

Матрицю наслідків з врахуванням компонентів системи згідно їх ймовірнісних значень з проміжку $[0; 1]$ задамо так:

$$M_{S,N,\bar{\bar{f}}} = \begin{pmatrix} \bar{\bar{f}}(m_{S,n_1,1,1}) & \cdots & \bar{\bar{f}}(m_{S,n_1,n,1}) \\ \vdots & \ddots & \vdots \\ \bar{\bar{f}}(m_{S,n_1,1,k_N}) & \cdots & \bar{\bar{f}}(m_{S,n_1,n,k_N}) \end{pmatrix}, \quad (2.18)$$

де n – кількість компонент підсистеми S; k_N – кількість наслідків; $\bar{\bar{f}}$ – функція, яка обчислює ймовірність прояву наслідку і її значення знаходиться в проміжку $[0; 1]$.

Задані подання впливів та їх наслідків необхідні системі S_o для прийняття рішення щодо подальших дій. Без впливів із зовнішнього середовища система S_o буде перебувати в поточному стані і за появи внутрішніх впливів оброблятиме їх. Система S_o під впливами та наслідками, які можуть породжувати знову впливи, буде змінювати стани, в яких вона перебуватиме у вузлах комп'ютерної мережі. Ці зміни можуть відбуватись синхронно в декількох компонентах одночасно або асинхронно в різних компонентах. При цьому впливи можуть збігатись в різних компонентах. Система S_o в своїх компонентах має різні стани. Її загальний стан аналізується підсистемою керування для визначення подальших кроків. До таких станів компонент системи S_o віднесемо активність / не активність функційних елементів підсистем компонент.

Нехай множина елементів підсистеми S задана множиною елементів так:

$$M_{S_i,p_e} = \{s_{i,0}, s_{i,1}, \dots, s_{i,N_S-1}\}, \quad (2.19)$$

де N_S – кількість елементів підсистем в компонентах підсистеми S; $i = 1,2, \dots, n$; n – кількість компонентів в підсистемі S.

Кількість функційних елементів кожної з підсистем компоненти всієї підсистеми S є не однаковою. Тому, для кожної з підсистем компоненти згідно формули (2.19) задамо вектори, компонентами яких будуть значення предикатів, які визначатимуть активність / не активність функційних елементів і позначатимуться відповідно значеннями $\{1\}$ або $\{0\}$. Тоді, з врахуванням функційних елементів в підсистемах компонентів вектори їх станів в компонентах задамо так:

$$s_{i,j} = \left(p(s_{i,j,1}), p(s_{i,j,2}), \dots, p(s_{i,j,N_{s_{i,j}}})\right), \quad (2.20)$$

де $s_{i,j}$ - j-а підсистема i-ої компоненти; $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l = 1,2, \dots, N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-

ої компоненти; $i = 1,2,\dots,n$; n – кількість компонент системи S; $J = 0,1,2,\dots,N_{s_i} - 1$; N_{s_i} – кількість підсистем в компоненті S_i.

Кількість функційних елементів підсистем компонентів різна, тому вектори матимуть різну кількість своїх компонентів і не можуть бути зведеними в єдину матрицю. Значення для векторів $s_{i,j}$ можуть бути обчислені двома способами. Перший спосіб полягає в тому, що якщо активний хоча б один функційний елемент, тоді значення вектору $p(s_{i,j}) = 1$, інакше $p(s_{i,j}) = 0$ при $i = 1,2,\dots,n$, n – кількість компонент системи S, $J = 0,1,2,\dots,N_{s_i} - 1$, N_{s_i} – кількість підсистем в компоненті S_i. Тобто, отримуємо предикативне значення. Але може бути потрібно отримувати значення порівнювані із значеннями решти підсистем, тоді застосуємо другий спосіб, який базується на визначенні кількості активних функційних елементів щодо усіх елементів і обчислюється так:

$$\bar{g}(s_{i,j}) = \frac{\sum_{l=1}^{N_{s_{i,j}}} p(s_{i,j,l})}{N_{s_{i,j}}}, \tag{2.21}$$

де $p(s_{i,j,l})$ – значення предикату, яке вказує на активність / не активність функційного елементу і визначається числами {1} або {0} відповідно; $s_{i,j}$ - j-а підсистема i-ої компоненти; $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l = 1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти; $i = 1,2,\dots,n$; n – кількість компонент підсистеми S; $J = 0,1,2,\dots,N_{s_i} - 1$; N_{s_i} – кількість підсистем в компоненті S_i.

Отримані значення для векторів $s_{i,j}$ в процесі функціонування системи S_o використовуватимуться як аргументи для визначення її подальших кроків і, тому, потребують унормування для врахування частки впливу. Здійснимо унормування їх предикативних значень так:

$$p_1(s_{i,j,l}) = \frac{p(s_{i,j,l})}{\sum_{l=1}^{N_{s_{i,j}}} p(s_{i,j,l})}, \tag{2.22}$$

де $i = 1,2,\dots,n$, n – кількість компонент підсистеми S, $J = 0,1,2,\dots,N_{s_i} - 1$, $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l = 1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти.

Тоді, справедливе співвідношення: $\sum_{l=1}^{N_{s_{i,j}}} p_1(s_{i,j,l}) = 1$. Так унормовані значення можуть бути аргументами в формулі (2.21):

$$\bar{g}(s_{i,j}) = \frac{\sum_{l=1}^{N_{s_{i,j}}} p_1(s_{i,j,l})}{N_{s_{i,j}}}, \tag{2.23}$$

де $p_1(s_{i,j,l})$ – унормоване значення предикату $p(s_{i,j,l})$; $s_{i,j}$ - j-а підсистема i-ої компоненти; $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l = 1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти; $i = 1,2,\dots,n$; n – кількість компонент підсистеми S; $J = 0,1,2,\dots,N_{s_i} - 1$; N_{s_i} – кількість підсистем в компоненті S_i.

Формули (2.21) – (2.23) визначають значення безпосередньо для компонент підсистеми S. Тобто, всі результати обчислень стосуються окремих компонент. Але важливими, також, є значення для однакових підсистем компонент. Підсистеми компонент містять однакові функційні елементи. Хоча не всі компоненти можуть містити всі підсистеми, особливо це стосується компонент, в яких міститься підсистема керування системою S_o та функційні елементи для роботи певних визначених приманок. Це потрібно для оцінювання функціонування системи S_o в розрізі її підсистем першочергово, а не компонент. Оскільки в переважній більшості компонент містяться однакові підсистеми, то дослідження процесів, які в них відбуваються, доповнить базу аргументів для визначення подальших дій системи S_o. Задамо вектори для кожної з підсистем у всіх компонентах підсистеми S. Якщо відповідна підсистема буде відсутньою у компоненті, тоді у відповідній компоненті вектору її значення встановимо $\{0\}$. Для обчислень в розрізі підсистем враховуватимемо тільки компоненти, в яких наявні підсистеми. Це важливо, бо система може мати таку архітектуру, що значна частина компонент не міститиме певної підсистеми, і обчислення для всіх компонентів спотворюватиме точність результату.

Тому, визначимо значення для підсистем з врахуванням їх задання формулами (2.19) і (2.20) так:

$$p_{2,j}(s_{i,j},j)=\begin{cases}1,\ \text{при всіх } i=1,2,\dots,n \text{ та існує хоча б одна активна активна } j-\text{а підсистема хоча б в одній зі всіх компонент системи } S, n-\text{кількість компонент системи, інакше}\\ 0,\text{при всіх } i=1,2,\dots,n \text{ та не існує жодної активної } j-\text{ї підсистеми хоча б в одній зі всіх компонент системи } S, n-\text{ кількість компонент системи } S,\\ J=0,1,2,\dots,N_{s_i}-1, N_{s_i}-\text{ кількість підсистем в компоненті } S_i.\end{cases}, \quad (2.24)$$

Тобто, отримуємо предикативне значення для підсистеми, яка міститься в більшості компонент або в усіх компонентах системи. Також, потрібні значення підсистем, які обчислюються за кількістю активних функційних елементів щодо усіх елементів так:

$$\overline{g_{2,j}}(s_{i,j},j)=\frac{\sum_{i=1}^{n}\sum_{l=1}^{N_{s_{i,j}}}p(s_{i,j,l})}{(n-k)N_{s_{i,j}}}, \quad (2.25)$$

де $p(s_{i,j,l})$ – значення предикату, яке вказує на активність / не активність функційного елементу і визначається числами $\{1\}$ або $\{0\}$ відповідно; $s_{i,j}$ - j-а підсистема i-ої компоненти; $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l=1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти; $i=1,2,\dots,n$; n – кількість компонент системи S; $J=1,2,\dots,N_{s_i}$; N_{s_i} – кількість підсистем в компоненті S_i; k – кількість компонент в системі, в яких відсутня J - а підсистема.

Отримані значення для підсистем в процесі функціонування підсистеми S використовуватимуться як аргументи для визначення її подальших кроків і, тому, потребують унормування для врахування частки впливу. Здійснимо унормування їх предикативних значень так:

$$p_{3,j}(s_{i,j,l},j)=\frac{\sum_{l=1}^{N_{s_{i,j}}}p(s_{i,j,l})}{\sum_{i=1}^{n}\sum_{l=1}^{N_{s_{i,j}}}p(s_{i,j,l})}, \quad (2.26)$$

де $p(s_{i,j,l})$ – значення предикату, яке вказує на активність / не активність функційного елементу і визначається числами $\{1\}$ або $\{0\}$ відповідно; $s_{i,j}$ – j-а підсистема i-ої компоненти; $s_{i,j,l}$ – функційний елемент j-ї підсистеми i-ої компоненти; $l=1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти; $i=$

$1,2,\dots,n$; n – кількість компонент системи S; $J = 0,1,2,\dots,N_{s_i} - 1$; N_{s_i} – кількість підсистем в компоненті S_i.

Тоді, справедливе співвідношення: $\sum_{l=1}^{N_{s_{i,j}}} p_{3,j}(s_{i,j,l}, j) = 1$.

Так унормовані значення можуть бути аргументами в формулі (2.25):

$$\overline{g_{3,j}}(s_{i,j}, j) = \frac{\sum_{l=1}^{N_{s_{i,j}}} p_{3,j}(s_{i,j,l}, j)}{(n-k)N_{s_{i,j}}}, \quad (2.27)$$

де $p_1(s_{i,j,l})$ – унормоване значення предикату $p(s_{i,j,l})$; $s_{i,j}$ - j-а підсистема i-ої компоненти; $s_{i,j,l}$ - функційний елемент j-ї підсистеми i-ої компоненти; $l = 1,2,\dots,N_{s_{i,j}}$; $N_{s_{i,j}}$ – кількість функційних елементів j-ї підсистеми i-ої компоненти; $i = 1,2,\dots,n$; n – кількість компонент підсистеми S; $J = 0,1,2,\dots,N_{s_i} - 1$; N_{s_i} – кількість підсистем в компоненті S_i; k – кількість компонент в системі, в яких відсутня J - а підсистема.

Формули (2.24) - (2.25) дозволяють обчислювати значення безпосередньо для підсистем компонент системи S_o. Тобто, всі результати обчислень стосуються окремих підсистем компонент.

Отримані таким чином числові характеристики елементів та компонентів системи S_o дозволяють здійснювати її оцінювання поточного стану для визначення її подальших кроків. Вони є частиною загальної сукупності характеристик і основною частиною для керування підсистемою S. Активність та не активність функційних елементів підсистем компонентів підсистеми S можна задати станами, в яких відображатиметься поточний стан системи S. Частина станів системи може мати зв’язки між собою. Зміна станів функційних елементів може відбуватись динамічно. Впливи на систему S будуть активувати функційні елементи і, відповідно, змінюватимуть їх стани. За результатами опрацьовування впливів будуть наслідки, за якими можуть змінюватись стани певних функційних елементів. Враховуючи, що система S має розподілену архітектуру, то її компоненти можуть перебувати в однакових станах в певні моменти часу. При залученні засобів проти ЗПЗ та КА стани відповідних спеціалізованих функційних елементів теж можуть збігатись. Перебуваючи в активному стані певні функційні елементи можуть активізувати інші функційні елементи, тобто змінити їх стан. Все це впливатиме на прийняття рішень

щодо подальших кроків підсистемою керування підсистемою S і повинно нею враховуватись. Тому, необхідним є розроблення відповідних методів, які б враховували такі особливості в архітектурі системи керування в розрізі завдань, які до неї висуваються з самостійного визначення подальших кроків системи S. В зв'язку з такими вимогами архітектура підсистеми керування підсистемою S матиме особливості та автономність в загальній архітектурі системи S. Таким чином, функційні елементи та їх стани, а також, зокрема, функційні елементами підсистеми керування зі своїми станами, будуть мати однакові способи переходу зі стану в стан, але наслідки з цих переходів хоча, і породжуватимуться ними та впливатимуть в прямому і зворотному напрямах, все-таки функційні елементи підсистеми керування матимуть вищий пріоритет щодо породження впливів на решту функційних елементів.

2.4 Модель функціонування обманних систем згідно популяційних алгоритмів

В процесі свого функціонування обманні системи згідно моделі їх задання (формули (2.8) та (2.9)) повинні самостійно обирати свої наступні кроки. Особливо це актуально в момент здійснення КА на ресурси та засоби корпоративних мереж. Варіантів таких кроків, як правило, є багато. В момент здійснення КА система може мати багато варіантів для відповіді на них. Також, під час штатного функціонування вона повинна реагувати на зміни в кількості її компонент, бо частина комп'ютерних станцій може вимикатись та вмикатись тим самим змінюючи склад системи. Приманки та пастки, які є частиною системи можуть виконувати певні завдання, або не виконувати, або частина з них буде виконувати і решта не буде виконувати. Тому, варіантів для вибору в таких системах багато. Вони не можуть здійснювати повний перебір таких варіантів, оскільки стан вузлів корпоративних мереж стрімко змінюється і актуальність частини варіантів для прийняття їх в якості можливого наступного кроку системи швидко втрачається. В зв'язку з цим, а також з тим, що відповіді та поведінка системи в момент КА повинна бути різною, необхідно в архітектурі обманних систем реалізувати модель такої поведінки, яка б дозволила уникнути повного перебору варіантів, не вела б до деградації та розбалансування дій, забезпечувала б ефективне функціонування, складність розуміння зловмисниками, не

була б збіжною за невелику кількість кроків та могла б підтримувати дії системи щодо певних впливів напротязі часу здійснення таких впливів. Таким чином, основними вимогами до організації функціонування обманних систем є забезпечення сталої поведінки під час тривалих КА та під час тривалої експлуатації.

При виборі з потенційних варіантів подальших кроків обманних систем необхідно розв’язувати задачу оптимізації, в якій множина розв’язків є дискретною чи бути зведеною до дискретної і є варіантами кроків. Така задача є задачею комбінаторної оптимізації і, при цьому, обраними мають бути не окремі одиничні наступні кроки системи, а послідовності кроків. В цих послідовностях кроків можуть змінюватись наступні кроки, але перші кроки повинні пов’язувати декілька наступних кроків. Так має бути сформовано послідовність кроків. Це доповнює вимоги та обмеження для таких задач комбінаторної оптимізації.

Оскільки зловмисники при проведенні КА використовують розроблені моделі поведінки, які все-таки переважно відповідають їх характерним рисам поведінки або типовим стратегіям поведінки зловмисників, то для забезпечення вимоги щодо підтримання тривалої роботи ОСПП в момент здійснення КА моделі поведінки обманних систем на противагу моделям КА повинні містити алгоритми оптимізації, які відображають природню поведінку. Тоді, зловмисникам будуть протидіяти обманні системи, які функціонуватимуть згідно поведінкових моделей.

Для розроблення такої поведінкової моделі, щоб забезпечити функціонування обманних систем з врахуванням того, що на кожному кроці потрібно буде обирати один варіант з багатьох, використаємо алгоритм, який обирають для вирішення задач, коли точне рішення невідоме або його отримання є занадто складним і затратним за часом. Він заснований на певному оцінюванні та досвіді, який формується з часом для того, щоб знайти прийнятне рішення у більшості випадків, хоча і не обов'язково оптимальне. Алгоритм для моделі виберемо з множини популяційних алгоритмів, бо вони характеризуються тим, що на кожному наступному кроці опрацьовують декілька розв’язків одночасно, а результати пошуку та вибору розв’язків формують та розширюють досвід, зокрема і колективний досвід для агентів. Взагалі популяційний алгоритм буде відображати роботу багатьох агентів в системі. Клас з множини популяційних алгоритмів для використання їх в поведінковій моделі обманних систем, враховуючи специфіку функціонування та завдань, візьмемо з

алгоритмів, які натхненні живою природою. На основі такого вибору задамо концептуальну модель обманних систем з приманками та пастками згідно формул (2.8) та (2.9) і вимоги щодо формування поведінки згідно популяційних алгоритмів так:

$$M_{S_o} = \left(S_o, A_{o,p}, G_{o,p}\right), \tag{2.28}$$

де S_o – обманна система, яку задано елементами множини за формулами (2.8) та (2.9); $A_{o,p}$ – множина популяційних алгоритмів, елементи якої означують функційний елемент $s_{i,0,6}$ (рис. 2.8) підсистеми керування для визначення подальших кроків у функціонуванні системи; $G_{o,p}$ – граф зв'язків між елементами множини S_o та одним з елементів множини $A_{o,p}$.

Згідно концептуальної моделі M_{S_o} (формула (2.8)) наявність в системі елементу з множини популяційних алгоритмів $A_{o,p}$ є обов'язковим. Тоді, зображення графу зв'язків $G_{o,p}$ між елементами множини S_o та одним з елементів множини $A_{o,p}$ задамо на рис. 2.9.

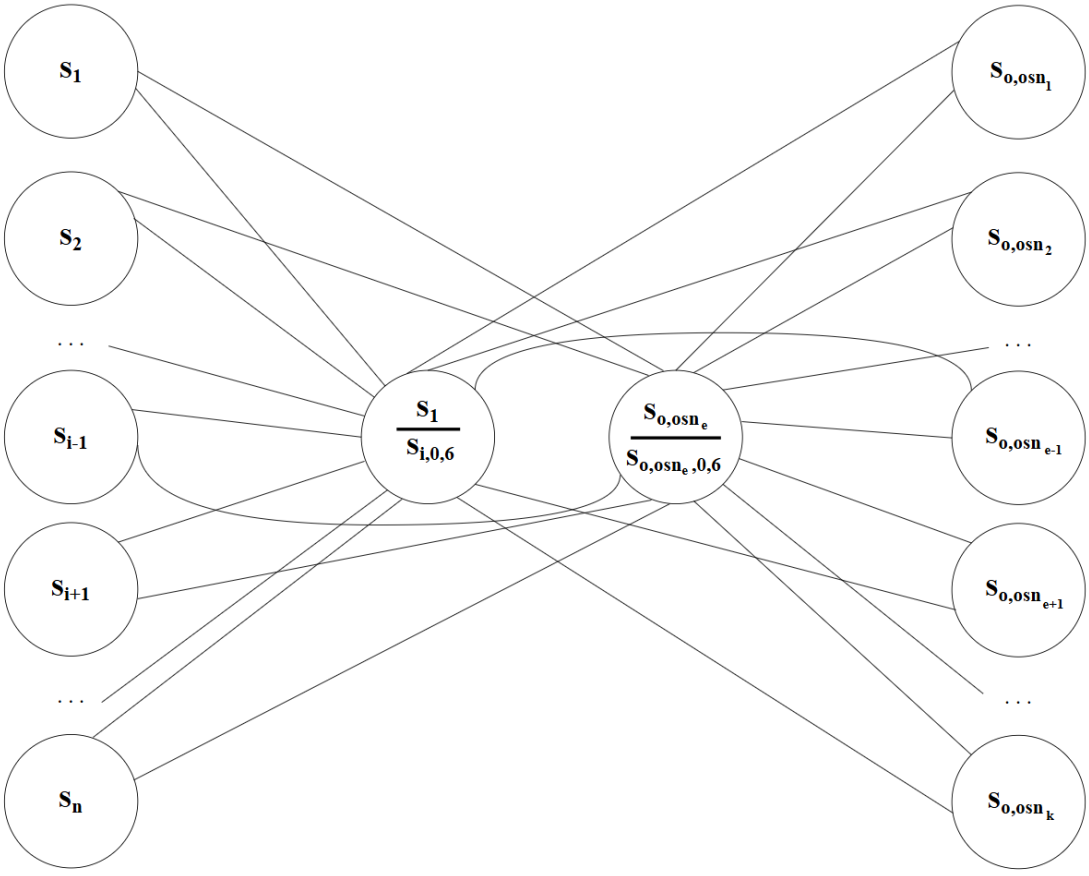


Рисунок 2.9 - Граф зв'язків $G_{o,p}$

Граф зв'язків $G_{o,p}$ є підграфом повного графу, в якому вершини, що відображають елементи множин основної частини обманних систем без приманок і пасток $S_{o,osn}$ і частини обманних систем саме з приманками і пастками S, пов'язані між собою. На рис. 2.9 такі позначення: n – це кількість приманок і пасток в системі; $n \geq 2$, S_i - i-та приманка чи пастка; k – це кількість компонентів та модульних елементів основної частини обманної системи; $k \geq 1$, S_{o,osn_j} - j-тий компонент або модульний елемент; $j = 1,2,\dots,k$. Граф зв'язків $G_{o,p}$ формується динамічно в системі в залежності від компоненти, в якій будуть прийматись рішення щодо подальших кроків згідно популяційного алгоритму. При зміні компоненти граф зв'язків $G_{o,p}$ буде змінено автоматично обманною системою.

Функційний елемент $s_{i,0,6}$ з підсистеми S має такого ж відповідника $s^*_{o,osn_l,0/6}$ з підсистеми $S_{o,osn}$. Центр системи означується наявністю саме одним з цих елементів.

Таким чином, особливістю розробленої концептуальної моделі ОСПП в корпоративних мережах є поділ компонентів на компоненти підсистеми, якою забезпечуються обманні технології, та підсистеми безпосередньо з приманками і пастками, а також наявністю в них підсистеми прийняття рішень на основі популяційних алгоритмів для заплутування зловмисників та підтримування стійкої відповіді на цілеспрямовані тривалі зловмисні дії в процесі здійснення двоцільових атак.

Розглянемо детальніше популяційні алгоритми в контексті їх використання в обманних системах для забезпечення підтримки тривалої поведінки системи на певну тривалу зловмисну дію чи КА. Такі алгоритми вирішують задачі оптимізації. Оптимізація стосується вибору наступних кроків системи з певної кількості можливих кроків для виконання завдань системи. Важливою особливістю при виборі має бути спроможність системи з використанням певного популяційного алгоритму не тільки виконати декілька наступних кроків, але підтримувати їх вибір тривалий час поки здійснюється певна КА чи дії ЗПЗ. Якщо є система вибиратиме кроки і через певний час такий вибір вичерпується, тобто вона починає деградувати, то це не дозволяє виконати належну протидію КА та ЗПЗ і переважно пов'язано з передчасною збіжністю реалізованого в ній методу.

При виборі наступних кроків системи з потенційних кроків може виникнути ситуація, коли система обиратиме кожного разу з меншої кількості варіантів і врешті кількість кроків для вибору стане нульовою, а КА чи дії ЗПЗ будуть продовжуватись. Таким чином, виникне подія, суть якої в передчасній збіжності і вона буде відображати стагнацію методу оптимізації в локальному оптимумі. Це перешкоджає збіжності методу до глобального оптимуму. Алгоритми, які базуються на популяціях, мають високу здатність уникати локального оптимуму, бо під час оптимізації вони опрацьовують набір рішень. Також, з реалізацією мультиагентних технологій в архітектурі обманних систем можливий обмін інформацією між компонентами та організація допомоги їм у подоланні проблем в пошукових просторах.

Популяційних алгоритмів відомо багато. Досить часто значна частина з них може не бути представлена деталізованою послідовністю кроків, як того вимагає означення та властивості алгоритмів, а є лише ілюстрацією ідеї чи дій, які натхненні природою, тобто не мають розроблених загальноприйнятих кроків за винятком загальної структури, за якою вони відносяться до популяційних алгоритмів, та відповідної загальноприйнятої ідентифікації.

В контексті розв’язуваної задачі щодо покращення функціонування обманних систем в частині забезпечення тривалого реагування на тривалу КА чи тривалі дії ЗПЗ, а також організації використання приманок і пасток, виникає модель відносин між діями та засобами зловмисників з однієї сторони та діями ОСПП з другої сторони. Суть цих дій полягає в тому, що КА та ЗПЗ спрямовані і переміщуються в напряму визначених зловмисниками об’єктів в корпоративних мережах, а назустріч їм для забезпечення безпеки та захисту корпоративних мереж та їх ресурсів спрямовані приманки і пастки, а також компоненти і елементи обманних систем, ховаючи за собою реальні об’єкти. Результатом контактів між засобами проведення КА та ЗПЗ з однієї сторони та приманками і пастками обманних систем будуть такі варіанти:

1) КА чи ЗПЗ завершились в приманках чи пастках;

2) КА чи ЗПЗ потрапили в приманки чи пастки, але при цьому їх активність продовжилась на інших напрямах;

3) КА чи ЗПЗ надали змогу зловмисникам отримати контроль над приманками чи пастками, з якими увійшли в контакт.

КА чи ЗПЗ можуть також обійти приманки чи пастки. А також, при цьому, продовжити свої дії для досягнення мети.

Обманні системи у варіанті з втратою приманок і пасток за певними ознаками можуть побачити таку втрату приманок і пасток та повинні використати захисні засоби для уникнення їх використання зловмисником. Тому, приманки і пастки потрібно застосовувати в складі обманних систем, які можуть забезпечити контроль їх застосування. Якщо є обманні системи не помітили втрати контролю над приманками і пастками, тоді рівень безпеки корпоративних мереж може бути знижений через впливи зловмисників. Для уникнення таких подій необхідні засоби в обманних системах, що відповідатимуть за контроль за приманками і пастками. Інакше, зловмисник успішно класифікує хибні об’єкти для атак, що відповідає двоцільовій моделі КА.

Назустріч КА та ЗПЗ можуть бути спрямовані декілька приманок і пасток. А також, КА і ЗПЗ можуть мати декілька цілей в корпоративних мережах, зокрема, вони можуть проводитись одночасно і, при цьому, бути різними. Тоді, обманні системи повинні забезпечувати всі наявні вектори КА та дій ЗПЗ відповідними хибними об’єктами для атак. Зловмисники теж можуть використовувати хибні КА чи організовувати хибні впливи ЗПЗ для відволікання ресурсів і часу систем безпеки та захисту корпоративних мереж.

В табл. 2.1 подано потенційне поле подій при проведенні зловмисниками двоцільових атак та використанні для забезпечення безпеки та захисту корпоративних мереж ОСПП.

Таблиця 2.1 - Поле подій

№ з/п	Варіанти подій	Зловмисник здійснив одну двоцільову КА	Зловмисник здійснив більше однієї різних двоцільових КА	ЗПЗ здійснило поширення в одній комп’ютерній станції	ЗПЗ здійснило поширення в багатьох комп’ютерних станціях
1	2	3	4	5	6
1	КА чи ЗПЗ завершились в приманках чи пастках;	+	+	+	+

Кінець таблиці 2.1

1	2	3	4	5	6
2	КА чи ЗПЗ потрапили в приманки чи пастки, але при цьому їх активність продовжилась на інших напрямах;	+	+	+	+
3	КА чи ЗПЗ надали змогу зловмисникам отримати контроль над приманками чи пастками, з якими увійшли в контакт	+	+	+	+
4	КА чи ЗПЗ досягли мети зловмисників і не були помічені обманними системами з приманками і пастками	+	+	+	+
5	КА чи ЗПЗ не досягли мети зловмисників і не були помічені обманними системами з приманками і пастками	+	+	+	+

З табл. 2.1 можна встановити що при одній чи більше однієї різних двоцільових КА і діях ЗПЗ обманна система повинна мати приманки і пастки для них і мати змогу їх перерозподіляти в залежності від наявних загроз. Також, з отриманого поля подій

в табл. 2.1 можна встановити закономірність для КА і ЗПЗ щодо їх переміщення в корпоративних мережах та різних результатів контактів із хибними об’єктами для атак. Крім того, згідно рис. 2.5 в архітектурі обманних систем її компоненти, тобто хибні об’єкти для атак, можуть бути окремими об’єктами, можуть бути в реальних комп’ютерних станціях та серверах. При цьому елементи та компоненти обманних систем можуть бути у вузлах комп’ютерних мереж для контролю або виконувати функції хибних об’єктів для атак працюючи поряд паралельно з реальними інформаційними системами. Таким чином, такий розподіл компонентів та елементів ОСПП в співвіднесенні з варіантами подій з табл. 2.1 збільшує кількість можливих варіантів подій до двадцяти для загальних випадків і до вісімдесяти для конкретних випадків. Всі вони повинні бути враховані в моделях подій, які опрацьовуватимуть ОСПП.

Обманні системи повинні спрямовувати КА та ЗПЗ на хибні об’єкти для атак. Крім того, вибір наступних кроків обманних систем і, зокрема, кроків для активізації та наближення приманок і пасток, передбачає вирішення завдання оптимізації. Якщо КА та ЗПЗ завершились як активності в приманках та пастках, повністю або частково, то така подія буде відповідати алгоритму молі і полум’я. Таким чином, вибір приманок та пасток для КА та дій ЗПЗ є кроками системи, які потребують оптимізації.

Алгоритм молі і полум’я відноситься до популяційних алгоритмів та вирішує завдання оптимізації і належить до класу алгоритмів натхненних природою. Суть натхнення природою для цього алгоритму полягає в способі руху молі в природі. Загально прийнято цей рух називають поперечною орієнтацією. Моль літає вночі. При побудові маршруту вона зберігає фіксований кут по відношенню до Місяця, завдяки чому може долати великі відстані по прямій лінії. За другим випадком вона попадає в пастку на спіральній доріжці навколо штучного освітлення. Ці два випадки є основою для досягнення оптимізації при розв’язуванні задач з використанням алгоритму молі і полум’я. Для застосування алгоритму введемо позначення об’єктів корпоративних мереж та встановимо зв’язки між ними.

Задамо вузли корпоративних мереж множиною $M_{kn,v} = \left\{m_{kn,v,1}, m_{kn,v,2}, \dots, m_{kn,v,\,N_{kn,v}}\right\}$, де $N_{kn,v}$ – кількість вузлів корпоративних мереж. Здійснимо їх поділ на підмножини за ознакою належності до певного сегменту корпоративних мереж, тоді

$$M_{kn,p} = \bigcup_{q=1}^{N_{kn,v,p}} M_{kn,v,p,q}, \quad (2.29)$$

де $M_{kn,v,p,q}$ – підмножини множини $M_{kn,p}$; $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$; $q = 1,2,\dots,N_{kn,v,p}$.

Задамо підмножини $M_{kn,v,p,q}$ ($q = 1,2,\dots,N_{kn,v,p}$, $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$) з формули (2.29) матрицею $M_{kn,v,p,z}$, в якій елементами рядків будуть вузли корпоративних мереж, що належатимуть до одного сегменту, так:

$$M_{kn,v,p,z} = \begin{pmatrix} m_{kn,v,p,z,1,1} & \cdots & m_{kn,v,p,z,1,N_{kn,v,p,z}} \\ \vdots & \ddots & \vdots \\ m_{kn,v,p,z,N_{kn,v,p}} & \cdots & m_{kn,v,p,z,N_{kn,v,p},N_{kn,v,p,z}} \end{pmatrix}, \quad (2.30)$$

де $m_{kn,v,p,z,i,j}$ – елемент підмножини $M_{kn,v,p,i}$ ($i = 1,2,\dots,N_{kn,v,p}$, $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$); $N_{kn,v,p,z}$ – максимальна кількість вузлів в сегментах мережі включно з маршрутизатором; j – номер стовпця матриці; $j = 1,2,\dots,N_{kn,v,p,z,N_{kn,v,p,z,j}}$; $j = 1,2,\dots,N_{kn,v,p,z,j}$; $N_{kn,v,p,z,j}$ – кількість вузлів в j-му сегменті мережі включно з маршрутизатором; $N_{kn,v,p,z,j} \le N_{kn,v,p,z}$; $i = 1,2,\dots,N_{kn,v,p}$; $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$.

Якщо $N_{kn,v,p,z,j} < N_{kn,v,p,z}$ для частини підмножини, то всі елементи матриці $M_{kn,v,p,z}$ від $N_{kn,v,p,z,j} + 1$ по $N_{kn,v,p,z}$ для всіх рядків, в яких виконується це співвідношення, встановлюємо таким, що дорівнює $p_{M_{kn,v,p,z}}$.

В матриці $M_{kn,v,p,z}$ відображено групування вузлів корпоративних мереж за сегментами. Встановимо порядок між вузлами в сегментах корпоративних мереж та задамо їх матрицею векторів $M_{kn,v,p,z,vect}$ на основі матриці $M_{kn,v,p,z}$ (формула (2.30)) так:

$$M_{kn,v,p,z,vect} = \begin{pmatrix} m_{kn,v,p,z,vect,1,1} & \cdots & m_{kn,v,p,z,vect,1,N_{kn,v,p,z}} \\ \vdots & \ddots & \vdots \\ m_{kn,v,p,z,vect,N_{kn,v,p},1} & \cdots & m_{kn,v,p,z,vect,N_{kn,v,p},N_{kn,v,p,z}} \end{pmatrix}, \quad (2.31)$$

де $m_{kn,v,p,z,vect,i,j}$ – компонента вектору з елементів підмножини $M_{kn,v,p,i}$ ($i = 1,2,\dots,N_{kn,v,p}$, $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині

$M_{kn,p}$); $N_{kn,v,p,z}$ – максимальна кількість вузлів в сегментах мережі включно з маршрутизатором; j – номер стовпця матриці; $j = 1,2,\dots,N_{kn,v,p,z,N_{kn,v,p,z,j}}$; $j = 1,2,\dots,N_{kn,v,p,z,j}$; $N_{kn,v,p,z,j}$ - кількість вузлів в j-му сегменті мережі включно з маршрутизатором; $N_{kn,v,p,z,j} \le N_{kn,v,p,z}$; $i = 1,2,\dots,N_{kn,v,p}$; $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$.

На рис. 2.10 зображено розміщення вузлів корпоративних мереж, які задано формулами (2.29) – (2.31).

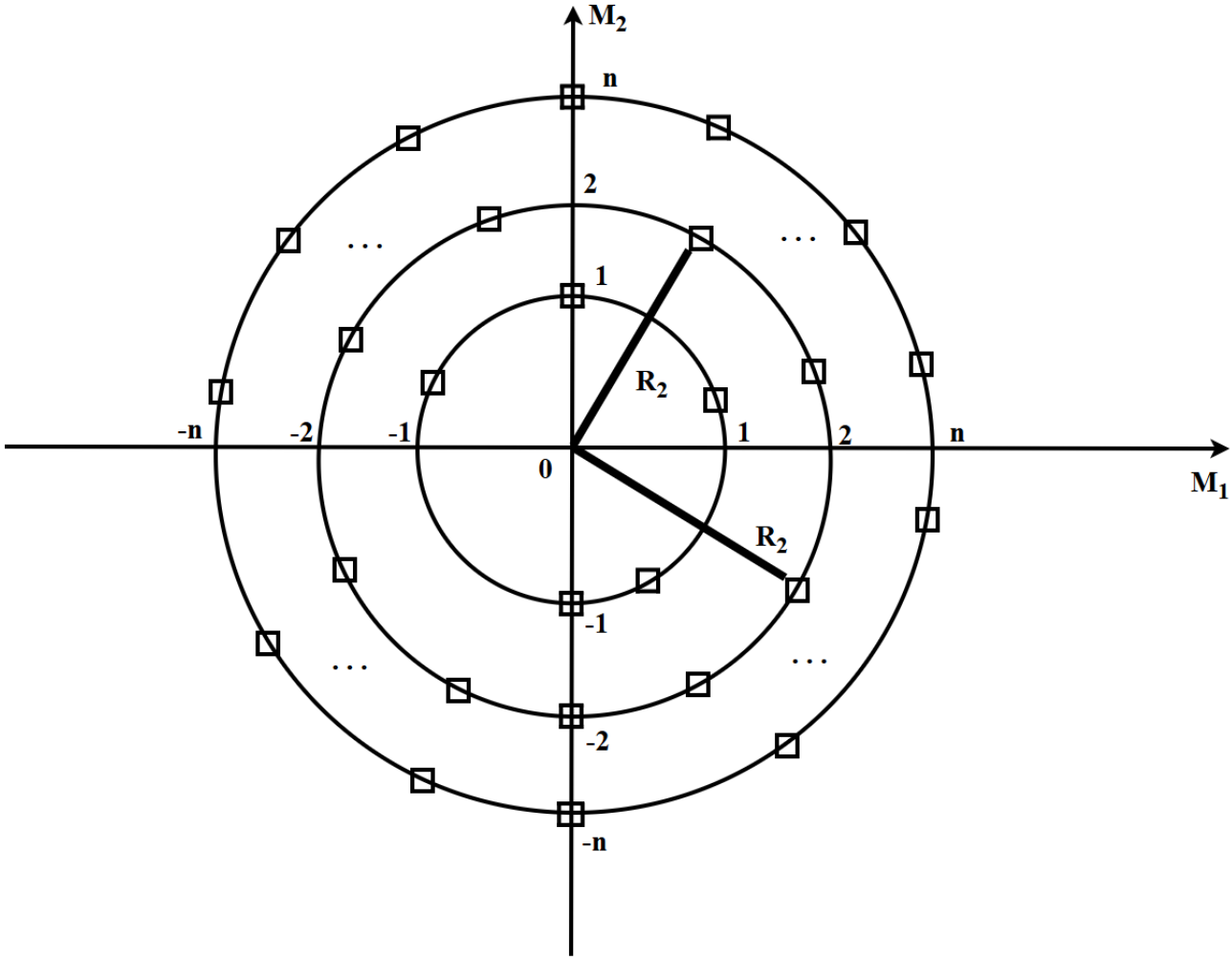


Рисунок 2.10 – Подання вузлів корпоративних мереж на координатній площині

Квадрати кожного кола відображають вузли корпоративних мереж, які належать одному сегменту. Перше коло відображає сегмент, в якому є засоби під'єднання до глобальних мереж, а також, його вузли можуть формувати демілітаризовану зону мережі. Чим далі кола знаходяться від центру координат тим вищий рівень безпеки у комп'ютерних станціях або тим довше за часом передається повідомлення від центру в точці О до відповідного кола порівняно з ближчими

колами. Відстань до вузлів, які зображено квадратами, до центру в точці O є однаковою для всіх них, якщо вони знаходяться на лінії певного кола. Наприклад, R_2 – є радіусом другого кола і два відрізки відображають однакову відстань до двох вузлів в одному сегменті мережі.

Порядок вузлів, які розміщені на певних колах, може змінюватись обманними системами. Частина вузлів в процесі КА може вимикатись ними або блокуватись. Тоді, їх можна зображати різними кольорами на колах в залежності від доступності до них і їх активності.

В контексті обманних систем введемо множину впливів на корпоративні мережі, які можуть здійснювати зловмисники з використанням ЗПЗ та проведенням КА, і множину реакцій на впливи. При цьому, зловмисники як правило здійснюють атаки за масками сегментів, яким відповідають кола на рис. 2.10, тобто вони здійснюють рух між вузлами конкретного сегменту, поки не здійснять їх повну розвідку.

Задамо множину дій M_{vp} зловмисників, які спричиняють впливи на корпоративні мережі і реалізують КА та поширення ЗПЗ, згруповано за типами так:

$$M_{vp} = \begin{pmatrix} m_{vp,1} \\ m_{vp.2} \\ \cdots \\ m_{vp,N_{vp}} \end{pmatrix}, \tag{2.32}$$

де $m_{vp.q}$ - q-а дія зловмисників, що реалізує вплив на корпоративні мережі для здійснення КА або поширення ЗПЗ, і характеризує певний тип таких дій; $q = 1,2, \dots, N_{vp}$; N_{vp} – кількість типів дій зловмисників.

Дії зловмисників, які задано множиною M_{vp} за формулою (2.32) характеризують певний тип і формують типовий клас. Цей типовий клас характеризує певний тип КА чи дії ЗПЗ. Перелік дій в самому класі може бути різним за кількістю етапів, послідовністю та залученими функціями, але він характеризує спорідненість їх та однакову спрямованість на корпоративні мережі. Такий поділ є необхідними для формування реакцій обманних систем, які можна задати окремою множиною реакцій та реалізувати як частину кроків для забезпечення безпеки та захисту корпоративних мереж. На певний типовий клас дій зловмисника як правило розробляють відповідні реакції систем. Тому, для обманних систем з приманками та пастками необхідно

задати множину дій або реакцій на типові або конкретні дії зловмисників, які здійснюють через впливи на корпоративні мережі.

Введемо множину дій $M_{os,d}$ ОСПП, зокрема і тих, які є реакціями на зловмисні впливи на корпоративні мережі, так:

$$M_{os,d} = \begin{pmatrix} m_{os,d,1} \\ m_{os,d.2} \\ \dots \\ m_{os,d,N_{os,d}} \end{pmatrix}, \quad (2.33)$$

де $m_{os,d.q}$ - q-а дія ОСПП, що реалізує також реакції на зловмисні впливи на корпоративні мережі для здійснення КА або поширення ЗПЗ, і характеризує певний тип таких дій; $q = 1,2, \dots, N_{os,d}$; $N_{os,d}$ – кількість дій ОСПП.

Множини дій зловмисників M_{vp} та дій ОСПП $M_{os,d}$, які визначені за формулами (2.32) і (2.33), є детермінованими щодо використовуваних команд. Кількість команд може з часом бути збільшена, але їх кількість скінчена і залежить від наявних операційних систем, систем програмування та апаратно-програмних засобів, які використовуються зловмисниками та користувачами корпоративних мереж. Тому, множини дій зловмисників та ОСПП мають є скінченними, але кожна з дій може бути подана різною кількістю допустимих в мережах та системах команд. Таким чином, множину дій $M_{os,d}$ ОСПП розділимо на такі підмножини:

1) $M_{os,d,1}$ – підмножина дій, які забезпечують функціонування безпосередньо обманних систем без приманок і пасток;

2) $M_{os,d,2}$ – підмножина дій, які забезпечують функціонування обманних систем винятково в частині керування приманками і пастками;

3) $M_{os,d,3}$ – підмножина дій, які забезпечують функціонування безпосередньо обманних систем без приманок і пасток в частині реакції на дії та впливи в корпоративних мережах включно з штатними та із зловмисними діями;

4) $M_{os,d,4}$ – підмножина дій, які забезпечують функціонування обманних систем для залучення приманок і пасток з метою реагування на впливи та дії в корпоративних мережах;

5) $M_{os,d,5}$ – підмножина дій, які забезпечують функціонування винятково приманок і пасток без залучення керування обманними системами.

Підмножини дій $M_{os,d,3}$ та $M_{os,d,4}$ відносяться до таких, що відповідають за опрацювання множини дій M_{vp} зловмисників. Вони спрямовані на здійснення класифікації подій в корпоративних мережах, які можуть бути віднесені до КА чи дій ЗПЗ.

В підмножинах дій $M_{os,d,1}$ та $M_{os,d,2}$ наявні елементи, які забезпечують адаптивність та перебудову обманних систем включно з керуванням приманками і пастками за наявності необхідних передумов. Частина дій з цих підмножин може поєднуватись з діями підмножин $M_{os,d,3}$ та $M_{os,d,4}$. Особливо коли для здійснення класифікації подій в корпоративних мережах в процесі здійснення КА чи поширення ЗПЗ зловмисниками необхідно здійснити зміну архітектури основної частини обманних систем, активізувати частину приманко і пасток тощо. Тобто, крім функціоналу з виявлення КА та дій ЗПЗ, який забезпечується діями підмножин $M_{os,d,3}$ та/або $M_{os,d,4}$ потрібно забезпечити виконання дій з підмножин $M_{os,d,1}$ та/або $M_{os,d,2}$.

Всі ці підмножини дають змогу формувати нові послідовності кроків обманних систем, тобто є основою для формування послідовності кроків обманних систем при тривалих атаках. Безпосередньо підмножини множини дій $M_{os,d}$ ОСПП містять конкретні послідовності кроків для виявлення та/чи класифікації КА та дій ЗПЗ, але при цьому в послідовностях цих кроків можуть бути засоби, які не завжди є чітко детермінованим, а можуть містити кроки, в яких задано ітераційні процеси тощо. Але такі кроки будемо розглядати одиничними кроками з урахуванням того, що при їх виконанні не повинно бути переривання та переключення на інші кроки, тобто вони повинні отримувати всі необхідні ресурси комп'ютерних систем та виконуватись за один такт. В такому позиціонуванні вони будуть розглядатись як окремі кроки із послідовності загальних кроків, які повинна визначити для себе кожна обманна система в залежності від поточних подій і стану корпоративних мереж. Таким чином, задамо послідовність кроків ОСПП такими векторами:

$$v_{os,i} = \left(v_{os,i,1}, v_{os,i,2}, \dots, v_{os,i,N_{os,i}}\right), \tag{2.34}$$

де $v_{os,i,j}$ – крок ОСПП; $j = 1,2, \dots, N_{os,i}$; $N_{os,i}$ – кількість кроків ОСПП для вирішення певного завдання, які визначено системою в поточний момент часу; i – i-та послідовність кроків для вирішення певного завдання системи.

Кроки обманних систем можуть обиратись з врахування попередньої історії з опрацювання повторюваних подій або частково використовувати такі відомості. Але в архітектуру обманних систем закладено послідовності кроків для вирішення типових відомих КА та дій ЗПЗ. Крім того, такі обманні системи повинні мати засоби для модифікації своїх наступних кроків на основі використання базових кроків. До таких кроків відносяться кроки із керування та залучення до процесу виявлення КА та дій ЗПЗ приманок та пасток. Враховуючи специфіку обманних систем, то підбір кроків повинен бути таким, щоб уникати повторюваності дій, інакше зловмисники зможуть вивчати поведінку засобів та систем захисту корпоративних мереж. Тобто, реакції обманних систем на впливи, причому на певні конкретні впливи, можуть бути різними, тому і послідовності кроків, які визначатимуться такими системами будуть різними.

Серед кроків ОСПП важливими є такі, що впливають на активність та розміщення приманок і пасток, безпосередньо під час впливів на корпоративні мережі. При виборі таких послідовностей кроків виникають проблеми, які пов’язані із здійсненням повного перебору варіантів та вибору одного з них. Але така стратегія не є ефективною, оскільки варіантів послідовностей кроків дуже багато і повний перебір всіх варіантів буде затратним по часу. Крім того, не має гарантії, що обраний варіант в поточний момент коли він буде виконуватись не втратив свою актуальність в швидко змінюваному середовищі корпоративних мереж. Тому, вибір такого варіанту може бути здійснено з певного локального набору кроків. Основною вимогою для такого вибору має бути критерій щодо часу та актуальності підібраного вектору послідовності кроків. Крім того, обрана послідовність кроків повинна бути такою, щоб адекватно відповідати на виявлені впливи в корпоративних мережах від КА та дій ЗПЗ. Першочергово така відповідність стосується забезпечення тривалої відповіді на здійснювані впливи, тобто уникнення збіжності обраного варіанту послідовності кроків, коли впливи продовжуються, а кроки обманних систем вже вичерпались.

Підмножина кроків з векторів, які задають послідовності кроків обманних систем, може відноситись до кроків з керування приманками і пастками. Тоді, використання популяційного алгоритму, наприклад алгоритму молі і полум’я, в архітектурі ОСПП може відноситись, наприклад, до прийняття рішень щодо

розміщення приманок і пасток під час здійснення зловмисником КА чи виконанні дій ЗПЗ, які були встановлені давачами в корпоративних мережах.

Розглянемо спочатку конфігурування приманок і пасток для випадку, коли здійснюється КА на корпоративні мережі. Тоді, точку потенційного проникнення в корпоративну мережу можемо прийняти за точку О (0;0) на координатній площині (рис. 2.10), а комп'ютерні станції, які зображені на рис. 2.10, можна прийняти такими, що серед них будуть приманки і пастки окремими вузлами або поєднаними з реальними комп'ютерними станціями чи серверами. В такій інтерпретації алгоритму молі і полум'я вплив від КА буде активувати рух приманок і пасток до нього для його нівелювання або фіксування в пастках. Традиційна схема алгоритму молі і полум'я передбачає рух молі, який в даному випадку є відповідником КА, до джерела штучного освітлення, яке є відповідником приманці чи пастці. Тоді, другим випадком для конфігурування приманок і пасток є такий, при якому КА чи дії ЗПЗ відбувають у вузлах корпоративних мереж та зміщують в сторону приманки і пастки, які активовані обманною системою і позиціонують себе як ресурс, що є метою КА чи дій ЗПЗ.

Якщо розглядати дії ЗПЗ, включно з поширенням між вузлами корпоративних мереж, то вузол, в якому активізується ЗПЗ, на рис. 2.10 можна вважати таким, що представляє точку О. Тоді, решта вузлів будуть формувати оточення цього вузла і в частині з них будуть приманки і пастки окремо, а в частині будуть приманки і пастки разом з компонентами реальних інформаційних систем. При активізації ЗПЗ приманки і пастки будуть активізовані обманними системами та будуть себе проявляти в напряму руху до точки О. Або може бути застосований зворотній погляд на використання алгоритму молі і полум'я, в якому активізація ЗПЗ здійснюється у певному вузлі на рис. 2.10, а далі здійснюється його поширення в напряму до приманки чи пастки, які знаходяться в центрі координат, тобто в точці О.

Згідно двох випадків можна виділити дві схеми та такі варіанти:

1) КА чи дії ЗПЗ рухаються по спіралі або по прямій, а в центрі системи координат розміщена приманка чи пастка, або об'єкт, який є метою зловмисника;

2) КА чи дії ЗПЗ рухаються по спіралі або по прямій, а в центрі системи координат розміщено приманку чи пастку, або об'єкт, який є метою зловмисника, і наявні багато об'єктів, на які спрямована одна атака;

3) КА чи дії ЗПЗ можна розглядати як об’єкт (центр освітлення) в точці О системи координат на рис. 2.10, до якого рухаються приманки по спіралі або по умовній прямій (дотична до кола);

4) КА чи дії ЗПЗ можна розглядати як об’єкт (центр освітлення) в точці О системи координат на рис. 2.10, до якого рухаються приманки по спіралі або по умовній прямій (дотична до кола) для випадку більше двох різних атак і, тоді, координатних систем буде більше двох або процес може бути відображено в паралельних площинах;

5) поєднання першого і третього варіантів;

6) поєднання другого і четвертого варіантів.

Перший і четвертий варіанти та другий і третій варіанти є не поєднувані між собою. Для різних атак можуть масштабуватись перший і другий варіанти.

Узагальнення введених паралельних площин та координатної площини, в яких зображено КА чи дії ЗПЗ та об’єкти корпоративних мереж включно з компонентами ОСПП, є поверхнями атак,

На рис. 2.10 рух об’єктів може здійснювати за двома траєкторіями.

Перший тип траєкторії стосується руху, який забезпечує поперечну орієнтацію і за своєю суттю є рухом по колу, тобто по дотичній до кола. Такий рух по умовній прямій характеризує те, що КА чи дії ЗПЗ не можуть досягти, не бачать об’єкту, який є метою зловмисника, або не можуть увійти в інші вузли комп’ютерних мереж),

Другий тип траєкторії стосується руху, який формує спіраль в напряму до центра системи координат, тобто до об’єкту, який цікавить зловмисника або який цікавить обманну систему з приманками і пастками, якщо розглядати такий рух в контексті двох потенційних схем для об’єктів, що зображені на рис. 2.10.

Таким чином, конфігурація вузлів у співвіднесенні з проникненням через засоби доступу корпоративних мереж до глобальних мереж або з середини системи в контексті використання алгоритму молі і полум’я для КА і дій ЗПЗ зводиться до однієї з двох схем представлення і може бути реалізована в архітектурі ОСПП. Дві схеми представлення, в яких відмінність полягає в трактуванні центру системи координат, можуть бути узагальнені до однієї схеми або розглядатись в тривимірному просторі, де на координатній площині подано одну схему і третя координата буде надавати можливість формувати ще одну площину для другої схеми. Тоді, дві схеми будуть

поєднані рух КА і дій ЗПЗ з однієї сторони буде рухом до приманок і пасток, а з другої сторони рух приманок і пасток до впливів КА і дій ЗПЗ. Підтримка зміни активності приманок і пасток в обманних системах дасть змогу забезпечити тривалу взаємодію з впливами в корпоративних мережах, які породжені КА чи діями ЗПЗ. Оптимізація при застосуванні популяційних алгоритмів в архітектурі ОСПП, зокрема алгоритму молі і полум'я, для формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ дає змогу уникнути повного перебору варіантів, не допустити швидкої збіжності обраних кроків при триваючих впливах та скоригувати послідовність кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж.

2.5 Висновки до другого розділу

Проблема покращення виявлення КА та дій ЗПЗ в корпоративних мережах залишається актуальною та є багатоаспектною в контексті показників, параметрів тощо, які можуть розглядатись такими для удосконалення їх безпосередньо або оновлення і які впливатимуть на вирішення проблеми. Для покращення виявлення КА та дій ЗПЗ в корпоративних мережах можуть бути оновлені або удосконалені моделі атак, архітектура ОСПП, синтез нових підсистем в архітектурі обманних систем, формування різних послідовностей наступних кроків систем із можливістю їх зміни в процесі виконання тощо.

Розроблена модель двоцільових КА включає особливості в захисті корпоративних мереж, які пов'язані із використанням хибних об'єктів для атак та їх включенням в сценарій КА з виділенням певних етапів. Це дає змогу зловмиснику не тільки здійснювати поділ на реальні та хибні об'єкти в корпоративних мережах, але й згідно розроблюваного сценарію КА залучати хибні об'єкти, які взяті під його контроль, для виконання своїх подальших дій. Такі дії зловмисника повинні бути враховані розробниками систем безпеки та захисту корпоративних мереж, в яких використовуються додатково хибні об'єкти для атак. Організація керування і взаємодії таких хибних об'єктів для атак та їх внутрішня архітектура повинні враховувати дії зловмисника, які базуються на розробленій моделі двоцільових КА.

Архітектура ОСПП повинна надавати можливість будувати засоби, які будуть адаптивними, гнучкими, самоорганізованими, прийматимуть рішення щодо своїх

подальших кроків та щодо виявлення, а також організовуватимуть колективну робота агентів. Така архітектура ОСПП в поєднанні з відомостями в ній про наявні ресурси комп'ютерних мереж дасть перевагу над засобами зловмисників. Отримані числові характеристики елементів та компонентів обманних систем дають змогу здійснювати оцінювання їх поточного стану для визначення подальших кроків. Вони є частиною загальної сукупності характеристик і основною частиною для керування приманками і пастками під час здійснення атак.

В архітектурі ОСПП запропоновано синтезувати популяційні алгоритми, зокрема алгоритм молі і полум'я, для здійснення оптимізації наступних кроків систем та підтримки функціонування систем тривалий час в процесі протидії атакам. Встановлено, що конфігурація вузлів мереж у співвіднесенні з проникненням через засоби доступу корпоративних мереж до глобальних мереж або з середини системи в контексті використання алгоритму молі і полум'я для КА і дій ЗПЗ зводиться до однієї з двох схем представлення і може бути реалізована в архітектурі ОСПП. Дві схеми представлення, в яких відмінність полягає в трактуванні центру системи координат, можуть бути узагальнені до однієї схеми або розглядатись в тривимірному просторі, де на координатній площині подано одну схему і третя координата буде надавати можливість формувати ще одну площину для другої схеми. Підтримка зміни активності приманок і пасток в обманних системах дасть змогу забезпечити тривалу взаємодію з впливами в корпоративних мережах, які породжені КА чи діями ЗПЗ. Оптимізація при застосуванні популяційних алгоритмів в архітектурі ОСПП, зокрема алгоритму молі і полум'я, для формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ забезпечує уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж.

Основні наукові результати розділу опубліковані в працях [103; 169; 170; 134-136]

РОЗДІЛ 3
МЕТОДИ ЗАБЕЗПЕЧЕННЯ ФУНКЦІОНУВАННЯ ОБМАННИХ СИСТЕМ З ПРИМАНКАМИ І ПАСТКАМИ НА ОСНОВІ ПОПУЛЯЦІЙНИХ АЛГОРИТМІВ В КОРПОРАТИВНИХ МЕРЕЖАХ

3.1 Метод синтезу популяційних алгоритмів в архітектурі обманних систем з приманками і пастками

В архітектурі ОСПП здійснення синтезу популяційних алгоритмів, зокрема алгоритму молі і полум’я, є необхідним для оптимізації послідовності наступних кроків при їх формуванні. Під час здійснення КА та дій ЗПЗ необхідно забезпечити в таких системах уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційних спроможностей зловмисників до здійснення двоцільових КА. Тому, здійснимо спочатку розроблення методу вибору кроків ОСПП на основі алгоритму молі і полум’я як частини забезпечення їх функціонування.

Вибір популяційних алгоритмів в архітектурі обманних систем відповідає завданням, які необхідно вирішувати в них, що пов’язані з невідомими та обмеженими просторами пошуку варіантів наступних кроків (формула (2.34)). Їх використовують для уникнення передчасної збіжності, щоб оптимізація забезпечила збіжність до глобального оптимуму. Для цього може бути використано набір рішень, які можуть бути сформовані завдяки наявності ресурсів обманних систем, зокрема певної кількості приманок і пасток, а також можливості формування багатьох варіантів наступних кроків систем.

Процес оптимізації при виборі рішень поділимо на два етапи; дослідження наявних варіантів; застосування обраних варіантів. На етапі дослідження з простору пошуку обираємо варіант послідовності кроків, формуючи вектор наступних кроків обманних систем (формула (2.34)). На етапі застосування обраних варіантів здійснюємо виконання обраних кроків та покращення обраного варіанту через послідовності наступних кроків з метою наближення до глобального оптимуму. Застосування такого поділу процесу оптимізації на два етапи є можливим, оскільки

тривалість КА чи дій ЗПЗ та їх змінюваність потребують гнучкої реакції на зміну оточуючого середовища корпоративних мереж. Процес оптимізації з розділенням на два етапи зображено на рис. 3.1.

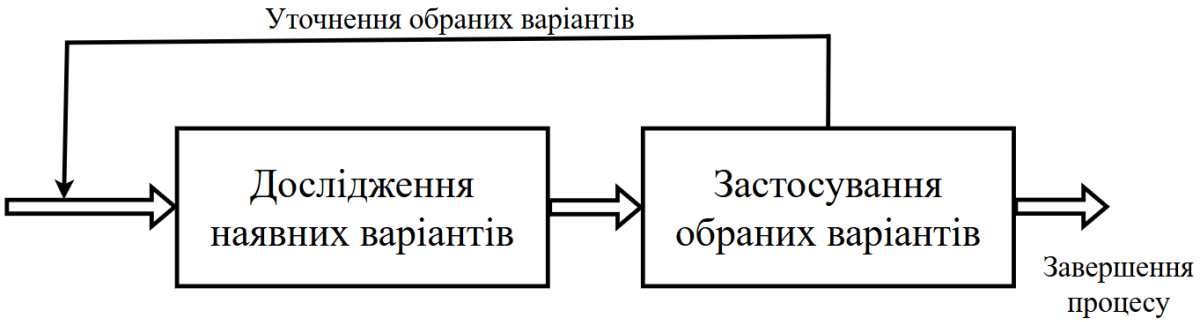


Рисунок 3.1 – Поділ процесу оптимізації на два етапи

Таким чином, спочатку в обманній системі буде сформовано вектор $v_{os,i}$ (формула (2.34)) певною заданою послідовністю кроків для забезпечення реакції на впливи в корпоративних мережах від КА чи дій ЗПЗ і це буде відповідати етапу дослідження наявних варіантів. На етапі застосування обраних варіантів частина кроків буде виконана обманною системою і в процес виконання будуть закладені кроки для оцінювання поточного стану в корпоративних мережах, стану системи та результативності кроків. Згідно опрацьованої такої інформації буде здійснено уточнення обраних варіантів кроків і сформовано новий вектор наступних кроків системи так:

$$v_{os,i,u} = \left(v_{os,i,u,1}, v_{os,i,u,2}, \dots, v_{os,i,u,N_{os,i,u}}\right), \tag{3.1}$$

де $v_{os,i,u,j}$ – крок обманної системи з приманками і пастками; $j = 1,2, \dots, N_{os,i,u}$; $N_{os,i,u}$ – кількість кроків ОСПП для вирішення певного завдання, які визначено системою в поточний момент часу; i – i-та послідовність кроків для вирішення певного завдання системи; u – уточнення кроків, які задано вектором $v_{os,i}$ (формула (2.34)).

Введемо цільову функцію оцінювання обраних варіантів так:

$$F_{os}\left(v_{os,i}, v_{os,i,u}\right) \to [0;1], \tag{3.2}$$

де $v_{os,i,u,j}$ – крок обманної системи з приманками і пастками; $j = 1,2, \dots, N_{os,i,u}$; $N_{os,i,u}$ – кількість кроків ОСПП для вирішення певного завдання, які визначено системою в поточний момент часу; i – i-та послідовність кроків для вирішення певного завдання системи; u – уточнення кроків, які задано вектором $v_{os,i}$ (формула (2.34)).

Кращим результатом оцінювання згідно цільової функції F_{os} (формула (3.2)) приймемо більше з її двох значень. Цільова функція F_{os} (формула (3.2)) дає оцінку уточненого варіанту послідовності кроків, який буде формувати вектор $v_{os,i,u}$ і відповідає в процесі оптимізації за уточнення обраних варіантів (рис. 3.1). На початку вибору рішення, коли здійснюється дослідження наявних варіантів, вона теж застосовується, але її аргументи будуть однакові, тоді вона оцінює обраний варіант, який фіксується в обманній системі. Частина кроків, які задано вектором $v_{os,i}$ (формула (2.34)) може бути повторена у векторі $v_{os,i,u}$. Застосування цільової функції буде до того часу, поки триває КА чи дії ЗПЗ, що потребуватиме формування відповідних дій ОСПП. Послідовності кроків систем будуть змінюватись і кожного разу будуть оцінюватись цільовою функцією та прямувати до досягнення глобального оптимуму. Така схема відповідає загальній структурі популяційних алгоритмів.

На рис. 2.10 зображено об’єкти корпоративних мереж, а в матриці $M_{kn,v,p,z}$ (формула (2.31)) відображено групування вузлів корпоративних мереж за сегментами. Згідно такого розподілу об’єкти корпоративних мереж розподілені між сегментами, але потрібно задати їх порядок розміщення між собою з урахуванням місць на координатній площині (рис. 2.10). Тобто, для керування приманками і пастками обманними системами потрібно їх позиціонувати і щодо решти вузлів корпоративних мереж і між собою. Тоді обманні системи будуть мати змогу залучати їх до процесу виявлення КА і дій ЗПЗ з врахуванням певних стратегій, зокрема і з використанням алгоритму молі і полум’я.

Задамо функцію кодування об’єктів корпоративних мереж на координатній площині з рис. 2.10 з врахуванням їх різних типів та формули (2.31):

$$F_{os,kod}: (M_{kn,v,p,z,vect}, O_{kn}) \to M_{os,kod}\left(X_1, X_2, \dots, X_{N_{kn,v,p,z}}, K_{os}\right), \qquad (3.3)$$

де $M_{os,kod}$ – матриця з координатами об’єктів корпоративних мереж та відомостями про тип об’єктів; K_{os} – код типу об’єктів корпоративних мереж; $K_{os} = \{k_{kod,1}, k_{kod,2}, \dots, k_{kod,N_{kn,v,p,z}}\}$; O_{kn} - тип об’єктів корпоративних мереж; X_j – вектор j – координат; $m_{kn,v,p,z,vect,i,j}$ – компонента вектору з елементів підмножини $M_{kn,v,p,i}$ ($i = 1,2, \dots, N_{kn,v,p}$, $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в

множині $M_{kn,p}$); $N_{kn,v,p,z}$ – максимальна кількість вузлів в сегментах мережі включно з маршрутизатором; j – номер стовпця матриці; $j = 1,2,\dots,N_{kn,v,p,z,N_{kn,v,p,z,j}}$; $j = 1,2,\dots,N_{kn,v,p,z,j}$; $N_{kn,v,p,z,j}$ - кількість вузлів в j-му сегменті мережі включно з маршрутизатором; $N_{kn,v,p,z,j} \le N_{kn,v,p,z}$; $i = 1,2,\dots,N_{kn,v,p}$; $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$.

Вектор j – координат X_j задамо з урахуванням формул (3.3.) та (2.31) так:

$$X_j = \begin{pmatrix} x_{kn,v,p,z,vect,1,j} \\ x_{kn,v,p,z,vect,2,j} \\ \dots \\ x_{kn,v,p,z,vect,N_{kn,v,p},j} \end{pmatrix}, \tag{3.4}$$

де $x_{kn,v,p,z,vect,i,j}$ – координата на координатній площині компоненти вектору з елементів підмножини $M_{kn,v,p,i}$ ($i = 1,2,\dots,N_{kn,v,p}$, $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$); $N_{kn,v,p,z}$ – максимальна кількість вузлів в сегментах мережі включно з маршрутизатором; j – номер стовпця матриці; $j = 1,2,\dots,N_{kn,v,p,z,N_{kn,v,p,z,j}}$; $j = 1,2,\dots,N_{kn,v,p,z,j}$; $N_{kn,v,p,z,j}$ - кількість вузлів в j-му сегменті мережі включно з маршрутизатором; $N_{kn,v,p,z,j} \le N_{kn,v,p,z}$; $i = 1,2,\dots,N_{kn,v,p}$; $N_{kn,v,p}$ – кількість підмножин вузлів корпоративних мереж в множині $M_{kn,p}$.

Множина K_{os} може містити такі елементи:

1) $k_{kod,1}$ – комп'ютерні станції з компоненти обманних систем, але не містять приманок чи пасток;

2) $k_{kod,2}$ – комп'ютерні станції з компоненти обманних систем і містять приманки та/чи пастки;

3) $k_{kod,3}$ – комп'ютерні станції приманок та/чи пасток;

4) $k_{kod,4}$ – маршрутизатори;

5) $k_{kod,5}$ – сервери з компоненти обманних систем, але не містять приманок чи пасток;

6) $k_{kod,6}$ – сервери з компоненти обманних систем і містять приманки та/чи пастки.

Об'єкти корпоративних мереж $k_{kod,1}$-$k_{kod,6}$ можуть бути поділені на такі що перебувають в демілітаризованій зоні і такі, що не знаходяться в ній. А також можуть об'єкти поділені за територіальним принципом, тобто якщо вони знаходяться в різних

будівлях чи приміщеннях, включно на приватній території. Тоді, кількість таких об’єктів збільшується. До них також можуть відноситись IP-камери.

Побудуємо один з варіантів функції $F_{os,kod}$ (формула (3.3)). Нехай n_R – кількість об’єктів корпоративних мереж в R – сегменті, причому R також буде номером сегменту, а на координатній площині з рис. 2.10 буде значенням радіусу кола. Якщо в корпоративній мережі є один радіус кола, тоді він дорівнює одинці $R = 1$ і це буде означати відсутність поділу на сегменти, тобто наявність всіх вузлів корпоративної\мережі в одному сегменті. Якщо буде два сегменти, то кіл на координатній площині буде два з радіусами $R = 1$ та $R = 2$. Аналогічно, можна узагальнити таке представлення для довільної кількості сегментів в корпоративних мережах. Тоді, для такого випадку буде кількість кіл така як і кількість сегментів, а радіус кіл буде обиратись з послідовності чисел $R = 1, 2, 3, \ldots, k_R$, де k_R – кількість сегментів.

Розмістимо об’єкти корпоративних мереж одного сегменту на колі у вершинах правильного багатокутника, який вписано в це коло. Можливі варіанти вибору довільного багатокутника, але тоді потрібно задати функцію, яка встановлює відстані на колі між двома точками, і для всіх точок ця функція має коректно встановлювати координати. У випадку вибору правильного багатокутника всі сусідні точки на колі будуть рівновіддалені. Для знаходження координат n-кутника, який вписаний в коло з радіусом R та центром в точці O(0;0) координатної площини, задамо початкову вершину від якої будуть визначатись координати всіх решти вершин на колі. Для того, щоб при наявності двох і більше кіл початкові точки не були на одній півосі задамо координати початкових точок з врахуванням їх зміщення та в залежності від номерів кіл, які визначаються різними за довжиною радіусами. Координати початкових точок з урахуванням довжин радіусів на координатній площині:

$$\begin{gathered} x_{1,R_m} = R_m * \cos(m^\circ); \\ x_{2,R_m} = R_m * sin(m^\circ); \\ R_m = m;\ m\ \epsilon\ N;\ m = 1, 2, 3, \ldots, \end{gathered} \tag{3.5}$$

де m – натуральне число, що змінюється від одиниці до значення кількості сегментів корпоративної мережі; R_m – радіус кола з початком в центрі координат; $(x_{1,R_m};\ x_{2,R_m})$ – координати початкової точки багатокутника на координатній площині.

Кут зміщення кожної наступної початкової вершини багатокутника на кожному наступному колі збільшується на 1°.

Перевірку правильності заданих координат початкових точок багатокутників здійснимо обчисленням довжини радіуса між центром координатної площини і отриманими за формулою (3.5) координатами точок здійснимо згідно формули рівняння кола:

$$\left(x_{1,R_m}\right)^2 + \left(x_{2,R_m}\right)^2 = {R_m}^2, \tag{3.6}$$

тоді

$$(R_m * \cos(m^{\circ}))^2 + (R_m * sin(m^{\circ}))^2 = {R_m}^2((\cos(m^{\circ}))^2 + (sin(m^{\circ}))^2) = {R_m}^2.$$

Отже, координати початкових точок для всіх кіл задані правильно.

Перша початкова точка на колі для багатокутника буде знаходитись на осі абсцис і її значення $x_{1,R_1} = 1$, $x_{2,R_1} = 0$. Друга початкова точка - $x_{1,R_2} = 2 * \cos(2^{\circ})$, $x_{2,R_2} = 2 * sin(2^{\circ})$, третя початкова точка - $x_{1,R_3} = 3 * \cos(3^{\circ})$, $x_{2,R_3} = 3 * sin(3^{\circ})$.

Правильні багатокутники, які вписані в кола, будуть мати різну кількість вершин, але частина з них може мати однакову кількість вершин. При цьому вони будуть мати різні розміри, бо радіуси кіл для них різні, та їх початкові точки і решта відповідних за номерами точок не будуть належати відповідним прямим, які проходять через центр кола через зміщення координат початкових точок для кожного багатокутника.

Фактично згідно формул (2.31) та (3.5) позначення точок для визначення їх значень буде таким:

$$x_{kn,v,p,z,vect,R_{m,1}} = x_{1,R_m}; x_{kn,v,p,z,vect,R_m,2} = x_{2,R_m}. \tag{3.7}$$

Решту вершин з їх координатами отримуємо, обертаючи початкову вершину на кут, що дорівнює $\frac{360^{\circ}}{n_{R_m}}$, де n_{R_m} – кількість об’єктів в сегменті корпоративних мереж. Якщо в сегменті знаходиться один об’єкт, тоді він розміщується в початкову точку багатокутника відповідного кола. Координати решти точок на колах, які є вершинами n_{R_m}- кутників визначаємо так:

$$x_{1,R_m} = R_m * \cos\left(m^{\circ} + \left(n_{R_m} - 1\right) * \frac{360^{\circ}}{n_{R_m}}\right); \tag{3.8}$$

$$x_{2,R_m} = R_m * sin\left(m^\circ + \left(n_{R_m} - 1\right) * \frac{360^\circ}{n_{R_m}}\right);$$

$$R_m = m;\ m\ \epsilon\ N;\ m = 1, 2, 3, ...;\ n_{R_m}\ \epsilon\ N;\ n_{R_m} = 1, 2, 3, ...$$

Таким чином, підготовчий етап для імплементації алгоритму молі і полум'я полягає у подані об'єктів корпоративних мереж на координатній площині з поділом їх між сегментами. Таке подання є необхідним для забезпечення керування обманними системами приманками і пастками в контексті оптимізації частини кроків систем під час КА чи дій ЗПЗ та співвіднесення їх до місць впливів КА та дій ЗПЗ. Геометричну інтерпретацію такого подання, в якому використано формули (3.5) та (3.8), та прикладу потенційної спіралі зображено на рис. 3.2.

Аналогічно, для тримірного та багатомірного просторів можна також задати формулами координати точок в просторі, причому для тримірного простору об'єкти корпоративних мереж будуть розміщені на сферах, які будуть мати один центр і вкладатись одна в одну, а спіраль буде лінією в просторі, а не в площині.

Геометрична інтерпретація розміщення об'єктів корпоративних мереж в сегментах та потенційної спіралі на рис. 3.2 може бути модифікована таким чином, що кожне коло буде в окремій паралельній площині. Тоді, спіраль також буде розташована в тримірному просторі, а не в площині.

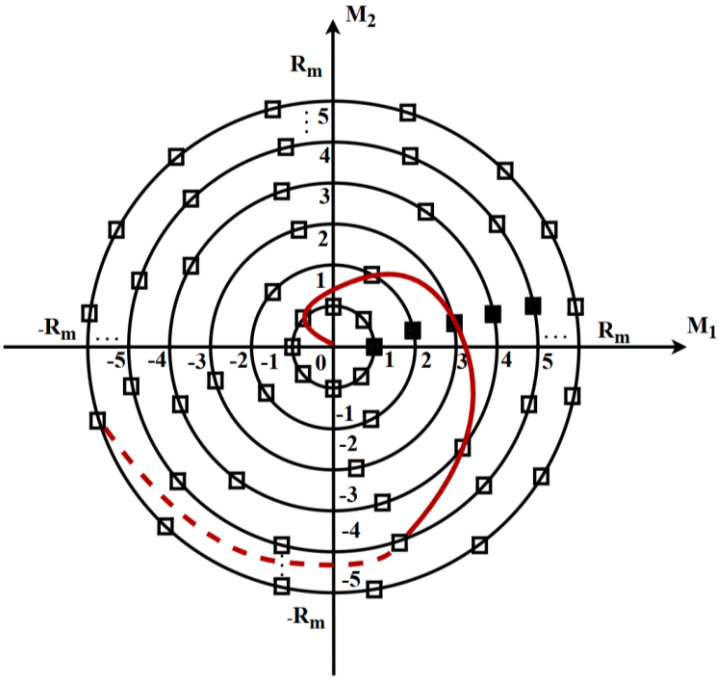


Рисунок 3.2 – Геометрична інтерпретація розміщення об'єктів корпоративних мереж в сегментах та потенційної спіралі

Функція для розміщення об’єктів корпоративних мереж є дискретною величиною, перелік наступних кроків обманних є дискретною величиною, тому будемо здійснювати дискретну оптимізацію алгоритмом молі і полум’я.

На наступному етапі, після підготовчого етапу, задамо об’єкти, на які здійснюються впливи КА та дій ЗПЗ, а також компоненти обманних систем та приманки і пастки з урахуванням рис. 3.2 та формул (2.31), (3.3)-(3.5), (3.7).

Розглянемо спочатку пряму схему, в якій в центрі системи координат знаходиться приманка та/або пастка, а в об’єктах корпоративних мереж, які розміщені на колах, відображено впливи КА чи дій ЗПЗ, причому ці впливи можуть бути не в усіх об’єктах корпоративних мереж. ЗПЗ може активним в певних вузлах корпоративних мереж і, тоді, такі вузли будемо вважати об’ктами, в яких наявні міль. Аналогічно, об’єкти на які спрямовані КА та на них відбуваються певні впливи теж вважатимемо такими, що в них може бути міль. А у випадку отримання контролю над об’єктами корпоративних мереж під час КА в них зловмисник втілює ЗПЗ для здійснення контролю над ними або здійснює знищення всього що в ньому наявне, то цей варіант будемо розглядати теж як наявність молі, що відповідатиме першим двом із наявним вже ЗПЗ та впливами КА. Таким чином, КА чи дії ЗПЗ, а також саме ЗПЗ, спрямовані чи наявні в об’єктах корпоративних мереж будемо позиціонувати як міль. Тобто, припускаємо, що кандидатами в контексті розв’язків є моль і її може бути більше однієї, а змінними задачі є положення молі у просторі. Тоді, моль, якої більше однієї, можуть літати в просторі будь-якого виміру зі зміною векторів свого положення. Таким об’єктам на координатній площині призначимо певний колір і задамо їх матрицею координат у певному вимірному просторі так:

$$A_M = \begin{pmatrix} a_{1,1} & \cdots & a_{1,N_{A_M,S}} \\ \vdots & \ddots & \vdots \\ a_{N_{A_{M,r}},1} & \cdots & a_{N_{A_{M,r}},N_{A_M,S}} \end{pmatrix}, \tag{3.9}$$

де $a_{i,j}$ – координата молі у певному вимірному просторі; $N_{A_M,S}$ – розмірність простору; $i = 1,2,3,\dots,N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об’єктів, в яких наявна міль; $j = 1,2,3,\dots,N_{A_M,S}$.

В поданні матрицею A_M сформовано популяцію метеликів молі в поточний момент часу. Через певний проміжок часу вона може змінитись кількісно. Для формул (3.5) та (3.8) і зображенню на рис. 3.2 значення $N_{A_M,S} = 2$.

Вплив, переміщення та перебування КА чи дій ЗПЗ в об’єктах корпоративних мереж є основою для визначення подальших кроків обманних систем. Серед цих об’єктів можуть бути і хибні об’єкти для атак, але керування ними здійснюють обманні системи. При цьому можна оцінити вплив, переміщення чи перебування КА чи дій ЗПЗ в об’єктах корпоративних мереж і для такого оцінювання введемо функцію допасованості, яка буде відображати близькість конструктивного рішення до поставленої цілі. Таке значення функції допасованості будемо визначати для кожної молі за її координатами, що задані за формулою (3.9) і тоді отримаємо таку матрицю:

$$F_1^{A_M} = \begin{pmatrix} F_{1,1}^{A_M} \\ F_{1,2}^{A_M} \\ \cdots \\ F_{1,N_{A_{M,r}}}^{A_M} \end{pmatrix}, \tag{3.10}$$

де $F_{1,j}^{A_M}$ – значення функції допасованості $F_1^{A_M}$ для j-тої молі; $N_{A_{M,r}}$ – кількість об’єктів, в яких наявна міль; $j = 1,2,3,\dots,N_{A_{M,s}}$.

Приманки і пастки можуть бути розміщені в просторі і визначатись координатами. В центрі координат може бути приманка чи пастка до якої прагне наблизитись моль, але приманки і пастки можуть бути не активними і перебувати на шляху молі. Задамо полум’я, що відповідатиме приманкам і пасткам, а також компонентам обманних систем, матрицею B_M аналогічно до матриці A_M так:

$$B_M = \begin{pmatrix} b_{1,1} & \cdots & b_{1,N_{A_{M,s}}} \\ \vdots & \ddots & \vdots \\ b_{N_{A_{M,r}},1} & \cdots & b_{N_{A_{M,r}},N_{A_{M,s}}} \end{pmatrix}, \tag{3.11}$$

де $b_{i,j}$ – координата полум’я у певному вимірному просторі; $N_{A_{M,s}}$ – розмірність простору; $i = 1,2,3,\dots,N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об’єктів, в яких наявне полум’я; $j = 1,2,3,\dots,N_{A_{M,s}}$.

Приймемо розмірності матриць A_M і B_M однаковими виходячи з того, що метелики молі є пошуковими агентами, які переміщаються в просторі пошуку, а полум’я є їх найкращою позицією в поточний момент часу. В контексті покращення функціонування обманних систем та дискретної оптимізації однакові розмірності матриць підходять для КА чи дій ЗПЗ, коли для протидії їм здійснюється підбір відповідних засобів і їх кількість залежить від виявлених зловмисних впливів, що є

кроками обманних систем, та в загальному підборі кроків з множини кроків систем. Тобто, полум'я можна розглядати орієнтиром для молі в просторі пошуку. Тоді, міль здійснює пошук навколо полум'я і оновлює його при знаходженні кращого рішення. Тому, внаслідок таких цілеспрямованих дій вона ніколи не втрачає свого найкращого рішення. Для КА чи дій ЗПЗ відбуваються спрямування на об'єкти, які є метою зловмисників. В цьому процесі наближення до мети покращує наступні варіанти. Зі сторони обманних систем теж наявне спрямування на опрацювання зловмисних впливів, яке досягається долученням нових приманок і пасток, що відображається певними кроками систем і, при цьому, вибір кроків здійснюється з уточненням стану в середовищах корпоративних мереж, тобто на кожному етапі обирається кращий варіант кроків. Введемо аналогічно до функції допасованості $F_1^{A_M}$ функцію допасованості $F_2^{B_M}$ для оцінювання варіантів з матриці B_M так:

$$F_2^{B_M} = \begin{pmatrix} F_{2,1}^{B_M} \\ F_{2,2}^{B_M} \\ \cdots \\ F_{2,N_{A_{M,r}}}^{B_M} \end{pmatrix}, \tag{3.12}$$

де $F_{2,j}^{B_M}$ – значення функції допасованості $F_1^{B_M}$ для j-того полум'я; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявне полум'я; $j = 1,2,3, \dots, N_{A_{M,s}}$.

Розглянемо кроки алгоритму дискретної оптимізації молі і полум'я, за яким здійснюється апроксимація глобального оптимуму. Сформуємо їх на основі стандартного подання, яке представлено в роботах [51, 53].

Задаємо три функції [51] формулами (3.13)-(3.15).

Функція F_1^A генерує випадкову популяцію молі, розміщує її у просторі, тобто формує матрицю A_M (формула (3.9)), та обчислює значення функції допасованості $F_1^{A_M}$ (формула (3.10)) для згенерованої популяції молі і задамо її так:

$$F_1^A: \emptyset \to \left\{A_M, F_1^{A_M}\right\}. \tag{3.13}$$

Функція F_2^A є основною функцією. Вона переміщує міль у просторі пошуку. Ця функція після отримання матриці A_M повертає її з оновленими значеннями елементів, тобто після її виконання знову отримуємо матрицю A_M. Задамо функцію F_2^A так:

$$F_2^A: A_M \to A_M. \tag{3.14}$$

Функція F_3^A повертає значення «істинне», якщо критерій завершення алгоритму задовольняється, і значення «хибне», якщо критерій завершення не задовольняється.

Задамо функцію F_3^A так:

$$F_3^A: A_M \to \{true, false\}. \tag{3.15}$$

Згідно введених функцій за формулами (3.13)-(3.15) визначимо алгоритм дискретної оптимізації молі і полум'я так:

$$\begin{aligned} &A_M = F_1^A(\,);\\ &\textit{поки } F_3^A(A_M) = \{false\}\\ &\qquad A_M = F_2^A(A_M);\\ &\textit{кінець} \end{aligned} \tag{3.16}$$

Функція F_1^A повинна генерувати початкові рішення та обчислювати значення цільової функції $F_1^{A_M}$, яку задано за формулою (3.10). У функції F_1^A можна використовувати будь-який випадковий розподіл. Задамо такий розподіл:

$$\begin{aligned} &\text{для } i = 1\,, n\,; \text{крок } 1\\ &\quad \text{для } i = 1\,, n\,; \text{крок } 1\\ &\qquad a_{i,j} = (c_i - d_i) * rand(\,) + d_i;\\ &\quad \textit{кінець}\\ &\textit{кінець}\\ &F_1^{A_M} = F_1^{A_M}(A_M); \end{aligned} \tag{3.17}$$

де $a_{i,j} \in A_M$; $a_{i,j}$ – координата молі у певному вимірному просторі; $N_{A_M,s}$ – розмірність простору; $i = 1,2,3, \dots, N_{A_M,r}$; $N_{A_M,r}$ – кількість об'єктів, в яких наявна міль; $j = 1,2,3, \dots, N_{A_M,s}$.

Для попереднього алгоритму матриці C та D, елементи яких визначають верхню і нижню межі зміни значень, задамо їх елементами так:

$$C = \left(c_1, c_2, \dots, c_{N_{A_{M,r}}}\right);\ D = \left(d_1, d_2, \dots, d_{N_{A_{M,r}}}\right), \tag{3.18}$$

де елементи матриці C є значеннями верхньої межі i-тої змінної; елементи матриці D є значеннями нижньої межі i-тої змінної; c_i – елемент матриці C; d_i – елемент матриці D; $i = 1,2,3, \dots, N_{A_M,r}$; $N_{A_M,r}$ – кількість об'єктів, в яких наявна міль.

Функція F_2^A виконується ітеративно і переміщує моль в просторі пошуку. Після повернення значення $\{true\}$ функцією F_3^A її виконання завершиться. Простір пошуку є дисретним і визначається матрицею векторів $M_{kn,v,p,z,vect}$ (формула (2.31) та матрицею A_M (формула (3.9). Він відображається на координатній площині на рис.2.10.

Алгоритм дискретної оптимізації молі і полум'я характеризується особливістю, яка полягає в дотриманні поперечної орієнтації при здійсненні вибору наступного положення молі щодо полум'я. Задамо такий рух так:

$$A_{M,i} = Q(A_{M,i}, B_{M,i}), \tag{3.19}$$

де $\bigcup_{i=1}^{N_{A_{M,r}}} A_{M,i} = A_M$; $A_{M,i}$ - i – та моль; $\bigcup_{i=1}^{N_{A_{M,r}}} B_{M,i} = B_M$; $A_{M,i}$ - i – та приманка чи пастка або компонента обманної системи; $i = 1,2,3, \dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль; Q – функція, яка задаватиме поперечну орієнтацію при здійснені вибору наступного положення молі щодо полум'я.

Функція Q буде визначати рух. Оскільки завдання з вибору кроків належить до дискретної оптимізації і побудоване пошукове поле теж є дискретним, то функція Q буде дискретною. Спіраль є основою схемою, згідно якої має відбуватись оновлення молі. Варіантів спіралей може бути багато і задаватись вони можуть різними варіантами функції Q. Але необхідне виконання таких умов при конструюванні варіантів функції Q:

1) рух повинен бути по спіралі;

2) рух повинен виконуватись винятково з дотриманням поперечної орієнтації;

3) початкова гранична точка спіралі повинна бути в одному з об'єктів корпоративних мереж, в якому встановлені потенційно зловмисні прояви, або в загальному підході – з точки чи кроку, в яких розпочато процес;

4) фінальною граничною точкою спіралі повинна бути точка з полум'ям;

5) відхилення спіралі не повинно виходити за межі пошукового простору, тобто спіраль повинна перебувати в межах крайніх точок, які максимально віддалені від центру (від точки з полум'ям);

6) спіраль не повинна вироджуватись у відрізок, що сполучає граничні точки по прямій лінії;

7) спіраль не повинна містити зворотні дуги і бути спрямована або за годинниковою стрілкою або проти годинниковою стрілки;

8) побудова спіралі повинна узгоджуватись з часом як параметром;

9) полум’я з часом може змінювати свої координати;

10) кожна наступна дуга між обраними для спіралі кроками повинна бути на тому ж колі, що й попередня дуга, або ближче до полум’я;

11) при оновленні місця полум’я оновлюються координати молі;

12) при наявності більше однієї молі та одного полум’я діє вимога, що моль, яка є першою серед всіх, повинна наближатись до полум’я раніше решти;

13) при наявності більше однієї молі та одного полум’я діє вимога, що моль, яка є останньою серед всіх, отримує розв’язок з решти, які залишились;

14) при наявності більше одного полум’я і більше однієї молі, але при їх однаковій кількості, рух кожної молі спрямовується до окремого полум’я і кількість полум’я зменшується з часом;

15) при наявності більше одного полум’я і більше однієї молі, але при їх різній кількості, рух кожної молі прагне до окремого полум’я, але за відсутності достатньої кількості полум’я розподіляється між ними, і кількість полум’я зменшується з часом.

Введемо в схему з рис. 3.2 третю вісь, яка буде відповідати за час. Геометрична інтерпретація розміщення об’єктів корпоративних мереж в сегментах та потенційної спіралі з урахуванням часу зображена на рис. 3.3. При досягненні часу, який відображає повне завершення дій обманних систем з обрання наступних кроків, зокрема і вибору приманок і пасток, полум’я вимикається. Інакше, полум’я змінює координати і процес продовжується в нових координатах, тобто система перебудовується. Таким чином, досягається уникнення збіжності в локальному оптимумі.

Перебудова місця полум’я та кількості полум’я буде впливати на перебудову місця молі та в результаті сприятиме досягненню кращого рішення. Спрямування кожної молі до окремого полум’я може бути відображено різними системами координат з центрами в точках кожного окремого полум’я.

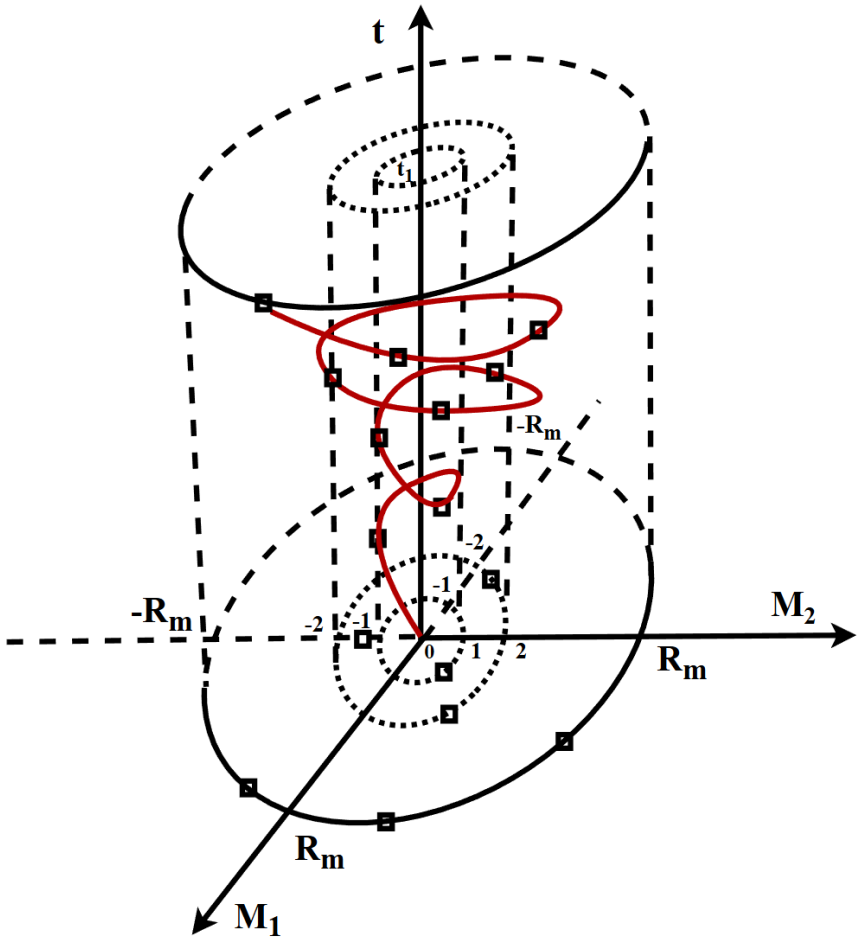


Рисунок 3.3 – Потенційна спіраль з урахуванням часу

На рис. 3.3 порівняно з рис. 3.2 введено вісь Ot, в якій відображено час. Це необхідно, бо час здійснення КА чи дій ЗПЗ повинен бути таким, що обманні системи мають цей параметр враховувати при виборі своїх кроків і підтриманні стійкості систем, переналаштуваннях їх, зміні наступних кроків протягом цього часу, щоб не завершити свої кроки достроково, до завершення КА чи дій ЗПЗ. Спіраль на рис. 3.3 спрямована до центру координат і вона відображає зменшення часу, тобто час КА чи дій ЗПЗ з урахуванням застосування приманок і пасток є збіжним до нуля, оскільки будь-яка КА чи дії ЗПЗ мають обмеження в часі, які можуть бути різними. КА чи дії ЗПЗ, наприклад, розпочинаються в час t_1, і припустимо, що впливи від них поміченні обманною системою, тоді вона починає вибір своїх кроків, зокрема і активізацію приманок у вузлах корпоративних мереж, в яких помічені потенційно зловмисні впливи. Активізація приманок першочергово необхідна для спрямування зловмисників на хибні об’єкти для атак в мережі. Для зацікавлення зловмисників в частині комп’ютерних станцій, де встановлено приманки і пастки обманна система

відкриває певні порти і розпочанає імітацію комуніквції для привертання уваги зловмисників. В результаті КА поширюються до нових вузлів корпоративних мереж. Частина комп'ютерних станцій напротязі часу здійснення КА чи дій ЗПЗ зловмисника блокується обманними системами або після проявів активності закривається. Обманна система таким чином спрямовує зловмисника до пастки, що знаходиться в центрі координат,здійснюючи зміну своїх кроків в процесі здійснення КА чи дій ЗПЗ зловмисником. При виникненні ускладнень для обманних систем в частині блокування КА чи дій ЗПЗ або їх виявленні, обманні системи змінюють місцезнаходження пастки, чим спонукають зловмисника перебудовувати вектор та площину КА. В новій системі координат обманні системи повторюють частину своїх дій, а частину оновлюють. Наприклад, вводять ще один центр координат, роздвоюючи та роззосереджуючи зловмисника на проведення двох КА.

Оскільки зловмисник згідно обмежень поставленої задачі здійснює двоцільову КА, тобто КА з метою виявлення і реальних і хибних об'єктів в корпоративних мережах, то його друга спроба після перебудови приманок і пасток в корпоративних мережах не може враховувати попередній крок, бо на ньому ідентифікація хибних об'єктів повністю не відбулась. Це пов'язано з тим, що обманна система оцінила тривалість КА чи дій ЗПЗ та здійснила перебудову, в результаті якої центр системи координат змістився на одне з кіл, в якому буде активізована приманка і пастка, а решта вузлів будуть заблоковані обманною системою. При цьому, попередній центр системи координат буде відображатись на колі, наприклад, того ж радіуса. Або обманна система розподілить об'єкти розміщені в одній системі координат на дві системи координат, сформувавши таким чином два полум'я.

Проєкції спіралі на площину M_1OM_2 на рис. 3.3 може відповідати зображення на рис. 3.2. Для конструювання функції Q згідно визначених умов введемо такі правила:

1) $\forall\, x_{l,i}\colon x_{l,i+1} \le x_{l,i}$; l – кількість координат; $i = 1,2,3,\dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль;

2)) $\forall\, x_{l,i}\colon x_{l,i+1} = x_{l,i}$ або $x_{l,i+1} < x_{l,i}$; l – кількість координат; $i = 1,2,3,\dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль;

3) a_1 (координати в матриці A_M, яку задано формулою (3.9)) міститься в одному з елементів матриці $M_{kn,v,p,z,vect}$ (формула (2.31)); $a_{i,j}$ – координата молі a_1 у певному

вимірному просторі; $N_{A_M,s}$ – розмірність простору; $i = 1,2,3, \dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль; $j = 1,2,3, \dots, N_{A_M,s}$;

4) $a_{N_{A_{M,r}}}$ (координати в матриці A_M, яку задано формулою (3.9)) міститься в одному з елементів матриці $M_{kn,v,p,z,vect}$ (формула (2.31)); $a_{i,j}$ – координата молі a_1 у певному вимірному просторі; $N_{A_M,s}$ – розмірність простору; $i = 1,2,3, \dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль; $j = 1,2,3, \dots, N_{A_M,s}$;

5) $\forall\, x_{l,i}$: $\left(x_{l,1,R_l}\right)^2 + \left(x_{l,2,R_l}\right)^2 \leq {R_{max}}^2$; l – кількість координат; $i = 1,2,3, \dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль; R_{max} – радіус найбільшого кола;

6) $\exists\, a_s$: $a_s\, \epsilon$ спіралі $\left[a_1; a_{N_{A_{M,r}}}\right]$, тобто між крайніми точками наявна мінімум одна точка;

7) якщо спіраль сформовано проти руху годинникової стрілки, тоді кут повороту спіралі $\varphi > 0$ та не може бути від'ємним; якщо навпаки – спіраль сформовано за рухом годинникової стрілки, тоді кут повороту спіралі $\varphi < 0$ та не може бути додатнім;

8) якщо t_1 – час, протягом якого обманна система підтримує дії та визначає кроки в межах однієї системи координат (рис. 3.3), тоді $t_0 = 0$ і $\lim\limits_{t=t_1,t_2,t_3,\dots,t_p,t_0} t = 0$ (p – номер передостанньої ітерації визначення часу), $t_1 > t_2 > t_3 > \dots > t_p > t_0$;

9) оновлення матриці B_M (формула (3.11)) з урахуванням вимоги п. 5;

10) $\forall\, x_{l,i}, x_{l+1,i}$: $\left(x_{l,1,R_s}\right)^2 + \left(x_{l,2,R_s}\right)^2 \leq {R_s}^2$; $\left(x_{l+1,1,R_{s-1}}\right)^2 + \left(x_{l+1,2,R_{s-1}}\right)^2 \leq {R_{s-1}}^2$; l – кількість координат; $i = 1,2,3, \dots, N_{A_{M,r}}$; $N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль; R_s – радіус більшого кола; R_{s-1} – радіус меншого кола; $[x_{l,i}; x_{l+1,i}]$ – дуга спіралі;

11) оновлення матриці A_M (формула (3.9)) з урахуванням вимоги п. 9;

12) якщо матриця A_M (формула (3.9)) більше одного рядка, тобто більше однієї молі, тоді прийнявши верхні рядки матриці за такі, що відповідають першим в послідовності молі, а нижні рядки – останнім, тобто послідовність молі сформовано за номерами $i = 1,2,3, \dots, N_{A_{M,r}}$ ($N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль) від

першої до останньої, то різниця відстаней з врахуванням координат з матриці A_M кожної молі і координатами центру буде формувати зростаючу послідовність;

13) якщо матриця A_M (формула (3.9)) більше одного рядка, то кількість розв'язків для послідовності молі зменшується від першої до останньої, які задано за номерами $i = 1,2,3, \dots, N_{A_{M,r}}$ ($N_{A_{M,r}}$ – кількість об'єктів, в яких наявна міль) від першої до останньої;

14) оновлення матриці B_M (формула (3.11)) з поділом її на дві матриці і більше;

15) оновлення матриці B_M (формула (3.11)) з поділом її на дві матриці і більше та оновлення матриці A_M (формула (3.9)).

Введені правила 1-15 відповідають алгоритму дискретної оптимізації молі і полум'я та дають змогу сформувати варіанти кроків, або вузлів, зокрема і з приманками і пастками, в які може переміститись міль в процесі руху по спіралі. Таких кроків може бути декілька. Задамо їх формально (елементи перепозначено) множиною $M_k = \left\{m_1, m_2, \dots, m_{N_{M_k}}\right\}$, в якій N_{M_k} – кількість кроків, $q = 1,2, \dots, N_{M_k}$. Ця множина міститься в локальному секторі, якщо відобразити геометричну інтерпретацію виконання правил 1-15 щодо варіантів кроків, які можуть бути наступними серед усіх кроків, які наявні. На рис. 3.4 зображено елементи множини M_k.

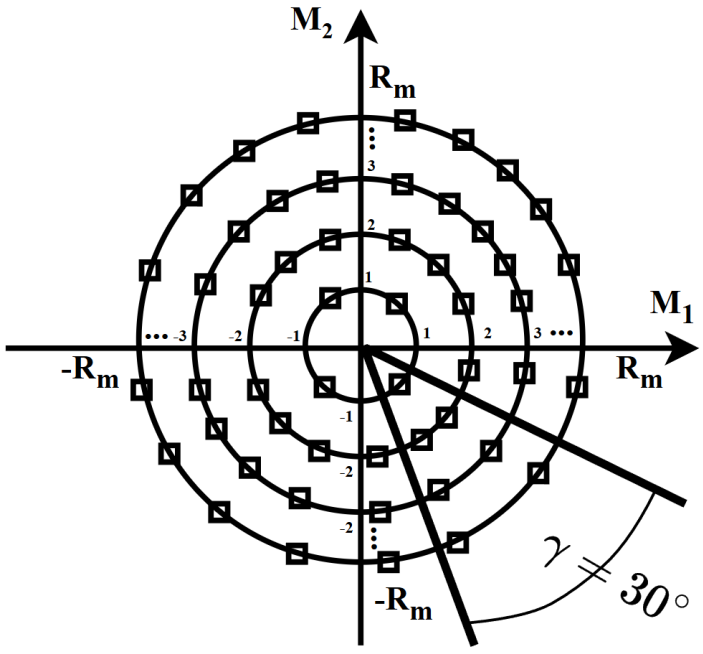


Рисунок 3.4 – Сектор розташування елементів множини M_k

Враховуючи правила 1-15 задамо функцію Q так:

$$Q(a_k) = \begin{cases} \varphi = \varphi + \theta, \text{якщо } M_k = \emptyset; \text{інакше} \\ m_s, \text{якщо } M_k \neq \emptyset \text{ та } \min\limits_{M_k} F_{os}\left(v_{os,i}, v_{os,i,u}\right), \end{cases} \quad (3.20)$$

де $m_s \in M_k$; F_{os} – цільова функція оцінювання обраних варіантів (формула (3.2)); $v_{os,i,u,j}$ – крок обманної системи з приманками і пастками; $j = 1,2, \dots, N_{os,i,u}$; $N_{os,i,u}$ – кількість кроків ОСПП для вирішення певного завдання, які визначено системою в поточний момент часу; i – i-та послідовність кроків для вирішення певного завдання системи; φ - кут повороту спіралі від крайнього значення кута; θ – значення додаткового кута для випадку відсутності варіантів для вибору в заданому сегменті; u – уточнення кроків, які задано вектором $v_{os,i}$ (формула (2.34)).

Всі елементи множини M_k повинні бути в секторі, який задано кутом φ. Кут φ визначається від променя, який розпочинається в початку координат і проходить через поточну точку, в якій знаходиться міль, що планує переміститись. Кут φ може бути різним. Але чим він менше, тоді спіраль швидко спрямовується до центру координат і необхідна далі буде зміна початку координат. Якщо ж кут буде більше 180^0, наприклад, тоді спіраль буде пологіша і наближення до центру координа буде повільнішим. Може бути так, що в наступному сегменті жодного елементу з множини M_k немає, тоді збільшуємо кут на величину θ та повторюємо дії.

Якщо в секторі наявні елементи множини M_k, то обираємо той з них, для якого цільова функція F_{os} оцінювання обраних варіантів (формула (3.2)) буде мати найменше, тобто найгірше, значення, що забезпечує продовження вибору варіантів в просторі пошуку з плавним наближенням до центру координат в наступному секторі.

Основні кроки методу синтезу популяційних алгоритмів в архітектурі ОСПП:

1) підготовка даних та їх кодування для формування дискретного простору пошуку (формули (3.1)-(3.8), рис. 3.2);

2) формування матриць молі та полум’я і відповідні їм функції допасованості (формули (3.9)-(3.12));

3) введення величини час до дискретного простору пошуку(рис. 3.3);

4) задамо три функції (формули (3.13)-(3.15)) для реалізації алгоритму дискретної оптимізації молі і полум’я;

5) згідно введених функцій з п. 4 за формулами (3.16)-(3.18) виконуємо кроки алгоритму дискретної оптимізації молі і полум’я;

6) виконуємо правила 1-15 та знаходимо значення функції Q (формула (3.20)) для формування варіантів кроків згідно заданого кута для сектора із значеннями;

7) задаємо та відображаємо рух по спіралі за формулою (3.19).

Основні сім кроків методу не є чітко детерміновані в контексті послідовного виконання. Частина з кроків, крім кроку 1, можуть виконуватись паралельно або багатократно, тобто будуть викликатись з певних інших кроків.

Крок 5 виконується багаторазово та реалізує популяційний алгоритм натхненний природою, тобто алгоритм молі і полум'я, в частині пошуку розв'язків. Але алгоритм спрямований на забезпечення дискретної оптимізації, тому на кроці 6 конструюється функція на множині елементів дискретного простору пошуку. І, згідно неї вибираються наступні кроки.

Таким чином, здійснено синтез алгоритму комбінаторної оптимізації молі і полум'я в архітектурі ОСПП в корпоративних мережах, особливістю якого є формування дискретного простору пошуку з поданням об'єктів координатами у вимірних просторах, синтез спіралі в дискретному просторі пошуку на основі оцінювання потенційних варіантів наступних кроків та множини елементів у визначеному секторі, врахування часу в контексті вибору кроків, забезпечення зміни розташування полум'я та молі, що дало змогу забезпечити вибір та зміну кроків для функціонування ОСПП протягом тривалого часу під час КА чи дій ЗПЗ для заплутування зловмисників.

Слід спіралі поєднує об'єкти в дискретному просторі пошуку. Він відображає рух молі до полум'я, що відповідає саме алгоритму молі і полум'я, та ним задано необхідні кроки із формування сліду спіралі та зміни позицій молі і полум'я. Також в алгоритмі враховано час як параметр, що дає змогу фіксувати побудову сліду спіралі і, відповідно, зміну кроків системою. Досліджено формування сліду спіралі на основі величини кута для сектору, в якому знаходяться наступні кроки состем, та визначено правила для формування сегментів спіралі. Для уникнення швидкої збіжності в локальному оптимумі в алгоритмі дискретної оптимізації молі і полум'я задані кроки з оновлення розміщення полум'я і молі, що дає змогу перебувати молі тривалий час навколо полум'я і обирати варіанти кроків з глобального оптимуму. Поперечна орієнтація, як ознака цього алгоритму натхненному природою, забезпечена рухом молі по дотичній до одного кола або перехід її в нове положення в колі меншого

радіуса за попереднє коло. При наявності більше однієї молі перша моль обирає кращі варіанти, а наступні моліобирають з тих, варіантів, які залишаються. Таким чином, остання моль обирає за залишковим принципом. Це дає змогу в подальшому сортувати їх за ступенем допасованості і оцінювати доцільність переміщення полум'я в дискретному просторі пошуку.

Розроблений метод синтезу популяційних алгоритмів в архітектурі ОСПП дає змогу створювати обманні системи для заплутування зловмисників в корпоративних мережах.

3.2 Метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах

Функціонування ОСПП в корпоративних мережах потребує забезпечення взаємодії їх розподілених компонентів, підтримування тривалого часу функціонування в змінюваному середовищі, опрацюванні штатних і позаштатних подій в оточуючому середовищі та при взаємодії між компонентами систем, виконання визначених функцій з виявлення КА та ЗПЗ, реалізації обманних дій та керування приманками і пастками. В поєднанні ці основні функції обманних систем повинні забезпечити організацію функціонування ОСПП протягом тривалого часу та в умовах впливів КА та дій ЗПЗ.

Комп'ютерні станції в корпоративних мережах можуть бути увімкнені в різний час, а сервери можуть працювати цілодобово. Якщо виокремити приманки і пастки, то вони можуть бути активними в увімкнених комп'ютерних станціях, які цілеспрямовано не вимикаються для їх підтримки в цілодобовому режимі. Також вони можуть вимикатись разом з рештою комп'ютерних станцій. Тому, виникає проблема щодо можливості класифікації реальних та хибних об'єктів в корпоративних мережах через різні добові графіки увімкнень серверів, комп'ютерних станцій тощо, а також через атаки, які здійснюються з різних часових поясів. Для уникнення таких дій, обманні системи повинні функціонувати в корпоративних мережах цілодобово із залученням додаткових ресурсів, зокрема додатково до необхідних функціонуючих серверів повинні бути долучені комп'ютерні станції з елементами та компонентами ОСПП. А також, приймати рішення на основі

популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп'ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах. Обманні системи повинні підтримувати імітацію функціонування корпоративних мереж у неробочий для користувачів час. Таким чином, застосування ОСПП повинно здійснювати цілодобово. Долучення нових комп'ютерних станцій або вилучення наявних в корпоративні мережі може бути здійснено без вимкнення всіх вузлів. Долучення нових комп'ютерних станцій в корпоративні мережі повинно супроводжуватись встановлення компонентів та елементів обманних систем, включно з встановленням приманок і пасток за потреби. При вилученні комп'ютерних станцій чи серверів з корпоративних мереж повинні бути скориговані відомості про вузли та їх типи в базах обманних систем. Всі такі дії впливають на оцінювання рівнів безпеки вузлів та всієї мережі в цілому. При зміні вузлів в корпоративних мережах та, відповідно, і архітектури ОСПП системи оновлюють відомості про стани та параметри оточуючого середовища та внутрішні процеси. Також, для підтримуванні цілісності розподілених компонентів обманних систем в них здійснюються постійні обміни повідомленнями щодо поточних станів і подій, які стосуються винятково їх.

Обманні системи повинні здійснювати опрацювання штатних і позаштатних подій в оточуючому середовищі та при взаємодії між компонентами систем, які стосуються їх функціонування, без залучення адміністраторів. Якщо ж все-таки залучення адміністраторів необхідне, тоді обманні системи повинні локалізувати вузли корпоративних мереж і надати повідомлення адміністраторам. Якщо ж обманні системи зіткнулись з внутрішніми проблемами свого функціонування і не змогли вирішити їх, тобто не змогли опрацювати позаштатні події, тоді теж залучають адміністраторів, блокуючи при цьому більшість або всі вузли, в які вони встановлені їх компоненти та елементи.

Для виконання визначених функцій з виявлення КА та ЗПЗ обманні системи повинні мати набір ознак для розпізнавання впливів КА чи дій ЗПЗ. Кількість функцій для опрацювання таких подій не є обмеженою і в систему можна їх доповнювати. Нові функції можуть бути доповнені через долучення нових елементів чи компонентів в комп'ютерні станції або з долученням нових комп'ютерних станцій.

Окремі функції можуть бути розподілені між компонентами обманних систем або бути включеними до складу приманок і пасток.

ОСПП окремо від приманок і пасток повинні мати набір обманних дій, які вони повинні застосовувати для заплутування зловмисників при проведенні КА чи дій ЗПЗ. Також, в цей процес повинні бути долучені приманки і пастки. Все ці дії повинні бути синхронізовані в корпоративних мережах для досягнення результативності з виявлення КА чи дій ЗПЗ.

Під час реалізації обманних дій можуть залучатись приманки і пастки, але можуть і не залучатись. Приманки і пастки можуть бути залучені обманними системами для опрацювання окремих дій в окремих вузлах корпоративних мереж.

Реалізація обманних дій, крім застосування самих обманних технологій, в обманних системах базується на поведінці, яка сформована згідно алгоритму дискретної оптимізації молі і полум'я. При його застосуванні, наприклад, щодо керування приманками і пастками можливим варіантом дії є активізація або переведення в пасивний стан приманок і пасток. Також, частина вузлів корпоративних мереж може бути вимкнена, а певна інша частина може імітувати інтенсивну взаємодію для привертання уваги зловмисників. Крім обманних дій, якими повинні бути наповнені обманні системи, вони повинні самі синтезувати нові дії з використанням базових дій та оцінювати їх ефективність при здійсненні протидії зловмисним впливам, що викликані КА чи діями ЗПЗ. Формування таких нових синтезованих дій може бути здійснено згідно кроків алгоритму молі і полум'я, а також можуть бути реалізовані в архітектурі обманних систем інші популяційні алгоритми або декілька з подальшим вибором одного з ним в певних станах систем.

Задамо основні кроки методу організації функціонування ОСПП в корпоративних мережах.

1. Формування ОСПП в корпоративних мережах та забезпечення взаємодії їх розподілених компонентів (формули (2.6)-(2.9), (2.19)-(2.27)).

2. Опрацюванні штатних і позаштатних подій в оточуючому середовищі та при взаємодії між компонентами систем (формули (2.10)-(2.18)) та перехід до кроку 3.

3. Виконання визначених функцій з виявлення та протидії КА та ЗПЗ (формули (2.1)-(2.5)).

4. Вибір наступних кроків обманних систем (формули (2.28)-(2.34)) згідно методу синтезу популяційних алгоритмів в архітектурі обманних систем (формули (3.13)-(3.20)).

5. Виконання обманних дій та керування приманками і пастками згідно рішень, які визначено на кроці 4.

6. Формування та доповнення ОСПП окремо від приманок і пасток набором обманних дій.

7. Формування та доповнення обманних систем приманками і пастками.

Ці кроки не є послідовними. Вся система є розподіленою, тому вони можуть виконуватись паралельно в різних її частинах.

Особливість методу організації функціонування ОСПП в корпоративних мережах в тому, що для уникнення реалізації двоцільових атак зловмисниками, суть яких у класифікації об’єктів корпоративних мереж на реальні та хибні, функціонування приманок і пасток в складі обманних систем організовано таким чином, що протягом часу функціонування ОСПП зловмисникам ускладнено такі дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп’ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах. Це, зокрема, дає змогу ускладнити зловмиснику розуміння реальних та хибних об’єктів в корпоративних мережах.

Реалізація популяційних алгоритмів в архітектурі обманних систем, зокрема алгоритму молі і полум’я, при виборі ними наступних кроків дає змогу
уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також враховувати потенційну спроможність зловмисників до здійснення двоцільових КА.

3.3 Висновки до третього розділу

Здійснено синтез алгоритму комбінаторної оптимізації «моль–полум’я» в обманних системах із приманками й пастками в корпоративних мережах. Особливістю підходу є формування дискретного простору пошуку з координатним

поданням об’єктів, побудова спіралі на основі оцінки можливих кроків у вибраному секторі, а також урахування часу для коректного вибору та зміни кроків. Це забезпечує динамічне переміщення молі й полум’я та дозволяє системі довго підтримувати роботу у сценаріях КА чи ЗПЗ, ускладнюючи дії зловмисників.

Слід спіралі відображає рух молі до полум’я та задає послідовність кроків і зміну їх позицій. Час виступає параметром фіксації побудови спіралі та переходів системи. Досліджено формування сегментів спіралі через визначення кута сектору пошуку. Для уникнення передчасної збіжності алгоритм передбачає оновлення позицій молі й полум’я, що дозволяє молі довше перебувати біля полум’я й досліджувати глобальні варіанти. Поперечна орієнтація руху забезпечується переходом молі між колами різного радіуса. За наявності кількох молей перша обирає найкращий варіант, інші обирають з доступних залишкових варіантів, що дозволяє впорядковувати їх за припасованістю та визначати наступне переміщення полум’я в дискретному просторі пошуку.

Розроблено метод організації функціонування ОСПП в корпоративних мережах, суть якого в тому, що для уникнення реалізації двоцільових атак зловмисниками функціонування приманок і пасток в складі обманних систем організовано таким чином, що протягом часу функціонування ОСПП зловмисникам ускладнено такі дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп’ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах. Реалізація алгоритму молі і полум’я в архітектурі обманних систем при виборі ними наступних кроків дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також враховувати потенційну спроможність зловмисників до здійснення двоцільових КА.

Основні наукові результати розділу опубліковані в працях [103; 134; 135; 169; 171]

РОЗДІЛ 4
МЕТОД ВИЯВЛЕННЯ АТАК ЗГІДНО АНАЛІЗУ СТАТИСТИЧНИХ ПОКАЗНИКІВ МЕРЕЖНОГО ТРАФІКУ В КОРПОРАТИВНИХ МЕРЕЖАХ, РЕЗУЛЬТАТИ ЕКСПЕРИМЕНТІВ ТА ОЦІНЮВАННЯ ОБМАННИХ СИСТЕМ З ПРИМАНКАМИ І ПАСТКАМИ

4.1 Метод виявлення комп'ютерних атак з використанням приманок і пасток згідно аналізу статистичних показників мережного трафіку

Розглянемо імплементацію в архітектуру ОСПП підсистеми виявлення КА. Для її реалізації будемо враховувати процес збору статистичних показників мережного трафіку приманками і пастками з подальшим опрацюванням зібраних даних. Виберемо тип КА спрямований на відмову в обслуговуванні у корпоративних мережах.

Основною метою проєктованої підсисетми виявлення атак відмова в обслуговуванні у корпоративних мережах є створення стійкої системи, яка буде адаптована до змін у більшості характеристик трафіку. Основним критерієм для групування, унікальним для кожного хоста в мережі, є його IP-адреса. З точки зору групування трафіку за IP-адресою такий підхід має ключову перевагу в тому, що це дозволяє ефективно аналізувати загальну комунікацію хоста, незалежно від змін у деталях пакета, таких як порти або налаштування маршрутизації. Це забезпечує стійкість системи до варіацій у трафіку, дозволяючи швидше виявляти аномалії, зокрема DDoS-атаки, навіть за умови високої мінливості параметрів потоку. Такий підхід полегшує завдання моделі машинного навчання, оскільки класифікація ґрунтується на спільному IP-ідентифікаторі хоста, що знижує ризик помилок у групуванні та дозволяє виявляти атаки в режимі реального часу. Таким чином, не ставиться за мету розрізняти легітимні потоки та зловмисні DDoS-потоки, а натомість проводити аналіз загальної комунікації клієнта.

Методи машинного навчання, що працюють із потоками, розглядають потік у цілому – від його початку і до завершення, оскільки лише потоки, позначені як завершені, експортуються із модуля-видобування до модуля опрацювання. Однак цей принцип не застосовується до статистики, яка зібрана за IP-адресою, оскільки у

даному випадку не можна точно визначити, чи надсилатиме IP-адреса більше даних чи ні. Крім того, очікування, поки хост припинить зв’язок, і класифікація його даних після цього є контрпродуктивним, оскільки завдання стоїть якнайшвидше виявити поточну атаку та зменшити чи зупинити зловмисний трафік. Тому, пропонована підсистема виявлення DDoS атак у мережах повинна опрацьовувати потік даних у режимі реального часу та формувати відповідну статистику, щоб модель машинного навчання могла прийняти відповідне рішення. Таким чином, вхідними даними для методу машинного навчання є набір даних на основі IP-адреси клієнта, і тому очікуваний результат методу буде залежати від створення конкретною IP-адресою зловмисного чи нормального трафіку.

З точки зору запропонованої архітектури підсистеми виявлення DDoS-атак проблема підробки IP-адрес і наявність NAT можуть створювати певні труднощі у точності класифікації та відокремленні зловмисного трафіку від нормального. Однак ці недоліки можна частково обійти. Використання зворотної фільтрації, рекомендованої в RFC 2827 допомагає зменшити обсяг підробленого трафіку на рівні провайдера. Хоча це не універсальне рішення і потребує підтримки з боку більшості інтернет-провайдерів, але воно може знизити ризик. Незважаючи на об'єднання декількох потоків трафіку під однією IP-адресою, високий обсяг зловмисного трафіку, наприклад, породжений великими DDoS-атаками, часто закриває чи ховає нормальний трафік, що дозволяє виявляти такі атаки з використанням статистичних методів і машинного навчання. Для атак з невеликим обсягом точність виявлення може знижуватися, але з урахуванням додаткових ознак трафіку або більш глибокого аналізу поведінки можна посилити ефективність підсистеми.

Таким чином, ці проблеми є реальними, але вони можуть бути пом'якшені відповідними механізмами та налаштуваннями.

Підсистема виявлення атак відмова в обслуговуванні у корпоративних мережах призначена для виявлення та запобігання зловмисній активності в комп’ютерних мережах за допомогою методів машинного навчання. В ній повинен бути здійснений аналіз мережного трафіку для виявлення потенційних атак відмова в обслуговуванні та оперативне реагування шляхом блокування зловмисного трафіку.

Трафік буде аналізуватись у межах визначених часових вікон, що дозволяє виявляти зміни в активності мережі в реальному часі. Він розглядається як

послідовність пакетів, для яких обчислюється статистика двох типів: усередині кожного часового вікна; між вікнами. Внутрішньо-віконна статистика відстежує параметри активності для кожної IP-адреси протягом певного інтервалу часу, а між-віконна статистика дозволяє аналізувати зміни в поведінці трафіку між вікнами. Такий підхід дає змогу моделі машинного навчання виявляти аномалії, характерні для атак, і своєчасно блокувати зловмисний трафік. У випадку виявлення загрози атаки відмова в обслуговуванні генерується правило для блокування пакетів із визначеною IP-адресою. Це правило зберігається в базі даних для швидкого пошуку механізмом обробки пакетів. Під час опрацювання пакетів обманна система перевіряє доцільність блокування чи передавання конкретного пакету. Цим вона забезпечує активний захист мережі та оперативне реагування на виявлені загрози.

Виділимо в архітектурі підсистеми виявлення атак відмова в обслуговуванні складається чотири модулі (рис. 4.1): модуль вилучення ознак; модуль збору статистики; модуль менеджера машинного навчання; модуль реагування. Модуль вилучення ознак опрацьовує пакети даних і отримує із них відповідні ознаки. Модуль збору статистики призначений для зберігання отриманого набору ознак та обчислення узагальнюючих статистичних показників на їх основі. Цей модуль також виконує експорт статистики у формат, що придатний для опрацювання за допомогою моделі машинного навчання. Модуль менеджера машинного навчання відповідає за попереднє опрацювання обчислених статистичних даних і керування процесом машинного навчання, включаючи створення, навчання та оцінювання моделей. Цей підхід дозволяє системі адаптуватися до нових загроз і покращувати точність виявлення зловмисної діяльності з часом. Модуль реагування призначений для збереження до бази даних інформації про перевірені IP-адреси та формування правил, що дозволять блокувати зловмисний IP-трафік.

З метою виявлення потенційних загроз і атак відмова в обслуговувані необхідно розробити метод, особливістю якого є опрацювання обчислених статистичних показників мережного трафіку та залучення механізмів машинного навчання, для імплементації його в ОСПП.

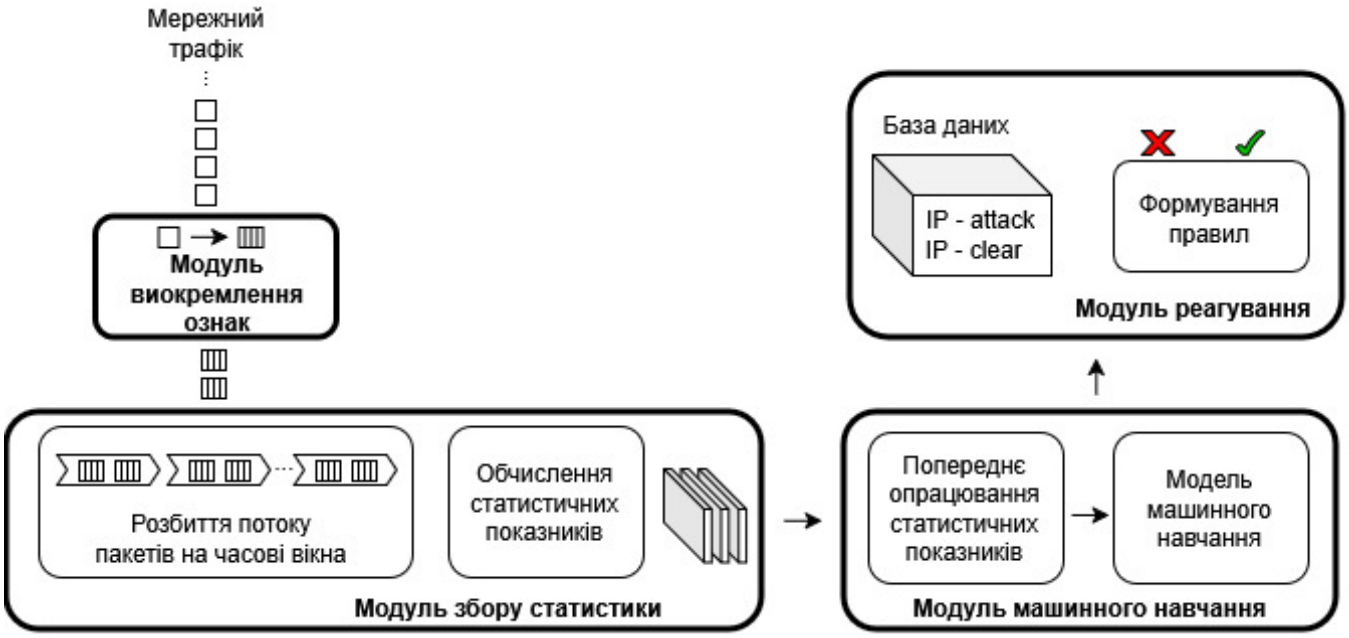


Рисунок 4.1 – Архітектура підсистеми виявлення атак відмова в обслуговуванні

Задамо покроковий процес реалізації методу. Він базується на періодичному отриманні даних із мережного IP-трафіку, отриманні із нього статистики з подальшим збереженням цієї інформації у внутрішніх структурах. Коли для певної IP-адреси накопичується достатня кількість даних, ці статистичні показники передаються до моделі машинного навчання, яка генерує прогноз щодо можливих загроз. Якщо модель виявляє зловмисну діяльність для даної IP-адреси, система відбирає деякі з цих даних і передає їх до механізму формування правил. Розглянемо кроки методу виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників.

Початковими даними для опрацювання на певних кроках методу будуть такі:

1) множина мережних пакетів $P = \{p_i\}_{i=1}^{N}$;

2) множина параметрів часового вікна $W = \{w_1, w_2, w_3, w_4\}$, де w_1 – час існування статистики (history_timeout); w_2 – тривалість вікна (window_length); w_3 – мінімальна кількість пакетів, які має надіслати хост в межах w_2 (history_min); w_4 – мінімальна кількість пакетів, для яких обчислюється статистика (packets_min);

3) кількість часових вікон K_W;

4) вектор статистичних показників мережного IP трафіку для IP адреси $F = \{f_i\}_{i=1}^{25}$, причому $F = F_{inside,IP_{dest}} \cup F_{outside}$, де $F_{inside,IP_{dest}}$ – узагальнені статистичні

показники всередині вікон K_W для IP адреси IP_{dest}, $F_{outside}$ – статистичні показники між часовими вікнами [1, K_W].

Результатом виконання кроків методу будуть множини дозволених $A = \{a_i\}_{i=1}^{N_A}$ та заблокованих $D = \{d_i\}_{i=1}^{N_D}$ IP адрес.

Задамо метод такими основними кроками.

1. Початкова ініціалізація часового вікна W_i, $W_i \leq K_W$.

2. Для кожного мережного IP-пакета здійснення відстеження без збереження до пам'яті його метаданих $M = \{m_1, m_2, m_3, m_4, m_5, m_6, m_7, m_8, m_9\}$, де m_1 – мітка часу, m_2 – IP адреса джерела, m_3 – IP адреса призначення, m_4 – протокол четвертого рівня, m_5 – порт джерела, m_6 – порт призначення, m_7 – довжина заголовка, m_8 – довжина корисного навантаження, m_9 – TTL період.

3. Групування пакетів за IP-адресою та оновлення внутрішньо віконної статистики. Для кожного пакету p_j в рамках кожного вікна W_i виконувати:

3.1) групування за IP-адресою призначення m_3: усі пакети, що мають однакову IP-адресу призначення m_3, об'єднуються в одну групу; для кожної IP-адреси призначення створюється та ініціалізується початковими значеннями частина вектору статистичних показників мережного IP-трафіку $F_{inside,W_i,IP_{dest}}$, які представляють узагальнену статистику всередині вікна;

3.2) оновлення статистичних показників вектору $F_{inside,W_i,IP_{dest}}$ цієї IP-адреси; для кожного елементу f_i із вектору $F_{inside,W_i,IP_{dest}}$ $f_i \in F$ і $f_i \in F_{inside,W_i,IP_{dest}}$ виконувати обчислення статистичних метрик (операції мінімуму, максимуму, середнього значення, стандартного відхилення, кількості пакетів) на основі значень елементів із множини M; для кожної IP-адреси призначення IP_{dest} в межах W_i, отримуємо оновлений вектор статистичних показників згідно формула так:

$$IP_{dest} \rightarrow F_{inside,W_i,IP_{dest}}. \quad (4.1)$$

4. Перевірка умов завершення часового вікна. Якщо час існування W_i перевищує w_1 або кількість пакетів у вікні досягла значення w_3, то вікно вважається завершеним. Якщо вікно завершене, то статистичні показники для кожної IP-адреси у вигляді вектору $F_{inside,W_i,IP_{dest}}$ зберігаються у базі даних для подальшого аналізу.

5. Якщо $W_i \leq K_W$, то перейти до кроку 1.

6. Агрегацію даних всіх векторів $F_{inside,W_i,IP_{dest}}$ для кожної IP адреси IP_{dest} здійснити так:

$$F_{inside,IP_{dest}} = aggregate(F_{inside,W_i,IP_{dest}} | i = 1, \dots, K_W), \quad (4.2)$$

де $aggregate$ – функція агрегації.

Результатом кроку є список векторів $F_{inside,IP_{dest}}$ для кожної IP-адреси, яка зустрічалась у часових вікнах $[1, .., K_W]$.

7. Обчислення статистичних показників між часовими вікнами та формування вектору $F_{outside}$.

8. Формування для кожної IP-адреси IP_{dest} вектора статистичних показників, як об'єднання $F_{inside,IP_{dest}}$ та $F_{outside}$ так:

$$F = F_{inside,IP_{dest}} \cup F_{outside}. \quad (4.3)$$

Результатом даного етапу буде список векторів $F_{list,IP_{dest}}$ для кожної IP-адреси із врахуванням статистичних показників як всередині часових вікон K_W для IP-адреси так і статистичних показників між цими вікнами і задамо їх так:

$$F_{list,IP_{dest}} = \{F_{inside,IP_{dest}} | i = 1, \dots, K_W\} \cup \{F_{outside}\}. \quad (4.4)$$

9. Нормалізація значень статистичних ознак для кожного вектора із $F_{list,IP_{dest}}$.

10. Класифікація IP-адрес, що представлені векторами $F_{list,IP_{dest}}$ із залученням моделі машинного навчання M, яку задамо так:

$$y_{IP} = \mathrm{M}(F_{list,IP_{dest},i}). \quad (4.5)$$

Якщо y_{IP}=”attack”, то додати до множини заблокованих IP-адрес $D = D \cup y_{IP}$ та сформувати правило для заборони вхідного трафіку від даної IP-адреси, наприклад, так:

iptables -A INPUT -s <IP> -j DROP

Якщо y_{IP}=”normal”, то додати до множини дозволених IP адрес, наприклад, $A = A \cup y_{IP}$.

Розглянемо детальніше основні концепції методу виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників. Основними вхідними даними для підсистеми виявлення атак відмова в обслуговуванні у корпоративних

мережах є мережний трафік. Тому, важливим етапом є процес отримання релевантних ознак із вхідних даних та їх перетворенні в форму, яка придатна для використання в моделі машинного навчання. Це включає визначення того, які аспекти мережних пакетів є найбільш інформативними для виявлення зловмисної діяльності, зокрема, заголовки пакетів, розмір пакетів, час між пакетами та інші метрики. З метою досягнення максимального універсального використання, запропонована система орієнтується на використання метаданих трафіку замість вмісту самих пакетів. Метадані використовуються для обчислення набору статистичних ознак. Отримані з них ознаки обираються на основі потреб у обчислених статистиках і їх зручності для отримання, що може бути реалізовано на апаратному рівні з мінімальними додатковими витратами.

До основних отриманих ознак, що отримуються із кожного мережного пакету належать мітка часу надходження пакета, яка дозволяє обчислювати різні статистики часу. Для визначення взаємодіючих хостів, також, отримуються IP-адреси джерела та призначення. Крім того, деякі статистики вимагають використання номерів портів 4-го рівня, тому також виконується отримання протоколу транспортного рівня і номерів портів. Повний список отриманих ознак із кожного пакета представлено у табл. 3.1.

Мітки часу витягуються у вигляді 64-бітних невід'ємних цілих чисел, які представляють кількість наносекунд. IP-адреси витягуються у вигляді рядків, щоб спростити їх маніпуляцію без додаткових перетворень пізніше в потоці опрацювання. Однак можна, також, використовувати байтове представлення, якщо буде потрібно оптимізувати використання пам’яті екстрактора. Протокол 4-го рівня представлений його номером, призначеним в IANA, ICMP для IPv4 та ICMP для IPv6. Вони обидва представляються одним значенням «1» для спрощення обчислення. Номери портів беруться у вигляді цілих чисел, оскільки вони безпосередньо отримуються з заголовків 4-го рівня. Заголовки без номерів портів (ICMP) залишають свої значення портових полів у значенні «0». Довжина пакета не витягується безпосередньо, а замість цього кожен пакет ділиться на довжину заголовка та довжину корисного навантаження, що дозволяє обчислювати більш складні статистики. Список отриманих ознак із кожного пакета подано в табл. 4.1. Дані витягуються лише з пакетів IPv4 та IPv6. Інші типи трафіку не надають жодної релевантної інформації і

лише можуть знизити продуктивність. З цієї причини не IP-трафік не повинен передаватись до модуля вилучення ознак.

Таблиця 4.1 - Список отриманих ознак із кожного пакета

Назва ознаки	Позначення	Тип даних	Опис
Часова мітка	m_1	Ціле число	Відображає час отримання або надсилання пакету у форматі UNIX-часу
IP-адреса джерела	m_2	Рядок	IP-адреса, з якої надходить трафік
IP-адреса призначення	m_3	Рядок	IP-адреса, на яку спрямований трафік
Протокол L4	m_4	Ціле число	Ідентифікатор протоколу транспортного рівня (TCP, UDP тощо)
Порт джерела	m_5	Ціле число	Номер порту, з якого надходить пакет
Порт призначення	m_6	Ціле число	Номер порту, на який спрямовано пакет
Довжина заголовка	m_7	Ціле число	Розмір заголовка пакету у байтах
Довжина корисного навантаження	m_8	Ціле число	Розмір корисних даних пакету у байтах
Час життя або TTL	m_9	Ціле число	Кількість переходів, за який пакет може існувати до свого зникнення

Виділені особливості пакетів у їхньому вигляді недостатньо корисні для методів машинного навчання. Вони мають низьку інформаційну цінність, а також їх кількість є великою. З цієї причини можна застосувати декілька алгоритмів видобування даних, щоб отримати інформацію, яка буде корисною для навчання та класифікації. Виділені ознаки далі обробляються у модуль збору статистики. На цьому етапі здійснюється контроль, яка частина безперервного потоку пакетів буде брати участь у обчисленні статистики та інших шаблонів видобування даних. На даному етапі використаємо модель вікна. Модель вікна розділяє нескінченний потік

даних на послідовності визначеної довжини та обчислює статистику для кожної з них незалежно. Відомо кілька моделей вікон, з яких найпопулярнішими є модель фіксованого вікна, ковзного вікна та згасаючого вікна (рис. 4.2).

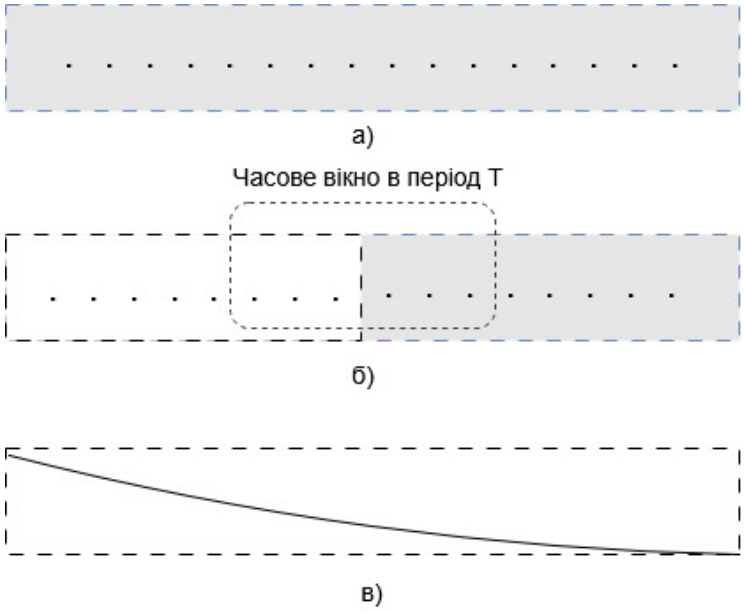


Рисунок 4.2 – Моделі часових вікон: а) фіксованого вікна; б) ковзного вікна; в) згасаючого вікна

У моделі фіксованого вікна аналіз даних розпочинається із початкового моменту часу, який називається фіксованою точкою, до поточного моменту часу. Фіксоване вікно можна визначити за кількістю об'єктів, спостережуваних з моменту встановлення нової фіксованої точки (наприклад, кожні 500 об'єктів) або за часом (наприклад, щогодини). Коли розпочинається новий період вікна, усі об'єкти, що зберігалися із попередньої фіксованої точки, видаляються. Особливий випадок моделі фіксованого вікна виникає, коли стартова точка дорівнює одиниці. У цьому випадку аналізується уся історія потоку даних. Одним із обмежень цієї моделі є складність у визначенні оптимальних фіксованих точок. Крім того, всі точки у вікні розглядаються як однаково важливі.

У моделі ковзного вікна існує вікно фіксованого розміру w від поточного моменту часу t_i. З плином часу, починаючи з поточного моменту t_i, вікно зберігає свій розмір w і ковзає вперед. Таким чином, кожне вікно містить лише об'єкти, що знаходяться в інтервалі $[t_i - \mathrm{w} + 1, t_i]$, тоді як старіші об'єкти відкидаються. Вікно можна визначити за часовими точками (наприклад, останні 50 часових точок) або за

об'єктами (наприклад, останні 500 об'єктів). У цьому випадку дані є рівновіддаленими. Ця модель підходить для застосувань, коли потрібно опрацьовувати лише найновіші об'єкти. Очевидно, що залежно від розміру вікна можна розглядати більше або менше історичних даних. Крім того, всі випадки у межах активного ковзного вікна вважаються однаково важливими.

У моделі згасаючого вікна кожен об'єкт асоціюється із вагою, яка залежить від часу його надходження. Коли новий об'єкт надходить, то йому присвоюється найвища можлива вага. Далі ця вага зменшується експоненційно із часом відповідно до функції старіння. На практиці досить часто в якості функції старіння обирають експоненціальну функцію виду:

$$\mathrm{f}(t) = 2^{-\lambda\cdot(t_{\text{п}}-t_{\text{с}})}, \tag{4.6}$$

де λ – коефіцієнт згасання, $t_{\text{п}}$ – це поточний момент часу, а $t_{\text{с}}$ – момент створення об'єкта.

На відміну від попередніх моделей, модель згасаючого вікна не відкидає об'єкти повністю, а старіші об'єкти роблять менший внесок, оскільки їм призначаються нижчі ваги. Інтерпретація цієї моделі є складнішою, оскільки потенційно всі об'єкти залишаються "активними", але з вагою, яка залежить від їхнього віку.

Оскільки є потреба у контролі активності клієнта із часом та обчисленні статистики цієї активності, то застосуємо не перекриваючу модель ковзного вікна. В такому випадку пропонується групувати вхідні дані за їх IP-адресою. При цьому кожній IP-адресі буде відповідати список вікон, у яких зберігатиметься статистика для певної IP-адреси. Зберігати лише одне крайнє вікно недостатньо, оскільки важливо детальніше дослідити процес комунікації. Наприклад, якщо активність певного клієнта різко зростає або падає протягом кількох вікон, то це може свідчити про зміну в його поведінці або потенційно підозрілу активність. З цієї причини необхідно зберігати кілька історичних вікон.

Формування вектору статистичних показників для мережного IP-трафіку за кожною IP адресою, що буде спостерігатися між часовими вікнами $[1, K_W]$, де K_W – кількість часових вікон, які задіяні для обчислення вектору статистичних показників, є важливим кроком. Множина метаданих M, яка отримується із кожного мережного пакету, не зберігаються у віконних структурах при їх надходженні на опрацювання.

Для збереження пам'яті та більш стислого опису трафіку використовуються статистичні ознаки, такі як кількість, середнє, медіанне, модальне значення, стандартне відхилення тощо, які обчислюються на основі отриманих даних. Кожна з цих характеристик представлена одним значенням, що дозволяє знизити використання пам'яті навіть у випадку, коли обробляється велика кількість клієнтів одночасно.

Процес формування вектору статистичних показників для мережного IP-трафіку при розбитті потоку мережних пакетів на часові вікна зображено на рис. 4.3. Вектор статистичних показників для мережного IP трафіку $F = \{f_i\}_{i=1}^{25}$ включає у себе дві групи статистичних показників:

1) узагальнені статистичні показники всередині вікон K_W для IP адреси IP_{dest} – $F_{inside,IP_{dest}}$;

2) статистичні показники між часовими вікнами $[1, K_W]$ – $F_{outside}$.

У підсистемі виявлення атак відмова в обслуговуванні у корпоративних мережах модуль збору статистики використовує узагальнені статистичні показники всередині вікон загальною кількістю чотирнадцять ознак (табл. 4.1), а також шістнадцять статистичних показників між часовими вікнами. Разом ці дві групи ознак дозволяють оцінити структурні та статистичні характеристики мережного трафіку.

Основні метрики включають загальну кількість пакетів, що передаються в межах заданого часового вікна, а також середній час між прибуттям цих пакетів, що допомагає оцінити ритм трафіку. Загальна кількість пакетів, що передаються у межах часового вікна, є основним показником, який допомагає виявити аномально високу активність, характерну для великих атак. Середній час між прибуттям пакетів використовується для аналізу ритму трафіку, адже під час атак він часто стає стабільно коротким через генерацію потоків із фіксованою швидкістю.

Ще одним показником, який використовується у векторі статистичних показників для мережного IP-трафіку, є середнє значення розміру пакетів, яке дозволяє оцінити типові розміри даних у потоці пакетів, тоді як медіанний і модальний розміри пакетів деталізують розподіл розмірів і допомагають виявити найчастіше використовувані параметри. Наприклад, розподілені атаки відмова в

обслуговуванні досить часто генерують мережні пакети одного розміру, що призводить до зменшення варіативності даних значень.

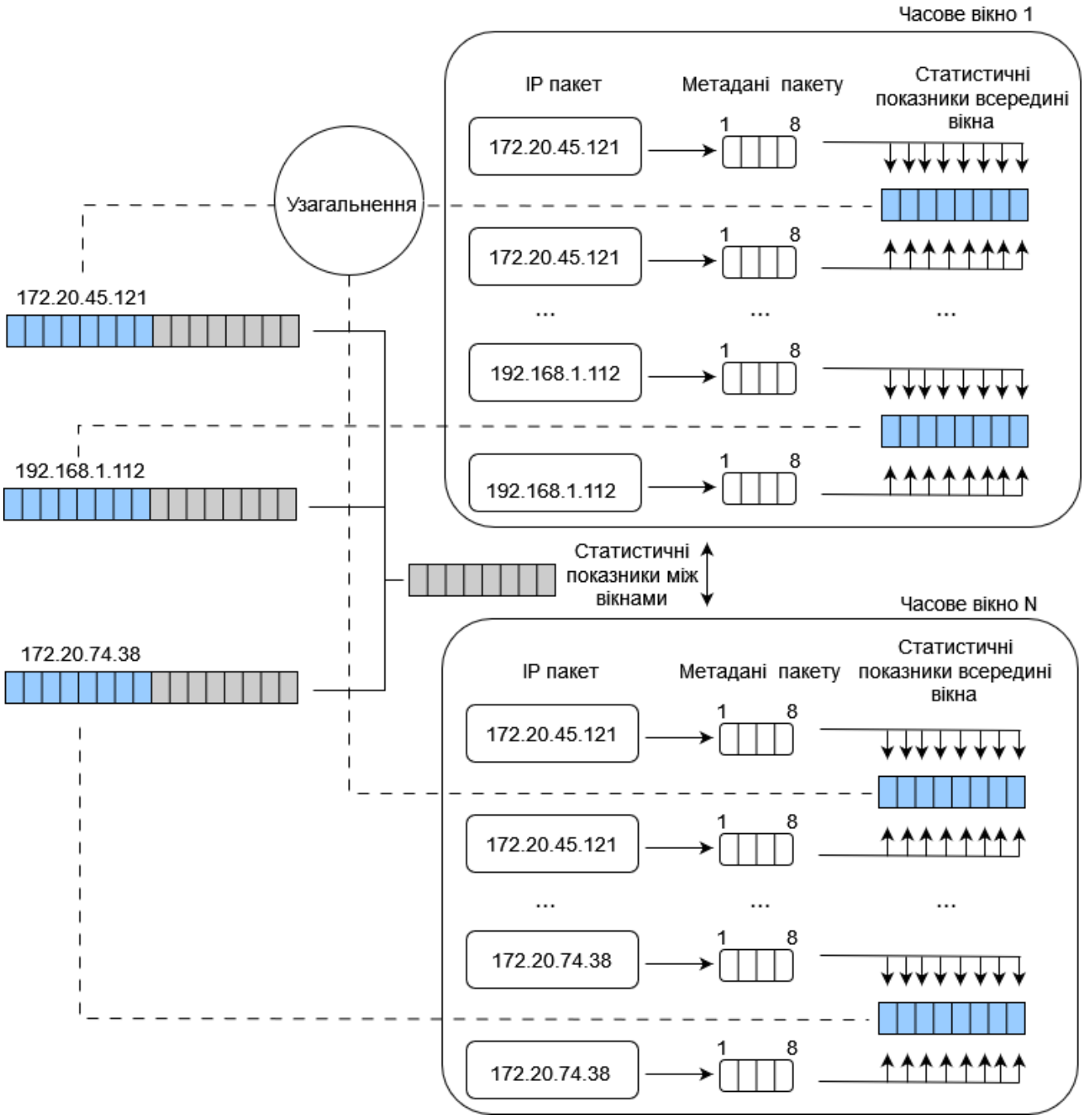


Рисунок 4.3 – Процес формування статистичних показників при розбитті потоку мережних пакетів на часові вікна

Кількість пакетів, що належать до TCP, UDP або ICMP, дозволяє класифікувати трафік за протоколами. Значний перекіс у бік одного з них, наприклад, домінування UDP або ICMP, може вказувати на специфічні атаки. Оцінка ентропії для кожного протоколу дозволяє зрозуміти ступінь

різноманітності пакетів і виявити впорядковані чи однотипні шаблони, характерні для бот-мереж. Ентропія портів-джерел використовується для оцінювання різноманітності джерел трафіку. У нормальних умовах вона має середнє значення, але під час атак, коли зловмисники використовують техніки рандомізації портів, ентропія зростає через велику кількість різних портів-джерел. Це є важливим індикатором автоматизованих атак. Таким чином, аналіз цих ознак допомагає виявляти аномалії у структурі мережного трафіку та своєчасно реагувати на можливі загрози.

Також використовуються середнє значення, медіанний і модальний розміри пакетів, які вказують на найбільш поширені характеристики обсягу даних у мережі. Ентропія кількості TCP, UDP та ICMP пакетів дає змогу оцінити ступінь упорядкованості або різноманітності трафіку за кожним із цих протоколів, що важливо для виявлення можливих аномалій. Також, враховується ентропія портів-джерел, яка дозволяє оцінити різноманітність джерел трафіку, що є критичним показником для аналізу динаміки мережі та визначення потенційних загроз.

Окрім цього, важливим показником є співвідношення розміру заголовка до загального розміру пакету. З цією метою здійснюється обчислення медіанного та модального співвідношення, що допомагає встановити відсоток кожного пакету в заголовку порівняно з корисним навантаженням.

Такі показники (табл. 4.2) створюють комплексну систему подання мережного трафіку, що дозволяє виявляти його особливості, закономірності, а також потенційні аномалії або небезпечні ситуації.

Проте, цього недостатньо для точного прогнозування наявності зловмисного трафіку. Якщо аналізувати лише одне вікно, наприклад, на часовий інтервал в 1 секунду, то швидкі сплески легітимного трафіку можуть створювати велику кількість пакетів за короткий проміжок часу, що в такому короткому періоді може мати схожі характеристики із DDoS-атаками. Тому, статистика одного вікна недостатня сама по собі. Замість цього значення кожного вікна не подаються безпосередньо в модель машинного навчання, а використовуються для обчислення більш складних характеристик.

Таблиця 4.2 - Статистичні показники, що обчислюються всередині кожного часового вікна w_i

№	Позначення	Опис
1	$f^{w_i}_{inside,p\_tot}$	Загальна кількість пакетів у межах часового вікна
2	$f^{w_i}_{inside,arr\_avg}$	Середній час між прибуттям пакетів
3	$f^{w_i}_{inside,s\_std}$	Середнє значення розміру пакетів
4	$f^{w_i}_{inside,s\_med}$	Медіанний розмір пакетів
5	$f^{w_i}_{inside,s\_mod}$	Модальний розмір пакетів
6	$f^{w_i}_{inside,c\_TCP}$	Кількість TCP пакетів
7	$f^{w_i}_{inside,c\_UDP}$	Кількість UDP пакетів
8	$f^{w_i}_{inside,c\_ICMP}$	Кількість ICMP пакетів
9	$f^{w_i}_{inside,entropy\_TCP}$	Ентропія TCP пакетів
10	$f^{w_i}_{inside,entropy\_UDP}$	Ентропія UDP пакетів
11	$f^{w_i}_{inside,entropy\_ICMP}$	Ентропія ICMP пакетів
12	$f^{w_i}_{inside,sp\_entropy}$	Ентропія портів-джерел
13	$f^{w_i}_{inside,med\_htratio}$	Медіанне співвідношення розміру заголовка пакету до розмір всього пакету
14	$f^{w_i}_{inside,mod\_htratio}$	Модальне співвідношення розміру заголовка пакету до розмір всього пакету

З цією метою здійснюється агрегація (узагальнення) статистичних показників за всіма вікнами для заданої IP-адреси IP_{dest}. Процес агрегації передбачає формування середніх значень всіх показників $f^{w_i}_{inside}$:

$$f_{inside} = \frac{1}{K_W}\sum_{i=1}^{K_W} f^{w_i}_{inside}. \tag{4.7}$$

Таким чином всі чотирнадцять ознак f_{inside} складають множину $F_{inside,IP_{dest}}$, що визначає узагальнені статистичні показники всередині вікон K_W для IP-адреси IP_{dest}. Для оцінювання мережного трафіку не лише в межах окремих часових вікон, але й між ними, використаємо статистичні показники між часовими вікнами [1, K_W], які формують множину $F_{outside}$.

Множину $F_{outside}$ формують ті ж самі показники, що й множину $F_{inside,IP_{dest}}$, але для кожного з них обчислюються середньоквадратичні відхилення між часовими вікнами так:

$$F_{outside} = \{std(f^{w_i}_{inside,p\_tot}), \dots, std(f^{w_i}_{inside,mod\_htratio})\} \quad (4.8)$$

Використання стандартного відхилення між усіма підсумковими вікнами необхідне через те, що зловмисники, як правило, створюють зловмисний трафік із передбачуваними шаблонами метаданих. Якщо припустити, що розмір пакетів рандомізований, а тривалість між їх надсиланням недетермінована, то шаблони рандомізації дадуть відносно подібну статистику в кожному проаналізованому вікні, якщо аналізувати мережний трафік з більш тривалої перспективи. Стандартне відхилення цих статистичних даних має вказувати на те, чи згенеровано пакети від легітимного клієнта, бо наявне велике значення дисперсії між вікнами, чи зловмисного джерела, бо наявне менше значення дисперсії. Це дозволяє виявляти динамічні зміни трафіку на рівні всього аналізованого періоду.

Крім стандартних відхилень, множина $F_{outside}$ включає два додаткові показники, які характеризують активність хоста як у межах окремих вікон, так і між усіма вікнами, коефіцієнти внутрішньовіконної і міжвіконної активності.

Згідно коефіцієнту внутрішньо віконної активності $f_{kwithin}$ будемо оцінювати активність хоста в підсумковому вікні так:

$$f_{kwithin} = \frac{t_e - t_s}{l_w}, \quad (4.9)$$

де t_e – це часова мітка часу завершення спілкування хоста; t_s – це часова мітка часу початку спілкування хоста; l_w – це довжина вікна.

Це дає приблизну оцінку того, як довго хост був активним у заданому вікні. ЗПЗ, яке генерує DDoS, переважно надсилає пакети через невеликі детерміновані інтервали. Таким чином, його внутрішньо віконна активність має бути близькою до одиниці, оскільки нові пакети надходять із заданої IP-адреси постійно. Легітимні хости можуть проявляти активність на початку вікна кілька разів. Їхній пакетний зв’язок не повинен досягати значень, близьких до одиниці у довгостроковій перспективі. Це значення обчислюється для кожного вікна і надається як середнє значення.

Згідно коефіцієнту міжвіконної активності $f_{kbetween}$ будемо оцінювати активність хоста між усіма підсумованими вікнами так:

$$f_{kbetween} = \frac{n_W}{id_E - id_S}, \tag{4.10}$$

де n_W – кількість підсумованих вікон, id_E – ідентифікатор останнього, а id_S ідентифікатор першого підсумованого вікна.

Якщо хост безперервно спілкується, то це значення дорівнює одиниці. Однак, якщо хост не надсилає достатньо даних або його дані взагалі не вибіркові, то для нього не буде створено запису вікна, тому можуть бути пропуски в ідентифікаторах вікон, які потрібно звести. Пропуски призведуть до зменшення значення міжвіконної активності, значення яких будуть менше ніж одиниця. Атаки, при яких дані надсилаються безперервно, матимуть значення, що дорівнює одиниці, майже завжди, крім повільних атак, тоді як легітимний трафік на основі періодичних збільшень активності досягатиме менших значень цього індикатора.

Таким чином, остаточно множина $F_{outside}$ буде складатись із 16 ознак, кожна із яких дозволяє оцінити зміни трафіку на рівні всього аналізованого періоду і задамо її так:

$$F_{outside} = \{std\left(f^{w_i}_{inside,p_{tot}}\right), \dots, std\left(f^{w_i}_{inside,mod_{htratio}}\right), f_{kwithin}, f_{kbetween}\} \tag{4.11}$$

Згідно кроків методу виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників можна обчислити та аналізувати характеристики мережного трафіку в умовах онлайн-сценарію з нескінченним потоком даних. В процесі функціонування із застосуванням даного методу в межах кожного часового вікна спочатку визначаємо чотирнадцять узагальнених статистичних показники, які включають чотири ознаки-лічильники, чотири ознаки середніх значень, дві ознаки центральної тенденції та чотири оцінки ентропії для кожної IP-адреси в кожному вікні (табл. 4.1). Після цього здійснюється отримання шістнадцяти статистичних показників між часовими вікнами, із яких чотирнадцять є показниками стандартного відхилення із назвами ознак, які визначені у першій множині, між часовими вікнами [1, K_W]. Оскільки підсистема виявлення атак відмова в обслуговуванні у корпоративних мережах здатна обробляти десятки гігабітів трафіку за секунду, то зберігати всі дані для кожного пакета та кожного вікна в пам’яті є складним завданням, яке потребує багато ресурсів. Для забезпечення ефективності обчислень

без збереження великого обсягу даних у пам’яті використаємо алгоритми потокового видобування та опрацювання даних.

Розглянемо детальніше процес обчислення статистичних потокових показників. Для кожного пакета, який відповідає певній умові, його відповідний лічильник збільшуємо на одиницю. Наприклад, якщо пакет містить TCP-заголовок, то збільшуємо лічильник для TCP-сегментів. Однак обчислення решти статистичних даних, таких як ознаки центральної тенденції або ентропії, є дещо складнішим завданням. Загальновідома форма середнього значення і стандартного відхилення вимагає опрацювання всього набору даних перед отриманням результату. Це стає проблемою, оскільки немає можливості зберігати дані для подальшого опрацювання. Тому, необхідно використовувати потокові, також відомі як ковзаючі або рухомі, алгоритми. Обчислення потокового середнього значення може бути виведене із його стандартної форми:

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i. \tag{4.12}$$

Для випадку потокового середнього значення опрацьовується безперервний потік значень x_t, де t – це індекс поточного етапу або кроку. Нехай μ_t – це середнє значення після t-го кроку або для перших t значень, x_t – це нове значення, яке надійшло на поточному кроці. На першому кроці $t = 1$, то поточне середнє μ_1 буде дорівнювати x_1. Для наступних кроків $t = 2,3,4,..$, то середнє значення на t-му кроці визначимо так:

$$\mu_t = \frac{1}{t}\sum_{i=1}^{t} x_i. \tag{4.13}$$

Наведена формула (4.13) є середнім значенням для перших t значень. Тепер визначимо це середнє значення через середнє значення попереднього кроку μ_{t-1} так:

$$\mu_{t-1} = \frac{1}{t-1}\sum_{i=1}^{t-1} x_i \tag{4.14}$$

Визначимо суму для поточного кроку t так:

$$\sum_{i=1}^{t} x_i = \sum_{i=1}^{t-1} x_i + x_t \tag{4.15}$$

Тому середнє значення на поточному кроці t задамо так:

$$\mu_t = \frac{1}{t}(\sum_{i=1}^{t-1} x_i + x_t) \quad (4.16)$$

Здійснимо перетворення, замінивши суму для $\sum_{i=1}^{t-1} x_i$ через μ_{t-1} і отримаємо:

$$\mu_t = \frac{1}{t}((t-1)\mu_{t-1} + x_t) \quad (4.17)$$

Поділимо отриманий вираз на час t і отримаємо формулу для визначення потокового середнього значення так:

$$\mu_t = \mu_{t-1} + \frac{x_t - \mu_{t-1}}{t} \quad (4.18)$$

де, μ_t – середнє значення на поточному кроці; μ_{t-1} – середнє значення на попередньому кроці; x_t – нове значення ознаки, що надійшло; t – номер поточного кроку.

Таким чином, згідно визначень та формули (4.18), здійснюється обчислення поточного середнього значення та значення лічильника кількості опрацьованих елементів із збереженням попередньо обчисленого середнього значення (μ_{t-1}). Для цього залучено чотири ознаки на основі обчислення ентропії TCP, UDP, ICMP пакетів та ентропії портів-джерел. Для визначення даних показників ентропії використаємо метод накопичення частотних таблиць, що дозволяє опрацьовувати великий обсяг даних, не зберігаючи всі елементи потоку. Замість цього накопичуються частоти унікальних значень у вигляді портів або інших елементів у хеш-таблиці або хеш-дереві.

Процес визначення ентропії задамо кроками.

1. Формування частотної таблиці. Під час отримання кожного нового пакету здійснюємо визначення порту, із якого надійшов пакет, а потім виконується оновлення відповідного значення частоти для цього порту в хеш-таблиці. Нехай p_i – це порт пакета, а частота $f(p_i)$ – кількість появ p_i у поточному інтервалі. Тоді, хеш-таблиця буде містити всі унікальні порти, які з'явилися в інтервалі, разом із їхньою частотою $f(p_i)$.

2. Обчислення значення ймовірності для кожного порту. Згідно частотної таблиці обчислюємо ймовірність появи кожного порту в потоці так:

$$P(p_i) = \frac{f(p_i).}{N} \tag{4.19}$$

де $P(p_i)$ – ймовірність появи порту; $f(p_i)$ – частота порту p_i, N –загальна кількість пакетів у даному інтервалі.

3. Розрахунок ентропії Шеннона. Ентропія в такому випадку обчислюється за стандартною формулою Шеннона:

$$H = -\sum_{i=1}^{k} P(p_i) \cdot \log_2(P(p_i)) \tag{4.20}$$

де k – загальна кількість унікальних портів у частотній таблиці, $P(p_i)$ – ймовірність появи порту p_i.

В методі виявлення розподілених атак відмова в обслуговуванні задіяно дві ознаки центральної тенденції такі, як модальний та медіанний розміри пакетів. Визначення даних показників є важливим при виявленні атак такого типу, оскільки у випадку модального розміру пакетів це дозволяє визначити найбільш поширену величину розміру пакета у звичайному незміненому трафіку, тоді як через медіанний розмір пакета здійснюється його визначення, а значення розміру пакета є типовим для більшості трафіку, навіть якщо в наборі даних є деякі значення, які значно відрізняються. Наприклад, пакети з великим розміром, які можуть бути результатом атаки.

Для потокових даних традиційні методи визначення медіани, які вимагають сортування всіх елементів, не підходять, оскільки це є неефективно в умовах нескінченного потоку. Тому, для потокових даних використаємо алгоритм із двома підмножинами: одна підмножина зберігає менші значення до медіани; друга – зберігає більші значення після медіани. Для кожного нового елемента визначається, в яку підмножину він має потрапити. Якщо елемент менший або рівний максимальному значенню в min-підмножині, то він включається до max-підмножини, в іншому випадку – до min-підмножини. Після кожного додавання елемента необхідно збалансувати підмножини так, щоб різниця в кількості елементів між ними не перевищувала одиниці. Якщо кількість елементів у min-підмножині і max-підмножині однакова, то медіаною буде середнє значення між їхніми коренями. Якщо

кількість елементів в одній з підмножин більша, то медіаною буде корінь більшої підмножини.

Іншою ознакою центральної тенденції є модальний розмір пакетів. В загальному випадку мода визначається як значення, яке з'являється найчастіше у наборі даних. Для потокових даних виконаємо кроки.

1. Інкрементування лічильника частоти. Для кожного нового елемента потоку оновлюємо його лічильник у хеш-таблиці, де ключем буде сам елемент, наприклад, порт, а значенням — кількість його поява у потоці.

2. Визначення елемента, що з’являється найбільш часто. Після кожного опрацювання нового елемента підтримується змінна, що зберігає елемент, який має найбільшу кількість появ, а також оновлюємо дану змінну, якщо поточний елемент з'являється більше разів, ніж попередній максимальний.

3. Адаптивне оновлення. Оскільки потік може містити великі обсяги даних, то періодично здійснюється оновлення найбільш частих елементів, а також видаляються елементи з малими частотами, щоб зменшити витрати пам'яті.

Для визначення статистичних показників між часовими вікнами обчислюється чотирнадцять показників стандартного відхилення. Для їх обчислення скористаємось алгоритмом Велфорда, який забезпечує обчислення дисперсії в реальному часі за один прохід. Це досягається шляхом оновлення допоміжного значення s для кожного нового елемента разом з поточним середнім $\bar{x}_n$ так:

$$s_n = s_{n-1} + (x_n - \bar{x}_n) \cdot (x_n - \bar{x}_{n-1}) \tag{4.21}$$

Після опрацювання всіх елементів дисперсія обчислюється так:

$$s^2 = \frac{s_n}{n-1} \tag{4.22}$$

Тоді стандартне відхилення потокових даних обчислюється як квадратний корінь.

Таким чином, розроблений метод виявлення комп’ютерних атак з використанням приманок і пасток згідно аналізу статистичних показників мережного трафіку. Особливість методу полягає в формуванні комплексного вектору з тридцяти статистичних ознак, які характеризують поведінку трафіку на двох рівнях: чотирнадцять внутрішньо віконних показників відстежують активність кожної IP-адреси протягом визначеного інтервалу часу; шістнадцять міжвіконних метрик

фіксують зміни характеристик трафіку між послідовними часовими вікнами. Це дозволяє виявляти як миттєві аномалії, так і відстежувати еволюцію поведінки трафіку протягом тривалого періоду. Для забезпечення роботи в режимі реального часу імплементовано в систему спеціалізовані алгоритми потокового опрацювання: потокові алгоритми обчислення середніх значень; метод накопичення частотних таблиць для визначення ентропії Шеннона; алгоритм з двома підмножинами для динамічного обчислення медіани та алгоритм Велфорда для визначення стандартного відхилення. Такий підхід дозволив забезпечити опрацювання великого обсягу мережного трафіку без збереження повного обсягу пакетів у пам'яті, що дозволило імплементувати розроблений метод виявлення комп'ютерних атак з використанням приманок і пасток згідно аналізу статистичних показників мережного трафіку у підсистему виявлення атак відмова в обслуговуванні корпоративних мереж.

4.2 Постановка та результати експерименту

Для підтвердження спроможності реалізувати запропоновані методи та архітектуру обманних систем було розроблено обманну систему з приманками і пастками, зокрема реалізація підтримки цілісності системи та функціонування в ізольованих середовищах, яка подана в [181] та додатках Г та Д. Розроблена обманна система може функціонувати в корпоративних мережах з кількістю вузлів до 2000 одиниць окремими компонентами, підтримувати комунікацію та здійснювати керування приманками і пастками, активувати приманки в своїх компонентах, здійснювати збір даних у вузлах корпоративних мереж, опрацьовувати показники мережного трафіку, приймати рішення щодо своїх наступних кроків в процесі здійснення зловмисниками КА, виявляти КА типу відмова в обслуговуванні.

Для забезпечення прийняття рішень щодо своїх наступних кроків центр прийняття рішень системи організовано згідно алгоритму дискретної оптимізації молі і полум’я. Метод виявлення КА типу відмова в обслуговуванні реалізовано в одній з компонент обманної системи. Обманні функції розробленої системи стосуються зміни в функціонуванні системи під час КА чи встановлених дій ЗПЗ, зміна вузлів з центром прийняття рішень системи, зменшення активності реальних об’єктів в

корпоративних мережах, збільшення активності між компонентами системи включно з приманками, активізація приманко і пасток.

Перший експеримент. Здійснимо спочатку постановку експерименту з розробленою обманною системою з приманками і пастками в частині її функцій, які реалізують вибір наступних кроків системи та їх коригування в процесі здійснення КА, керування приманками і пастками, а також збільшення і зменшення активності у вузлах корпоративних мереж.

Розроблена обманна система доповнює засоби захисту корпоративних мереж. Вона спрямована на виконання таких дій:

1) надання хибних об’єктів для виявлення КА чи дій ЗПЗ;

2) активізує хибні об’єкти для КА (приманки - 37, хибні сервери – 3);

3) збір даних про вектори КА та процеси, які використовує зловмисник;

4) збір індикаторів компрометації для попередження про потенційні КА;

5) створення та видалення хибних об’єктів для атак з метою здійснення заплутування зловмисників.

Серед хибних об’єктів для атак наявні пастки, які в процесі здійснення на них атак зберігають всі дії атакуючого. Це допомагає аналізувати шлях зловмисника. Імітування мережних взаємодій вони не здійснюють. Їх наявність затримує зловмисників та відволікає їх, а також спонукає витрачати ресурси і час. Таких об’єктів у встановленій системі наявні десять. Пастка повідомляє центр системи про будь-яку спробу взаємодії з нею, тобто вона є індикатором атаки. З неї можна отримати деталі події, зокрема, адресу, порт джерела та цілі, протокол взаємодії, час спрацьовування.

В розгорнутій системі наявні чотири приманки, які імітують повноцінні операційні системи, для емуляції окремих робочих місць та сервера і є окремими сервісами.

Кожна компонента обманної системи є програмою, яку встановлено в комп’ютерні станції та сервери для здійснення комунікації з центром системи, між рештою компонентів, виконання команд з центру системи та передавання даних до центру системи. Таким чином, кожна компонента обманної системи виконує чотири універсальні дії:

1) комунікація з центром системи;

2) виконання команд системи, які надходять з центру системи;

3) передавання даних до центру системи;

4) комунікація з компонентами системи.

Також, компонента системи здійснює такі спеціальні дії:

1) збір даних про стан вузлів корпоративних мереж, в які вона встановлена;

2) поширення приманок згідно команд з центру системи;

3) здійснення емуляції активності у вузлі в мережі;

4) реагування на події.

Компоненти системи налаштовані на виконання у привілейованому режимі роботи, щоб уникнути виявлення користувачами корпоративних мереж. В приманках розміщено дані для отримання доступу до пастки, тобто логін, пароль або ключ. Також, приманки формують паралельне середовище корпоративної мережі для заплутування зловмисника. Для цього між ними здійснено обмін повідомленнями через певний випадковий час з певною щільністю і саме між ними. Ці повідомлення містять заготовлені файли в різних форматах, в яких міститься відкрита та закодована інформація.

Збір даних компонентами системи здійснюються при їх встановлення і з певним періодом часу. До таких даних віднесено дані про встановлене ПЗ, його версії та дати встановлення. Така інформація може порівнюватись з базами вразливостей.

Для отримання інформації про атаку, яка породжена зловмисником з середини периметру корпоративної мережі, обманна система аналізує вміст пастки щодо часу, порта джерела з’єднання із даними компоненти, щоб встановити процес, який її запустив.

Для керування реальними та хибними об’єктами в корпоративних мережах розроблена обманна система виконує такі дії:

1) закриває можливість комунікації з комп’ютерною станцією чи сервером на певний час до отримання відповідного повідомлення з командою від центру системи до компоненти;

2) закриває частину портів в комп’ютерній станції чи сервері під час КА чи дій ЗПЗ на певний час до отримання відповідного повідомлення з командою від центру системи до компоненти;

3) повертає можливість комунікації з комп'ютерною станцією чи сервером на певний час до отримання відповідного повідомлення з командою від центру системи до компоненти;

4) здійснює активізацію приманки, зокрема створюючи паралельну хибну корпоративну мережу з декількох приманок для імітації активності;

5) закриває частину портів в комп'ютерній станції чи сервері з приманкою під час КА чи дій ЗПЗ на певний час до отримання відповідного повідомлення з командою від центру системи до компоненти;

6) відкриває всі порти в комп'ютерній станції чи сервері з приманкою під час КА чи дій ЗПЗ на певний час до отримання відповідного повідомлення з командою від центру системи до компоненти;

7) активує вузол мережі з пасткою;

8) спрямовує в пастку трафік КА або поширення ЗПЗ;

9) залишати поточний стан компоненти і комп'ютерної станції чи серверу без змін;

10) активізація центру системи в компоненті;

11) деактивація центру системи в компоненті.

Ці дії задамо вектором так:

$$v_{os,r,i} = \left(v_{os,r,i,1}, v_{os,r,i,2}, \dots, v_{os,r,i,11}\right), \tag{4.23}$$

де $v_{os,r,i,j}$ – j-та координата i-того вектору; $j = 1,2, \dots ,11$.

Дії згідно формули (4.23) стосуються винятково однієї компоненти. Якщо застосувати ці дії до кожної окремої компоненти системи, то отримуємо, зокрема для розглядуваного експерименту, 275 векторів. Цими векторами буде визначатись простір пошуку. Кожен вектор буде характеризувати окремий стан окремої компоненти, а в цілому вони будуть відображати стан всієї системи. Таким чином, обманна система в просторі пошуку своїх наступних кроків буде обирати послідовності кроків, які стосуватимуться переведення вузлів корпоративної мережі в нові стани або залишати їх в поточних станах згідно дій, що визначені за формулою (4.23). Всі 40 приманок можуть бути активізовані в процесі КА в компонентах системи, тобто у вузлах корпоративної мережі, в які вони встановлені.

За результатами функціонування розробленої обманної системи з приманками і пастками здійснимо аналіз вибору наступних кроків в процесі прийняття рішень центром системи. Відомості про вузли корпоративних мереж подано частково в табл. 4.3 та детально в табл. В.1 (Додаток В), координати розміщення вузлів корпоративних мереж подано частково в табл. 4.4 та детально в табл. В.2 (Додаток В).

Таблиця 4.3 - Відомості про вузли корпоративних мереж (фрагмент)

Номер компоненти системи	Тип вузла: комп'ютерна станція; сервер	Номер сегменту: 1..25	IP-адреса вузла	Наявність та тип активної приманки та/чи пастки	Належність вузла демілітаризованій зоні	Тип публічного сервісу	Операційна система	Наявність центру системи	Номер будівлі
1	2	3	4	5	6	7	8	9	10
1	КС	1	10.0.1.10	-	-	-	Windows 10	-	1
2	КС	1	10.0.1.11	-	-	-	Windows 10	-	1
3	сервер	1	10.0.1.12	-	-	-	Windows 10	-	1
4	КС	1	10.0.1.13	-	-	-	Ubuntu 22.04	-	1
5	КС	1	10.0.1.14	-	-	-	Ubuntu 22.04	-	1
6	КС	1	10.0.1.15	-	-	-	Ubuntu 22.04	-	1
7	сервер	1	10.0.1.16	+	-	-	Windows 10	+	1
8	КС	1	10.0.1.17	-	-	-	Windows 10	-	1
9	КС	2	10.0.2.10	-	-	-	Windows 10	-	1
10	КС	2	10.0.2.11	-	-	-	Windows 10	-	1
11	КС	2	10.0.2.12	-	-	-	Windows 10	-	1
…	…	…	…	…	…	…	…	…	…
273	КС	25	10.0.25.17	-	-	-	Windows 10	-	3
274	КС	25	10.0.25.18	-	-	-	Windows 10	-	3
275	КС	25	10.0.25.19	-	-	-	Windows 10	-	3

Формування слідів спіралей протягом часу дослідження системи подано їх координатами в табл. 4.5 та детальніше в табл. В.3 (Додаток В). Час, протягом якого було отримано сліди спіралей, є третьою координатою точки в координатному тримірному просторі. Під час експерименту тривалістю 2 год. Відбулось сімнадцять оновлень кроків для трьох окремих впливів на корпоративну мережу. Після

виконання кожної ітерації алгоритму молі і полум'я в центрі системи відбувалось оновлення кроків системи. Кількість спіралей відповідає кількості оновлень кроків.

Таблиця 4.4 - Координати розміщення вузлів корпоративних мереж (фрагмент)

Номер компоненти системи	Номер сегменту: 1..25	IP-адреса вузла	x_{1,R_m}	x_{2,R_m}	R_m	Номер дії: 1..11
1	1	10.0.1.10	0,5403023	0,841471	1	3
2	1	10.0.1.11	-0,432178	0,9017883	1	6
3	1	10.0.1.12	-0,994367	0,1059875	1	5
4	1	10.0.1.13	-0,612548	-0,790433	1	7
5	1	10.0.1.14	0,3507973	-0,936451	1	3
..	…	…	…	…	…	…
273	25	10.0.25.17	21,44469	12,850108	25	4
274	25	10.0.25.18	10,000324	-22,91274	25	11
275	25	10.0.25.19	-24,00405	-6,986111	25	4

Таблиця 4.5 - Координати слідів спіралей (фрагмент)

Номер компоненти системи	Номер сегменту: 1..25	IP-адреса вузла	x_{1,R_m}	x_{2,R_m}	t, мс	R_m	Номер дії: 1..11
Слід 1							
249	23	10.0.23.13	21,28941656	-8,704064725	18432	23	3
237	21	10.0.21.17	17,41586021	-11,73404504	39785	21	11
219	19	10.0.19.14	1,715469356	-18,9223985	61204	19	7
198	17	10.0.17.22	6,878939022	-15,54606696	75519	17	8
174	16	10.0.16.12	-1,846605705	-15,89308175	104883	16	9
153	14	10.0.14.20	7,961602657	11,51576672	129447	14	8
131	13	10.0.13.12	-0,336948651	12,99563256	171902	13	5
Слід 2-16							
…	…	…	…	…	…	…	…
Слід 17							
221	18	10.0.18.23	-17,84158	-2,382869	6869327	18	4
171	16	10.0.16.11	-10,27336	-12,26613	6931140	16	7
170	16	10.0.16.10	-15,32255	-4,606453	6984023	16	6
115	12	10.0.12.10	10,126248	-6,438875	7029918	12	11
83	9	10.0.9.11	-5,688675	-6,974165	7083376	9	10
32	9	10.0.3.19	-1,112535	2,7860843	7198642	3	9

Таким чином, за 2 год. функціонування обманної системи було отримано 17 нових кроків. На об'єкти було здійснено в часі три різних впливи і кількість оновлень

кроків для опрацювання цих впливів подано в табл. 4.6. У векторі кроків наявні 275 координат, кожна з яких може приймати одне з одинадцяти значень з формули (4.23).

Таблиця 4.6 - Результати першого експерименту щодо кількості впливів в корпоративній мережі

№ з/п	Номер впливу: 1..3	Час оновлення кроків обманної системи, мс	Кількість кроків, які оновлено, в загальному векторі кроків
1	1111111	18432	7
2	11111	236926	5
3	111	562043	3
4	111111	757114	6
5	11111111	1147255	8
6	22222	1667443	5
7	2222222	1992560	7
8	222222222	2447725	9
9	2222222222	3032936	7
10	222222222	3488101	5
11	22222	3813218	7
12	3333333	4268383	8
13	33333333	4788571	10
14	3333333333	5438806	8
15	33333333	5958994	7
16	333333	6414158	6
17	3333333	6869327	7

В табл. 4.6 наявна інформація про кількість кроків, які оновлено, в загальному векторі кроків для кожного оновлення, які подано в табл. 4.5.

Кожному впливу відповідає декілька спіралей. Результатом завершення кожного впливу були попадання відомостей про атаку (вплив) в пастку. Крім того, під час атаки здійснювалось сканування вузлів корпоративної мережі і, при цьому, компоненти системи з приманками допускали проведення сканування портів комп’ютерних станцій, в яких вони розміщені, а в значній частині комп’ютерних станцій порти поступово закривались. Компоненти обманної системи в процесі здійснення КА формували комунікацію між собою для створення хибної корпоративної мережі і поступово вилучали свої компоненти із зовнішніх сегментів

(зовнішніх кіл), орієнтуючи впливи на внутрішні сегменти (внутрішні кола). Обмеженнями цього експерименту є відсутність зміни центром системи вузла корпоративної мережі, а також координати пастки, оскільки основною метою експерименту була перевірка спроможності системи реалізувати підбір кроків системи у відповідних компонентах на основі алгоритму молі і полум'я.

На рис. 4.4 зображено сімнадцять слідів спіралей на координатній площині та 275 об'єктів на 25 колах. Оскільки, зміна стану об'єктів відбувалась 17 разів згідно даних табл. 4.6, то на рис. 4.4 зображено 17 різних систем координат.

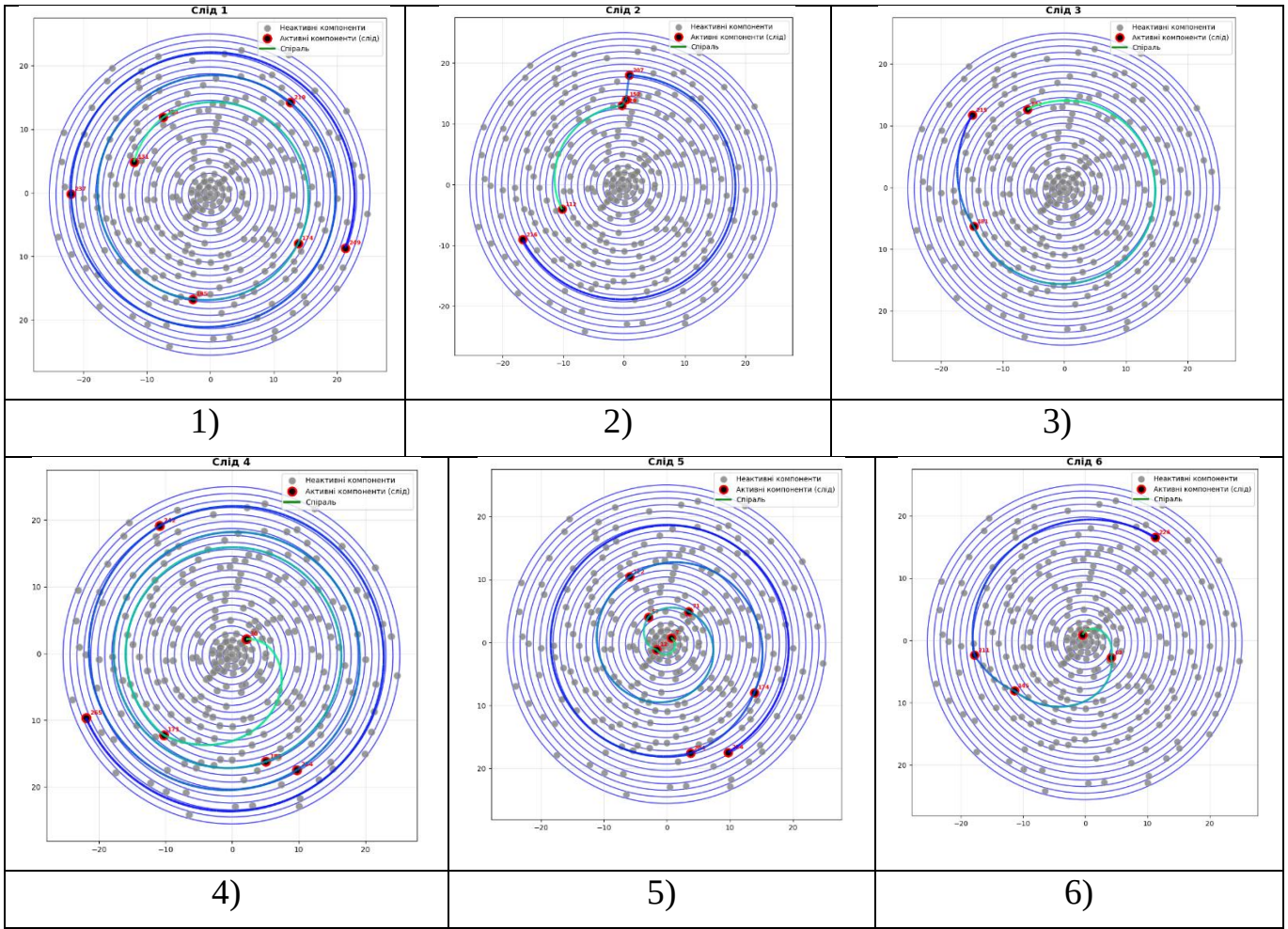
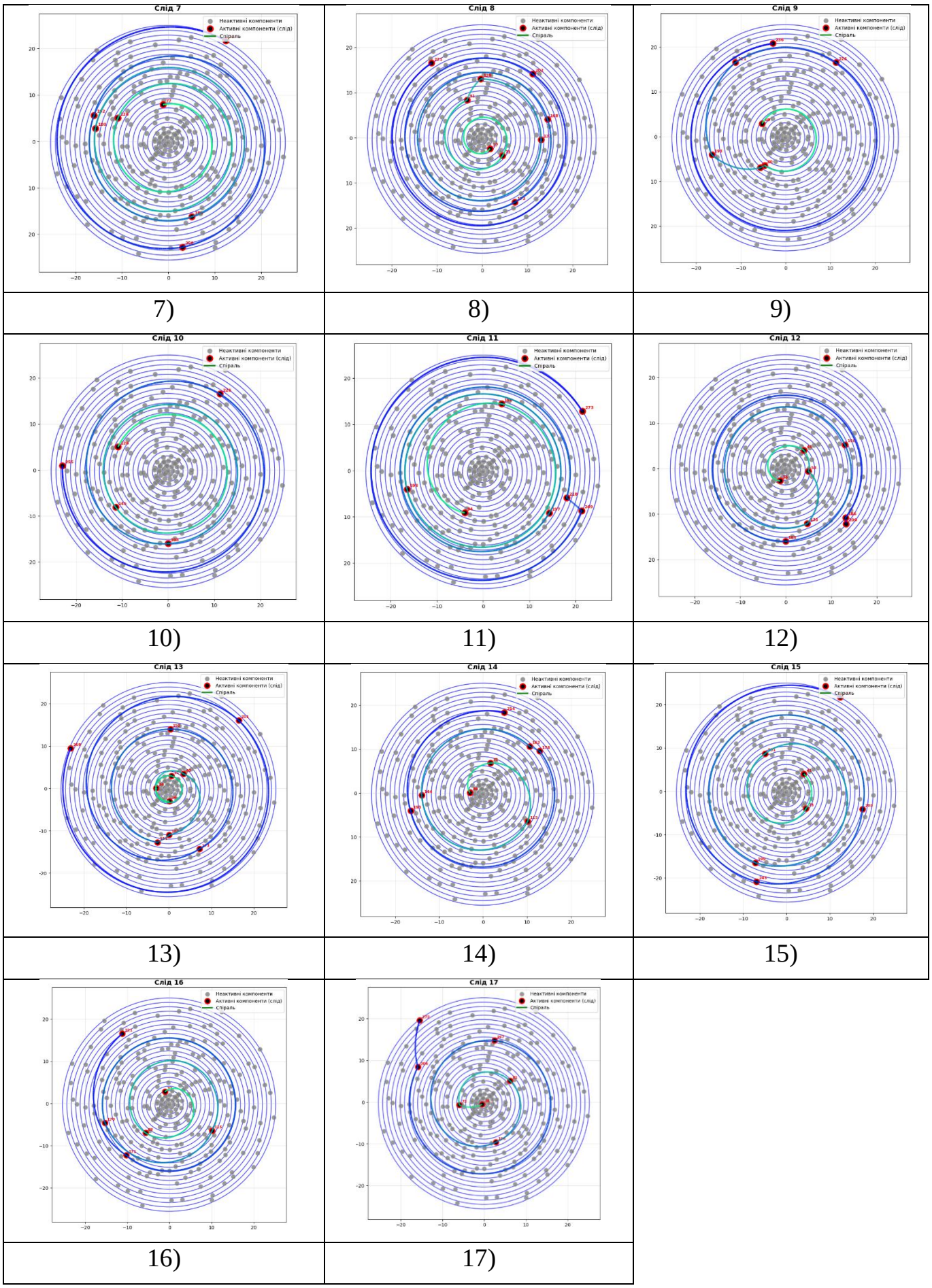


Рисунок 4.4 – Сімнадцять слідів спіралей на координатній площині (початок)



7) 8) 9)

10) 11) 12)

13) 14) 15)

16) 17)

Рисунок 4.4 – Сімнадцять слідів спіралей на координатній площині (кінець)

Таким чином, розроблена ОСПП здійснює керування в корпоративній мережі, зокрема і приманками та пастками. При впливах в операційному середовищі корпоративної мережі вона забезпечує вибір кроків згідно алгоритму молі і полум'я. На рис. 4.4 та в табл. 4.5 показано перебудову кроків системи в результаті кожного впливу, причому в межах одного впливу було здійснено декілька перебудов, що встановлено за результатами трьох впливів. Тому, завдання з вибору кроків та їх зміни розроблена система виконує коректно. Оскільки зміна кроків системи відбувалась багатократно і завершувалась винятково після завершення впливів, то система досягала глобального оптимуму в просторі пошуку, що відповідає досягненню мети дослідження.

Другий експеримент. Для проведення експериментів з розробленою обманною системою здійснимо дослідження методу виявлення КА. Основним завданням експерименту було встановлення впливу параметрів методу на якість детектування зловмисної активності та порівняння ефективності різних алгоритмів машинного навчання при роботі з базовими та запропонованими векторами ознак.

Для дослідження ефективності запропонованого методу було розроблено дослідний зразок системи на мові програмування Python із використанням бібліотек Scapy для захоплення та аналізу мережних пакетів, NumPy та Pandas для обробки статистичних даних, Scikit-learn для імплементації алгоритмів машинного навчання. Модуль вилучення ознак реалізовано як асинхронний обробник пакетів, який функціонує в режимі реального часу. Модуль збору статистики використовує оптимізовані структури даних для зберігання інформації про часові вікна та ефективного обчислення агрегованих показників. Модуль менеджера машинного навчання забезпечує управління життєвим циклом моделей, включаючи навчання, валідацію, збереження та завантаження натренованих моделей машинного навчання.

Для проведення експериментального дослідження було використано набір даних CIC-IDS2017. Для цілей дослідження було відібрано зразки нормального трафіку та DoS/DDoS атак, а саме slowloris, Slowhttptest, Hulk та GoldenEye. Загальний обсяг досліджуваних даних становив 128325 мережних потоків, з яких 56863 представляли нормальний трафік, а 71462 відносилися до атак відмови в обслуговуванні. При проведенні експерименту дані було розділено у пропорції 70% для навчання, 15% для валідації та 15% для тестування моделей.

Експериментальне дослідження передбачало порівняльний аналіз трьох конфігурацій векторів ознак, які відображають послідовне нарощування інформативності вхідних даних. Перша конфігурація представляла базові метадані пакетів без статистичної обробки та включала дев’ять ознак, що представлені в табл. 4.7. Друга конфігурація застосовувала запропонований метод обчислення внутрішньо-віконних статистичних показників та включала чотирнадцять ознак, тоді як третя конфігурація представляла повний вектор статистичних показників із тридцятьма ознаками, який об’єднував чотирнадцять внутрішньо-віконних показників та шістнадцять міжвіконних статистичних метрик, що включали стандартні відхилення для всіх внутрішньо-віконних ознак між часовими вікнами, а також коефіцієнти внутрішньо-віконної та міжвіконної активності хоста.

Таблиця 4.7 - Гіперпараметри методу виявлення КА

№	Гіперпараметр	Досліджені значення	Обране значення
1	Тривалість одного часового вікна	5 с, **10 с**, 15 с, 30 с	10 с
2	Мінімальна кількість пакетів у вікні	20, 50, 100	50
3	Кількість часових вікон у вибірці	3, 6, 9	6
4	Час існування статистики	30 с, **60 с**, 120 с	**60 с**

Параметри часових вікон було визначено на основі аналізу характеристик досліджуваного трафіку та обмежень обчислювальних ресурсів. Вибір оптимальних значень здійснювався експериментально серед кількох досліджуваних варіантів гіперпараметрів (табл. 4.7). Час існування статистики встановлено на рівні 60 секунд, що дозволяє фіксувати короткострокові аномалії без надмірного накопичення застарілих даних. Тривалість одного часового вікна становила 10 секунд, мінімальна кількість пакетів для формування статистики всередині вікна дорівнювала 50, а загальна кількість часових вікон для обчислення комплексного вектору статистичних показників було встановлено на рівні шести вікон.

Для кожної конфігурації вектору ознак було проведено навчання та оцінку трьох алгоритмів машинного навчання, а саме методу випадкового лісу, градієнтного бустингу та методу опорних векторів. Алгоритм випадкового лісу використовував двісті дерев рішень із максимальною глибиною дерева п’ять рівнів та критерієм розбиття на основі індексу Джині. Градієнтний бустинг застосовував двісті естиматорів із швидкістю навчання 0.1 та максимальною глибиною дерев три рівні

для запобігання перенавчання. Метод опорних векторів використовував радіальну базисну функцію як ядро із параметром регуляризації встановленим на рівні одиниці та коефіцієнтом гамма у режимі автоматичного визначення. Усі гіперпараметри було налаштовано через процедуру крос-валідації на валідаційній вибірці для забезпечення оптимальної продуктивності кожного алгоритму за відповідної конфігурації вектору ознак.

Оцінка ефективності здійснювалася за стандартними метриками класифікації бінарних задач, включаючи точність класифікації як відношення кількості правильно класифікованих зразків до загальної кількості зразків, точність виявлення атак як відношення істинно позитивних спрацювань до суми істинно позитивних та хибно позитивних спрацювань, повноту виявлення як відношення істинно позитивних спрацювань до суми істинно позитивних та хибно негативних випадків, збалансовану F-міру як гармонічне середнє точності та повноти. Додатково було обчислено площу під ROC-кривою для аналізу роботи класифікаторів у різних точках прийняття рішень та оцінки стабільності методів за різних порогових значень. Обчислювальна складність методів оцінювалася через вимірювання середнього часу обробки одного мережного пакету протягом ста ітерацій для кожної конфігурації із усередненням результатів для зменшення впливу випадкових флуктуацій продуктивності системи.

Результати (рис. 4.5) підтвердили гіпотезу про те, що врахування як внутрішньо-віконних, так і міжвіконних статистичних показників дозволяє досягти вищої якості класифікації незалежно від обраного алгоритму машинного навчання, при цьому найбільший приріст ефективності спостерігався при використанні повного вектору із тридцятьма ознаками.

Аналіз впливу конфігурації векторів ознак на ефективність класифікації виявив наступні закономірності для методу випадкового лісу. Використання базових метаданих пакетів без статистичної обробки забезпечило F-міру на рівні 86.3%, точність класифікації 87.4%, точність виявлення атак 86.8% та повноту 85.9%. Впровадження внутрішньо-віконних статистичних показників підвищило ефективність до 93.5%, 94.2%, 93.7% та 93.3% відповідно, що становить приріст приблизно семи відсоткових пунктів за всіма метриками.

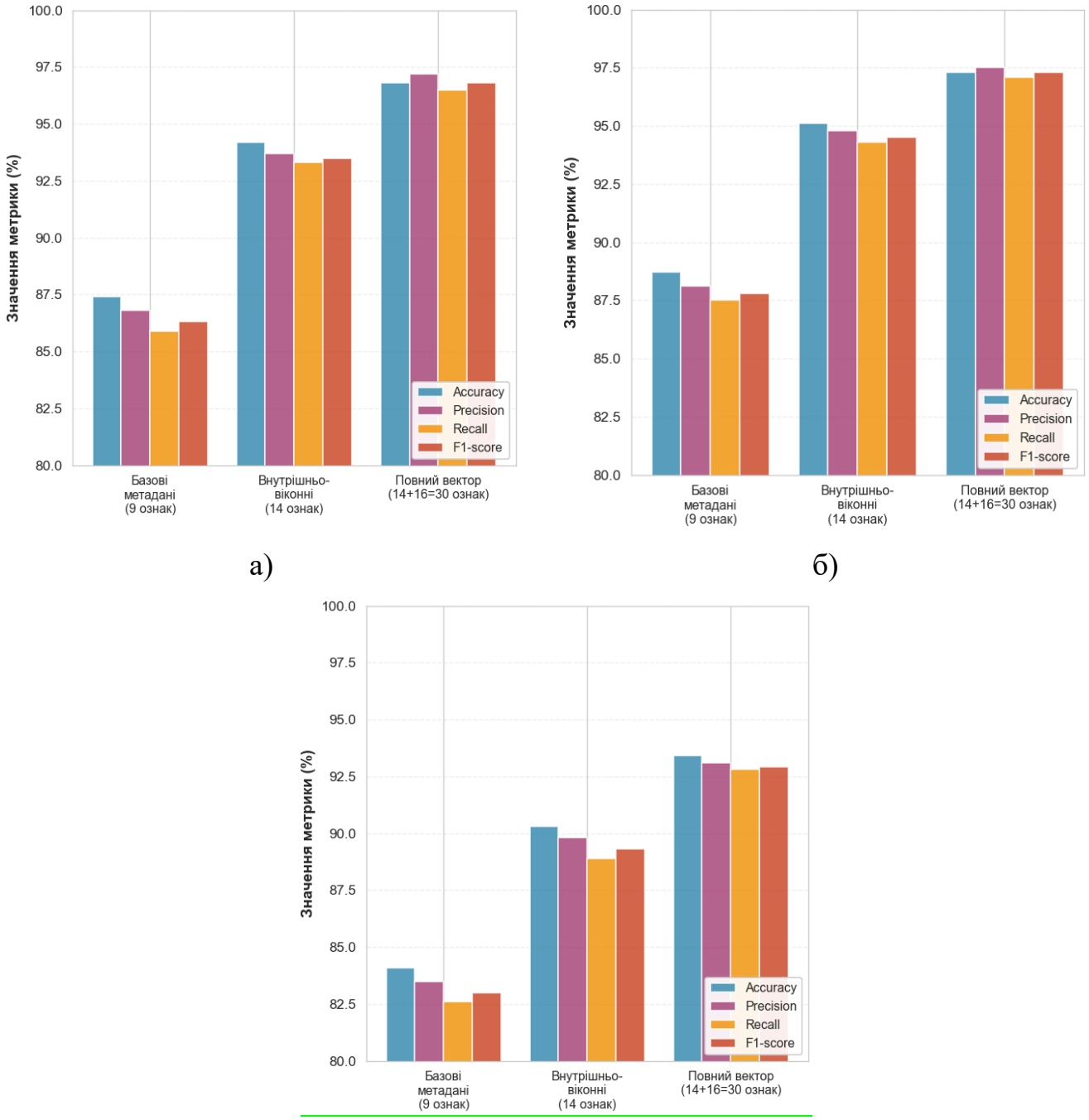


Рисунок 4.5 – Метрики якості класифікації для алгоритмів машинного навчання: а) Random Forest, б) Gradient Boosting, в) SVM

Використання повного вектору статистичних показників дозволило досягти F-міри 96.8%, точності класифікації 96.8%, точності виявлення 97.2% та повноти 96.5%, що демонструє додатковий приріст близько трьох відсоткових пунктів порівняно з конфігурацією лише внутрішньо-віконних ознак. Градієнтний бустинг продемонстрував найвищу ефективність серед досліджуваних алгоритмів за всіх конфігурацій векторів ознак. Для базових метаданих пакетів F-міра становила 87.8%,

точність класифікації 88.7%, точність виявлення 88.1% та повнота 87.5%. Використання внутрішньо-віконних статистик підвищило показники до 94.5%, 95.1%, 94.8% та 94.3% відповідно. Метод опорних векторів показав найнижчу ефективність серед трьох досліджуваних підходів, проте також продемонстрував суттєве покращення при використанні запропонованих статистичних ознак. Для базових метаданих F-міра становила 83.0%, точність класифікації 84.1%, точність виявлення 83.5% та повнота 82.6%. Внутрішньо-віконні статистики підвищили показники до 89.3%, 90.3%, 89.8% та 88.9% відповідно.

Також, було проведено ROC аналіз, криві якого для різних наборів даних наведено на рис. 4.6. Для методу випадкового лісу площа під ROC-кривою зросла з 0.901 при використанні базових метаданих до 0.956 при застосуванні внутрішньо-віконних статистик та до 0.987 при використанні повного вектору показників. Градієнтний бустинг продемонстрував аналогічну динаміку із збільшенням площі під кривою з 0.918 до 0.968 та до 0.992 відповідно. Метод опорних векторів показав зростання з 0.862 до 0.925 та до 0.958.

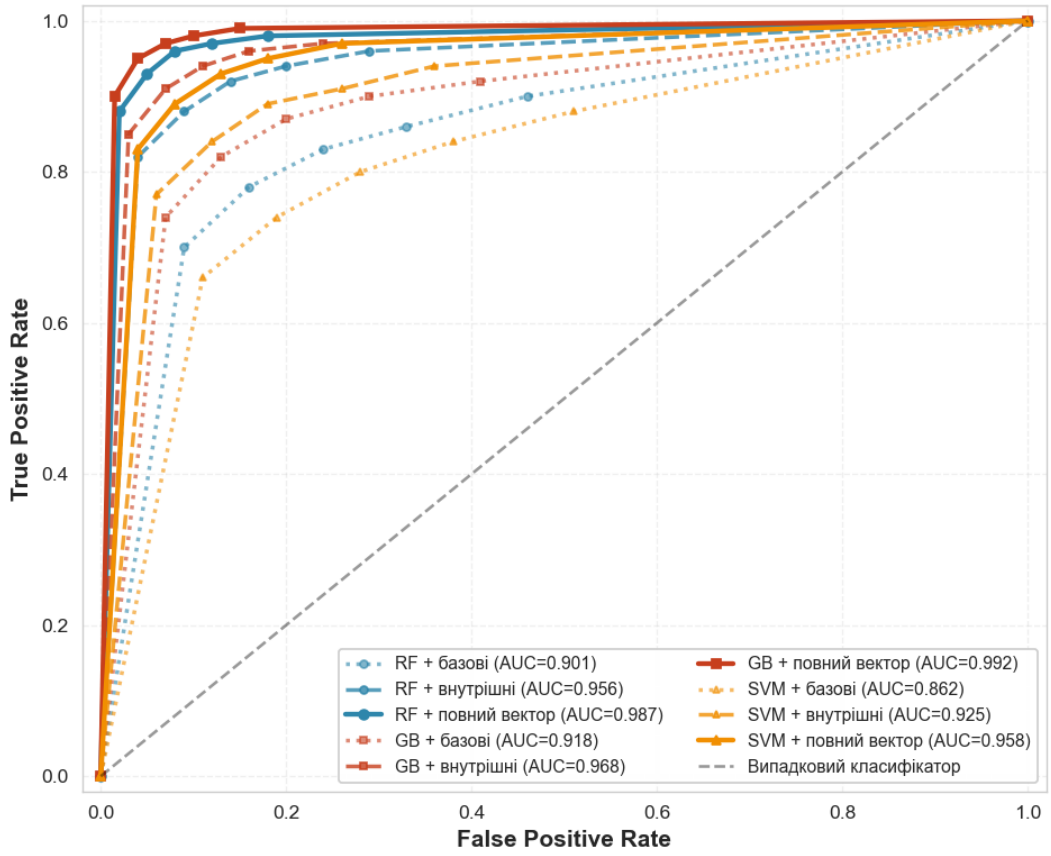


Рисунок 4.6 – ROC-криві для порівняння конфігурацій векторів ознак

Таким чином, експериментальне дослідження підтвердило ефективність запропонованого методу формування статистичних ознак мережного трафіку для виявлення атак типу відмова в обслуговуванні. Встановлено, що використання повного вектору із тридцятьма статистичними показниками, який включає чотирнадцять внутрішньо-віконних та шістнадцять між-віконних ознак, дозволяє підвищити F-міру класифікації на 9-10 відсоткових пунктів порівняно з базовими метаданими пакетів при помірному збільшенні обчислювальних витрат. Найкращі результати досягаються при використанні алгоритму градієнтного бустингу із F-мірою 97.3% та площею під ROC-кривою 0.992.

Таким чином, було досліджено виявлення КА типу відмова в обслуговуванні в корпоративних мережах на основі статистичних показників мережного трафіку. Запропонований метод базується на формуванні комплексного вектору статистичних показників, який включає тридцять ознак, що характеризують поведінку мережного трафіку на різних рівнях аналізу. До складу вектору входять чотирнадцять внутрішньо-віконних статистичних показників, які відстежують параметри активності для кожної IP-адреси протягом визначеного інтервалу часу, та шістнадцять між-віконних статистичних метрик, які фіксують зміни характеристик трафіку між послідовними часовими вікнами. Такий підхід дозволяє виявляти не лише миттєві аномалії в мережній активності, але й відстежувати еволюцію поведінки трафіку протягом тривалого періоду спостереження.

Експериментальне дослідження на наборі даних CIC-IDS2017 підтвердило ефективність запропонованого методу. Використання повного вектору статистичних показників у поєднанні з алгоритмом градієнтного бустингу дозволило досягти F-міри класифікації 97.3%, точності виявлення атак 97.5% та повноти 97.1%, що перевищує результати використання базових метаданих пакетів на 9.5 відсоткових пунктів. Площа під ROC-кривою становила 0.992, що свідчить про високу стабільність роботи методу за різних порогових значень класифікації. Час обробки одного мережного пакету не перевищував 2.8 мілісекунд, що є прийнятним для застосування в системах захисту корпоративних мереж з високою інтенсивністю трафіку.

Результати порівняльного аналізу продемонстрували, що врахування між-віконних статистичних показників забезпечує додатковий приріст ефективності на

2.8-3.6 % порівняно з використанням лише внутрішньо-віконних ознак. Особливо значущими виявилися коефіцієнти внутрішньо-віконної та між-віконної активності хоста, які дозволяють відрізнити постійну активність ЗПЗ від збільшення обсягу легітимного трафіку. Це підтверджує обґрунтованість запропонованого підходу до формування комплексного вектору статистичних показників для виявлення атак відмови в обслуговуванні.

Третій експеримент. В архітектурі обманних систем реалізовано різнотипні обманні технології. Розглянемо експериментальні дослідження щодо зміни сервера. Для дослідження алгоритму зміни сервера в розподіленій системі було розгорнуто експериментальне середовище на базі програмного емулятора Cisco Packet Tracer, що відтворює типову інфраструктуру корпоративної мережі середнього підприємства з підвищеними вимогами до безпеки. Повна інфраструктура організації налічує 275 обчислювальних компонентів, розподілених між двадцятьма п'ятьма віртуальними локальними мережами з адресацією 10.0.X.0/24, де X відповідає номеру VLAN від 1 до 25. Для цілей експериментального дослідження було обрано репрезентативний фрагмент інфраструктури, що включає три віртуальні локальні мережі (VLAN 7, 8, 9) та демілітаризовану зону, які демонструють типові сценарії міграції серверних компонентів у сегментованій корпоративній мережі.

Досліджувана топологія мережі складалась із чотирьох комутаторів другого рівня моделі Cisco Catalyst 2960-24TT, об'єднаних у ієрархічну структуру типу core-distribution, маршрутизатора Cisco ISR4321 для забезпечення міжмережної взаємодії між віртуальними локальними мережами, та міжмережного екрана Cisco ASA 5505 для сегментації демілітаризованої зони від внутрішніх мереж.

Мережна інфраструктура експериментального сегмента була організована відповідно до функціонального призначення та політик інформаційної безпеки: VLAN 7 призначено для управлінських завдань з адресним простором 10.0.7.0/24 та містить десять робочих станцій з адресами 10.0.7.10-10.0.7.19 (позначені як PC0-PC9), VLAN 8 виділено для продуктивних систем з адресацією 10.0.8.0/24 та включає десять робочих станцій з адресами 10.0.8.10-10.0.8.19 (PC10-PC19), VLAN 9 зарезервовано для середовища розробки з діапазоном адрес 10.0.9.0/24 також десятьма робочими станціями з адресами 10.0.9.10-10.0.9.19 (PC20-PC29). Демілітаризована зона DMZ використовує адресний простір 192.168.40.0/24 для

розміщення публічних серверів, включаючи HTTP-сервер, поштовий сервер та зовнішній FTP-сервер. Серверна інфраструктура також включає внутрішні сервери DHCP та Internal FTP, розташовані поза DMZ для обслуговування внутрішніх потреб організації. Досліджувану мережну інфраструктуру, у якій здійснювалось проведення експериментів, подано на рис. 4.7.

Початкове розміщення серверного вузла було здійснено на робочій станції PC18 з IP-адресою 10.0.8.18, що належав до сегмента VLAN 8. Вибір даного вузла обумовлений його центральним розташуванням у топології та типовим навантаженням, характерним для корпоративних розподілених застосунків.

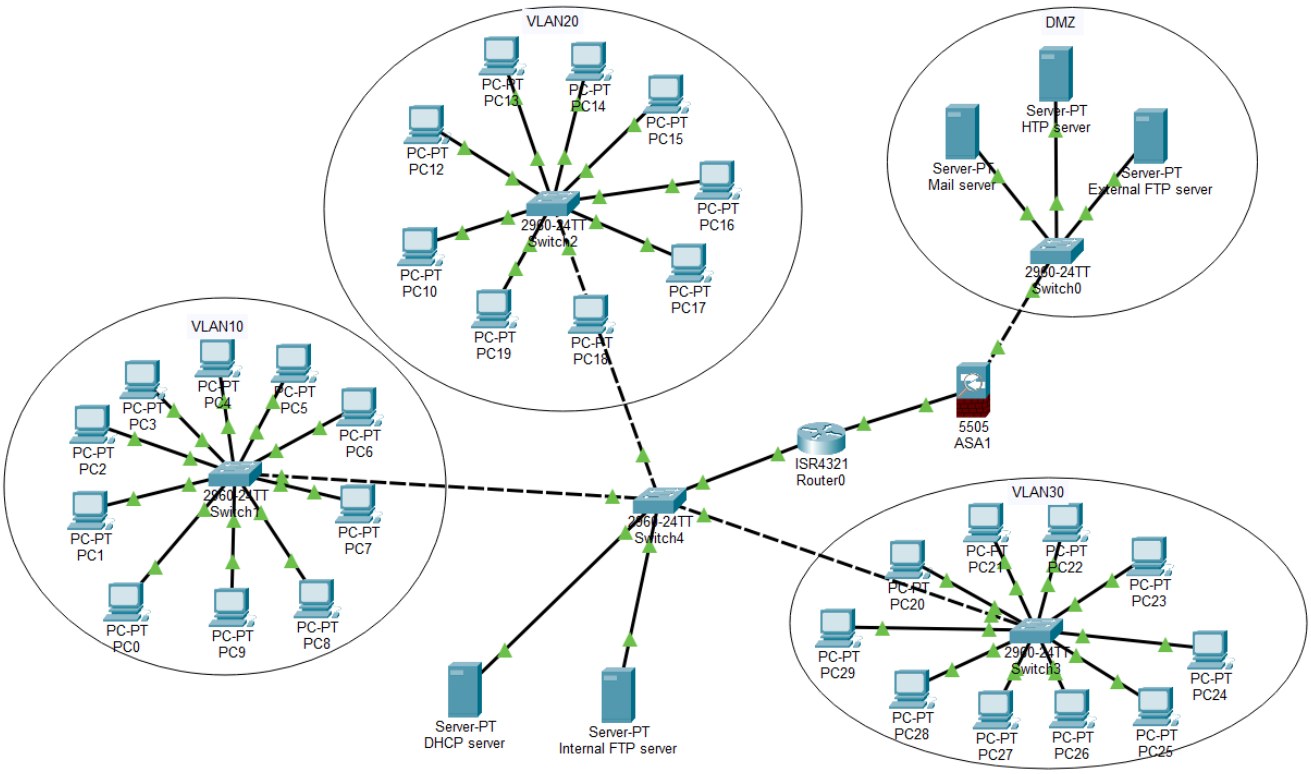


Рисунок 4.7 – Мережна інфраструктура для проведення експериментів

Для кількісної оцінки мережних характеристик між серверним вузлом PC18 та двадцятьма дев'ятьма іншими робочими станціями в експериментальному сегменті мережі було проведено серію систематичних вимірювань з використанням стандартних діагностичних утиліт. Метрика затримки Round Trip Time визначалась за допомогою протоколу ICMP через надсилання послідовності з десяти пакетів розміром 32 байти до кожного цільового вузла з подальшим усередненням отриманих значень часу відгуку. Кількість проміжних вузлів на шляху передачі даних встановлювалась методом трасування маршруту з реєстрацією кожного

маршрутизатора, міжмережного екрана або комутатора третього рівня, через які проходить пакет від джерела до призначення.

З метою симуляції типових затримок корпоративної мережі на інтерфейсах міжсегментних з'єднань було застосовано обмеження пропускної здатності шляхом зміни режиму дуплексу з повного (full-duplex) на напівдуплексний (half-duplex), що призводить до збільшення часу передачі даних через необхідність арбітражу доступу до середовища. Додатково на магістральних trunk-портах між комутаторами Switch1, Switch2 та Switch4 швидкість інтерфейсів було знижено зі стандартних 1 Гбіт/с до 100 Мбіт/с, що імітує застаріле обладнання або перевантажені канали в реальних умовах експлуатації. На інтерфейсах міжмережного екрана Cisco ASA 5505 було введено додаткові затримки на рівні 2-3 мілісекунди для моделювання процесів глибокої інспекції пакетів та обробки правил безпеки, характерних для з'єднань через DMZ.

Результати вимірювань продемонстрували очікувану залежність між топологічною віддаленістю вузлів та значеннями мережних метрик. Для вузлів у межах одного сегмента VLAN 8 середній час відгуку становив від 0.6 до 1.2 мілісекунди при кількості хопів від одного до двох, що відповідає прямій комутації на рівні L2. Міжсегментні з'єднання до вузлів VLAN 7 та VLAN 9 характеризувались RTT у діапазоні 8.5-12.8 мілісекунди та п'ятьма хопами, що обумовлено необхідністю проходження через розподільний комутатор, центральний комутатор Switch4, маршрутизатор ISR4321 та комутатор цільового сегмента. З'єднання до вузлів у демілітаризованій зоні DMZ демонструвало найвищі значення RTT у діапазоні 14.2-18.5 мілісекунди при шести-семи хопах через обов'язкове проходження через міжмережний екран ASA 5505 з додатковою інспекцією трафіку.

Для формалізації задачі вибору оптимального цільового вузла зміни сервера у розподіленій системі визначимо комбіновану метрику відстані, що враховує як часові характеристики мережного з'єднання, так і топологічну складність маршруту, так:

$$D = \alpha \cdot \left(\frac{RTT}{RTT_{max}}\right) + \beta \cdot \left(\frac{Hops}{Hops_{max}}\right), \quad (4.24)$$

де D - нормалізована відстань від поточного серверного вузла до потенційного кандидата на зміну; RTT - виміряний середній час кругового обігу пакета в мілісекундах; $Hops$ - кількість транзитних вузлів на маршруті; RTT_{max} та $Hops_{max}$ - максимальні значеннями відповідних параметрів у межах досліджуваної мережі для

забезпечення нормалізації; α та β – коефіцієнти, що визначають вагові внески кожної компоненти у підсумкову метрику з умовою $\alpha + \beta = 1$.

Було обрано $\alpha = 0{,}7$ та $\beta = 0{,}3$, що надає пріоритет часовим характеристикам передачі даних над топологічною відстанню, оскільки для більшості розподілених застосунків саме латентність виступає критичним фактором продуктивності.

Було досліджено (рис. 4.8) два альтернативних варіанти вибору цільового вузла для зміни серверного компонента. Перший варіант передбачав пошук глобального оптимуму шляхом мінімізації метрики D серед усіх доступних вузлів мережі, незалежно від їх сегментної приналежності. За результатами обчислень найменше значення метрики $D = 0{,}158$ було зафіксовано для вузла PC12 з IP-адресою 10.0.8.12, який знаходиться в тому самому сегменті VLAN 8, що й початковий сервер PC18. Така конфігурація забезпечувала RTT на рівні 0.8 мілісекунди та потребувала лише двох хопів для досягнення, що робить цей варіант оптимальним з точки зору мінімізації мережних додаткових витрат.

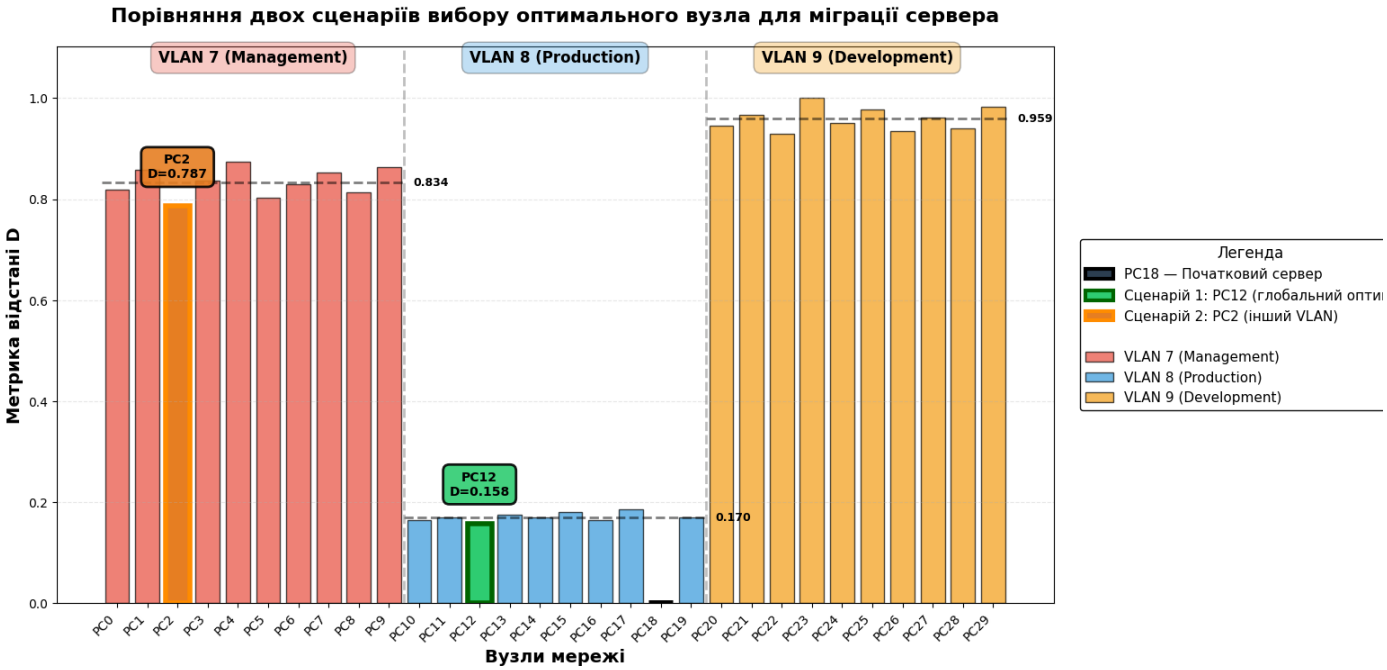


Рисунок 4.8 – Порівняння двох сценаріїв

В другому варіанті розглянуто ситуацію, при якій зміна мала відбутися до альтернативного сегмента внутрішньої мережі, виходячи з балансування навантаження або диверсифікації ризиків відмови. У цьому випадку множина потенційних кандидатів обмежувалася вузлами VLAN 7 та VLAN 9, виключаючи

поточний сегмент VLAN 8. Застосування критерію мінімізації до цієї обмеженої множини виявило вузол PC5 з адресою 10.0.7.15 як оптимальний вибір з метрикою $D = 0{,}787$. Цей вузол належить до управлінського сегмента VLAN 7 і характеризувався RTT у 9.2 мілісекунди та п'ятьма хопами, що є найкращим показником серед міжсегментних з'єднань у межах внутрішньої мережі.

Аналіз експериментальних результатів демонструє чітку диференціацію мережних метрик між сегментами віртуальних локальних мереж. Середня відстань до вузлів VLAN 7 становить 0.834, до VLAN 8 – 0.170 (без урахування самого PC18), а до VLAN 9 – 0.959, що відображає суттєві відмінності у мережній доступності між сегментами. Виявлено, що внутрішньосегментна міграція забезпечує п'ятикратне зменшення метрики відстані порівняно з міжсегментною (0.158 проти 0.787), що підтверджує критичну важливість топологічного розміщення серверних компонентів для мінімізації латентності. Теплову карта мережних метрик для оцінки зміни сервера наведено на рис. 4.9. Запропонований підхід на основі комбінованої метрики дає змогу систематизувати процес вибору цільового вузла, забезпечуючи баланс між оптимізацією продуктивності через мінімізацію мережних затримок та стратегічними вимогами до розподілу навантаження і відмовостійкості системи шляхом міжсегментної диверсифікації.

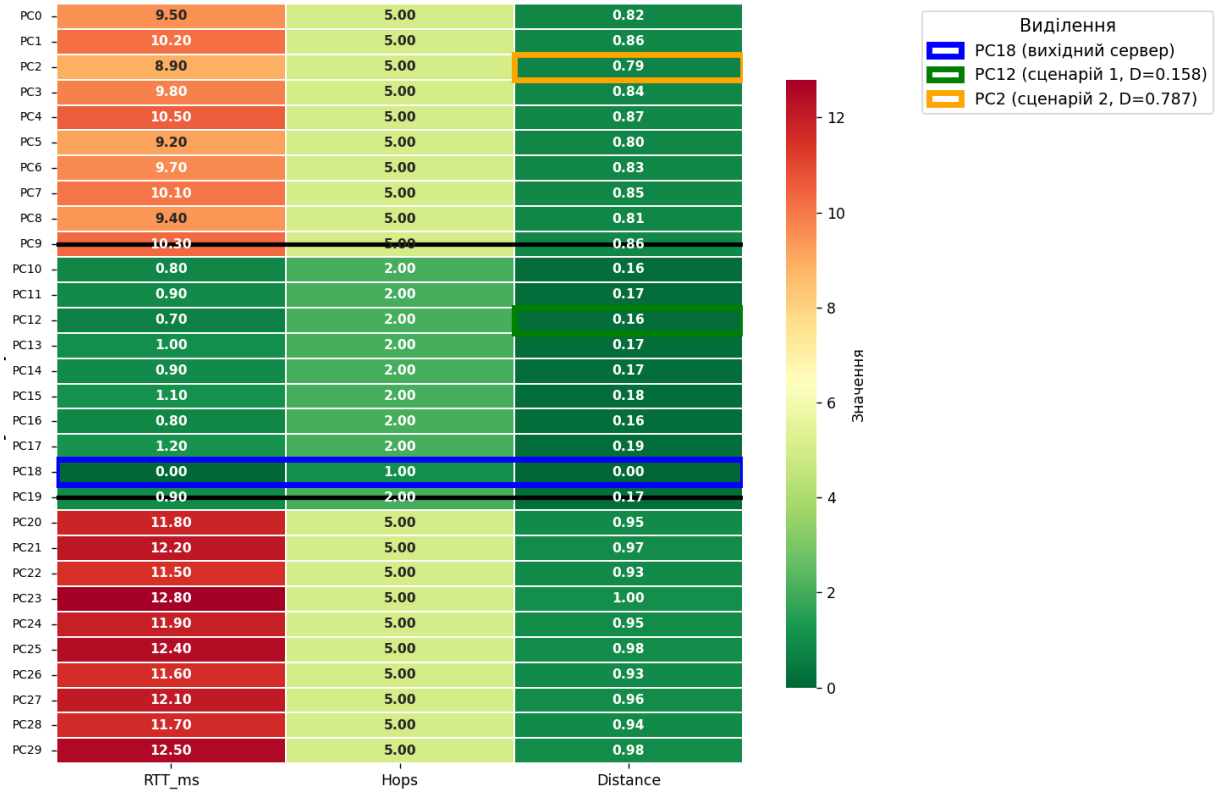


Рисунок 4.9 – Теплова карта мережних метрик для оцінки зміни сервера

Четвертий експеримент. Було проведено експеримент із перевірки спроможності розробленої системи виявляти КА чи дії ЗПЗ. Для цього було здійснено впливи двоцільовими КА та ЗПЗ на певні вузли корпоративних мереж.

Обманна система згідно результатів експерименту, які подано в табл. 4.8, адекватно реагувала на КА чи дії ЗПЗ, що підтверджено відсотком правильних спрацювань.

Таблиця 4.8 - Поле подій та результати

№ з/п	Варіанти подій	Зловмисник здійснив одну двоцільову КА	Зловмисник здійснив більше однієї різних двоцільових КА	ЗПЗ здійснило поширення в одній комп'ютерній станції	ЗПЗ здійснило поширення в багатьох комп'ютерних станціях
1	КА чи ЗПЗ завершились в приманках чи пастках;	67	62	6	11
2	КА чи ЗПЗ потрапили в приманки чи пастки, але при цьому їх активність продовжилась на інших напрямах;	8	17	2	2
3	КА чи ЗПЗ надали змогу зловмисникам отримати контроль над приманками чи пастками, з якими увійшли в контакт	4	3	1	3
4	КА чи ЗПЗ досягли мети зловмисників і не були помічені обманними системами з приманками і пастками	17	12	0	1
5	КА чи ЗПЗ не досягли мети зловмисників і не були помічені обманними системами з приманками і пастками	4	6	0	1
Разом:		100	100	9	18

За результатами експерименту в ізольованому середовищі корпоративних мереж встановлено, що 67% двоцільових КА завершились в приманках чи пастках, 8% двоцільових КА потрапили в приманки чи пастки, але при цьому їх активність продовжилась на інших напрямах, 4% двоцільових КА надали змогу зловмисникам

отримати контроль над приманками чи пастками після здійснення з ними контакту, 17% двоцільових КА досягли мети зловмисників і не були помічені обманними системами з приманками і пастками і 4% двоцільових КА не досягли мети зловмисників і не були помічені обманними системами з приманками і пастками. Якщо ж зловмисник здійснив більше однієї різних двоцільових КА, що було забезпечено штучними КА в ізольованому середовищі корпоративних мереж, то 62% двоцільових КА завершились в приманках чи пастках, 17% двоцільових КА потрапили в приманки чи пастки, але при цьому їх активність продовжилась на інших напрямах, 3% двоцільових КА надали змогу зловмисникам отримати контроль над приманками чи пастками після здійснення з ними контакту, 12% двоцільових КА досягли мети зловмисників і не були помічені обманними системами з приманками і пастками і 6% двоцільових КА не досягли мети зловмисників і не були помічені обманними системами з приманками і пастками. Таким чином, здійснення декількох типів двоцільових КА порівняно з однією двоцільовою КА в корпоративній мережі не збільшило суттєво ефективність виявлення. Крім того, поширення ЗПЗ було помічено обманною системою. Для підвищення відсотку виявлення необхідно доповнити систему методами виявлення КА та ЗПЗ.

Запропонована комбінована метрика враховує як часові характеристики з'єднань, так і топологічну віддаленість вузлів, забезпечуючи адаптивність рішень через налаштування вагових коефіцієнтів відповідно до поточних потреб системи, що дозволяє визначити оптимальний варіант зміни сервера у розподілених обманних системах.

П'ятий експеримент. Метою експерименту є кількісна оцінка ефективності алгоритму молі і полум'я (MFO) при керуванні ОСПП та порівняння його з альтернативними метаевристичними алгоритмами.

Алгоритм молі і полум'я розроблено з фокусом на дискретний простір пошуку. При цьому, потрібно оцінити не тільки його коректне виконання і здійснити порівняння з іншими подібними рішеннями. Потрібно оцінити його ефективність для визначення наступних кроків обманних систем, тобто стратегії дій під час КА. Таким чином, дослідження щодо алгоритму молі і полум'я повинно стосуватись не тільки класичної оптимізації функції, а того як згідно нього відбувається вибір та зміна наступних кроків обманних систем під час КА, вплив на поведінку зловмисників,

уникнення збіжності в локальних оптимумах та функціонування обманних систем в динамічно змінюваному середовищі корпоративних мереж під час здійснення КА. Тому, для здійснення оцінювання синтезу алгоритму молі і полум'я в архітектурі обманних систем недостатньо використовувати стандартні метрики оптимізації, потрібно розробити багатокритеріальну функцію оцінювання та застосувати її для цього алгоритму та інших алгоритмів, які можуть бути синтезовані в архітектурі обманних систем з приманками і пастками і виконувати завдання з вибору наступних кроків систем.

Задамо простір рішень множиною $M_{PR}^{E} = \left\{m_{PR,1}^{E}, m_{PR,2}^{E}, \dots, m_{PR,N_{M_{PR}^{E}}}^{E}\right\}$, де $N_{M_{PR}^{E}}$ – кількість елементів в множині M_{PR}^{E}, які формують простір рішень. Визначимо елементи множини M_{PR}^{E} так: $m_{PR,1}^{E}$ – конфігурації приманок і пасток в корпоративних мережах; $m_{PR,2}^{E}$ – увімкнення та активізація приманок і пасток; $m_{PR,3}^{E}$ – черговість кроків обманних систем та час реакції; $m_{PR,4}^{E}$ – міграція зловмисників між вузлами корпоративних мереж.

Оскільки, для обманних систем з приманками і пастками важливим є рівень дезорієнтації зловмисника, час утримання зловмисника в обманному середовищі, витрати ресурсів, а також проблеми у випадку розкриття обманних об'єктів зі сторони зловмисника, то введемо цільову функцію за адитивним критерієм для оцінювання ефективності різних алгоритмів, включно з алгоритмом молі і полум'я, які синтезовано в архітектурі обманних систем з приманками і пастками для вибору наступних кроків під час КА, так:

$$F_1^E(A) = \sum_{i=1}^{N_{F_1^E}} \alpha_i * q_i, \tag{4.25}$$

де A – номер алгоритму, для якого здійснюється оцінювання; $N_{F_1^E}$ – кількість показників, які враховуються при оцінюванні; α_i – ваговий коефіцієнт для оцінювання ваги кожного i – того показника в функції $F_1^E(A)$; q_i – значення i – того показника; $i = 1,2, \dots, N_{F_1^E}$; $\sum_{i=1}^{N_{F_1^E}} \alpha_i = 1$; $q_i \in [0,1]$.

Згідно формули (1) приймемо значення показників так: q_1 – рівень дезорієнтації зловмисника; q_2 – відсоток часу утримання зловмисника в обманному середовищі від

загального часу протягом якого тривала КА; q_3 – вартість використаних ресурсів; q_4 – відсоток виявлених зловмисником обманних об'єктів в корпоративних мережах. Для встановлення кількісних значень показників з формули (4.25) введемо метрики, за якими вони будуть визначені.

Поведінку зловмисників визначимо з урахуванням таких кількісних показників:

1) $t_{p,1}^E$ – середній час встановлення зловмисником факту здійснення атаки саме на приманку чи пастку;

2) $n_{p,2}^E$ – кількість кроків зловмисника, які він зробив у обманному середовищі корпоративних мереж;

3) $q_{p,3}^E$ – частка повторюваних дій, тобто ознака дезорієнтації, від всіх дій;

4) $t_{p,4}^E$ – час, протягом якого зловмисник діє, вважаючи середовище корпоративних мереж та об'кти взаємодії реальними.

Показники щодо оптимізації алгоритмів визначимо так:

1) $w_{o,1}^E$ – середня швидкість знаходження ефективних стратегій, тобто вибору наступних кроків обманних систем;

2) $n_{o,2}^E$ – середня кількість різних варіантів вибору кроків, тобто варіантів з уникнення збіжності в локальному оптимумі;

3) $q_{o,3}^E$ – частка обраних кроків від всіх розглядуваних кроків обманних систем;

4) $n_{o,4}^E$ – частка обраних повторюваних наступних кроків з усіх кроків, які розглядались, тобто частота перебування в окремому локальному оптимумі.

Середня кількість різних варіантів вибору кроків, тобто варіантів з уникнення збіжності в локальному оптимумі, та частка обраних повторюваних наступних кроків з усіх кроків, які розглядались, тобто частота перебування в окремому локальному оптимумі, є критичними в контексті оцінювання популяційних алгоритмів, зокрема алгоритму молі і полум'я, та їх порівняння між собою.

Для оцінювання безпеки корпоративних мереж з наявними обманними системами, які також керують приманками і пастками, введемо такі показники:

1) $n_{s,1}^E$ – кількість реальних об'єктів в корпоративних мережах, які не помітив зловмисник в процесі КА;

2) $t_{s,2}^{E}$ – час до виявлення КА;

3) $q_{s,3}^{E}$ – відсоток КА, які були спрямовані в пастки та на приманки.

Для порівняння алгоритму молі і полум'я з іншими популяційними алгоритмами виберемо наступні алгоритми.

Метод рою частинок PSO (Particle Swarm Optimization), особливістю якого є те, що для знаходження найкращих рішень кожна частинка рухається в просторі рішень, коригуючи свою швидкість та напрямок на основі власного досвіду та найкращого досвіду інших частинок у рої, що дозволяє системі поступово сходитися до оптимального значення. Він підходить для розглядуваного завдання, тобто для вибору наступних кроків обманних систем, бо має швидку збіжність, а частинки мають два параметри, які характеризують напрям та швидкість, що підходить для керування приманками і пастками.

Генетичний алгоритм GA (Genetic Algorithm) забезпечує належну глобальну оптимізацію і може бути використаний в обманних системах з приманками і пастками, оскільки вибір наступних кроків може відбуватись на основі модифікації їх послідовності, що є допустимим.

Мурашинні алгоритми ACO (Ant Colony) мають особливість, яка базується на їх використанні при наявності шляхів, що підходить для адаптації в архітектурі обманних систем з приманками і пастками.

Метод багатовимірної математичної оптимізації DE (Differential Evolution) відноситься до класу стохастичних алгоритмів оптимізації, тобто працює з використанням випадкових чисел, і використовує деякі ідеї генетичних алгоритмів.

Маши́нне навча́ння на осно́ві пра́вил Random / Rule-based може бути використаний, бо визначальною характеристикою є те, що відбувається навчається на основі правил.

Для коректності порівняння було імплементовано в архітектурі обманних систем окремо всі алгоритми. Також, при проведенні експерименту було забезпечено однакові сценарії атак, однакові ресурси, однакові обмеження за часом та велику кількість серій повторів.

Отримання статичної оцінки буде базуватись на середньому значенні цільової функції $F_1^E(A)$ (формула (4.25)) для кожного алгоритму A, дисперсії для значень цільової функції $F_1^E(A)$, підтримання тривалості протидії КА протягом часу її здійснення.

Середовище корпоративної мережі для проведення експерименту містить 100 вузлів, з яких 20 реальних і 80 хибних об'єктів. Було імітовано КА їх етапами: розвідка; підвищення привілеїв; горизонтальне переміщення. При проведені розвідки здійснювався збір інформації про IP-адреси та використовуване програмне забезпечення. Для підвищення привілеїв в системі базувався на отриманні ролі звичайно користувача. Реалізація горизонтального переміщення полягала в спробах переміщення від одного вузла до іншого. Модель зловмисника базувалась на реалізації випадкових процесів з частковим навчанням. Для порівняння вибрано проаналізовані алгоритми, які подано в табл. 4.9.

Таблиця 4.9 - Порівнювані алгоритми

Алгоритм	Призначення
MFO	базовий
PSO	швидка збіжність
GA	глобальна оптимізація
ACO	маршрутні атаки
Random	базовий рівень

Таблиця 4.10 - Результати

Алгоритм	Значення цільової функції	Середньо-квадратичне відхилення	Час підтримки протидії КА	Виявлення зловмисником реальних об'єктів
MFO	**0.87**	**0.03**	78	**1.2**
PSO	0.81	0.05	**45**	2.7
GA	0.83	0.04	92	1.9
ACO	0.79	0.06	110	3.1
Random	0.62	0.08	—	5.6

Таким чином, алгоритм молі і полум'я має спіральну траєкторію пошуку, що забезпечує плавний перехід, а також уникнення передчасної збіжності в локальному оптимумі. Завдяки його використанню при виборі наступних кроків ОСПП здійснення вибору не обмежується одним рішенням, відбувається зміна в розміщенні

об’єктів в корпоративних мережах, заплутує зловмисника своєю поведінкою, а не мінімізує шлях, рідше збігається в локальному оптимумі. Також, цей алгоритм підтримує багатоваріантість та є ефективніший при тривалому обмані і на відміну від інших не забезпечує швидку збіжність та є адаптивнішим до нестаціонарної поведінки КА.

Алгоритм PSO є швидко збіжним, що може призвести до вичерпування варіантів з вибором кроків при тривалій КА, яка може продовжуватись. Генетичний алгоритм GA є ресурсомістким порівняно з алгоритмом молі і полум’я. Мурашинні алгоритми ACO орієнтовані на пошук шляху, а не стратегії. Алгоритм на основі пра́вил Random не дає змогу змінювати кроки обманних систем в процесі тривалих КА.

Для адаптивного зловмисника з обмеженою пам’яттю та частковою видимістю, вибір алгоритму молі і полум’я має нижчу ймовірність розкриття обману, ніж стратегії, оптимізовані алгоритмами з швидкою збіжністю, бо адаптивний агент потребує стаціонарності, а згідно алгоритму така стаціонарність порушується.

Застосування алгоритму молі і полум’я в архітектурі обманних систем згідно проведених досліджень з розробленою системою підтверджує підвищення ефективності функціонування таких систем та виявлення КА в корпоративних мережах.

4.3 Висновки до четвертого розділу

Розроблено метод виявлення атак типу відмова в обслуговуванні у мережах на основі статистичних показників з урахуванням його імплементації в компоненти ОСПП для підвищення достовірності виявлення атак відмова в обслуговуванні.

Розроблено ОСПП для виявлення двоцільових КА, зокрема КА типу відмова в обслуговуванні у мережах на основі статистичних показників, для експериментального дослідження таких систем з метою покращення її характеристик, здійснено постановку експерименту, проведено експериментальні дослідження, які дозволили встановити покращення її характеристик.

Розроблена ОСПП керує корпоративною мережею та приманками і пастками, обираючи кроки за алгоритмом молі й полум’я. У межах кожного впливу КА відбувалося кілька перебудов. Зміни тривали до завершення впливів, що підтвердило коректність вибору кроків і досягнення глобального оптимуму.

Результати експериментів щодо виявлення КА: F1 – 97.3%; точність – 97.5%; повнота – 97.1% (+9.5 п.п.); AUC – 0.992; час – ≤ 2.8 мс/пакет. Між-віконні ознаки дали +2.8–3.6% до ефективності. Ключовими стали показники активності для виявлення DoS.

Аналіз VLAN виявив різну доступність: 0.834 (VLAN 7); 0.170 (VLAN 8, без PC18); 0.959 (VLAN 9). Внутрішньосегментна міграція зменшує відстань у 5 разів (0.158 проти 0.787). Комбінована метрика (час + топологія) забезпечує баланс продуктивності та відмовостійкості.

В ізольованому середовищі одна КА дала 67% – у пастках, 8% – з продовженням, 4% – контроль, 17% – успіх без виявлення, 4% – невдача без виявлення. За кількох КА відповідні результати – 62%, 17%, 3%, 12%, 6%. Таким чином, одночасність здійснення декількох КА суттєво не зменшили показники виявлення порівняно з однією КА.

Алгоритм молі й полум’я завдяки спіральному пошуку уникає локальних оптимумів і підтримує тривалий обман. На відміну від PSO, GA, ACO та Random, він менш передбачуваний для адаптивного зловмисника й підвищує ефективність виявлення КА та дій ЗПЗ в корпоративних мережах.

Основні наукові результати розділу опубліковані в працях [108; 110; 113; 136; 168; 181].

ВИСНОВКИ

У результаті виконання дисертаційного дослідження було розв’язано актуальну науково-прикладну задачу покращення протидії КА та ЗПЗ в корпоративних мережах шляхом оптимізації кроків ОСПП за рахунок синтезу популяційних алгоритмів в центрах прийняття рішень.

У роботі отримано наступні наукові та практичні результати.

1. Здійснено аналіз та систематизацію типів комп’ютерних атак та відповідного зловмисного програмного забезпечення, методів та систем їх виявлення в корпоративних мережах, зокрема обманних систем, приманок та пасток, а також популяційних алгоритмів натхненних живою природою, які стали основою для їх застосування в архітектурі обманних систем для підвищення ефективності вибору ними наступних кроків та їх коригування.

2. Розроблено моделі комп’ютерних атак в корпоративних мережах, в яких врахована особливість щодо наявних в корпоративних мережах реальних та хибних об’єктів для атак та відповідних дій зловмисників, які спрямовані на класифікацію таких об’єктів, що дозволили врахувати їх поведінку в моделі функціонування обманних систем з приманками і пастками.

3. Розроблено архітектуру обманних систем з приманками і пастками з підсистемою прийняття рішень на основі синтезу в архітектурі популяційних алгоритмів, зокрема алгоритму молі і полум’я, для оптимізації формування послідовності наступних кроків та їх коригування при здійсненні КА та дій ЗПЗ.

4. Розроблено метод синтезу алгоритму дискретної оптимізації молі й полум’я в архітектурі обманних систем з приманками і пастками для забезпечення їх довготривалого й адаптивного функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій.

5. Розроблено метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах на основі синтезу в їх архітектурі популяційних алгоритмів, зокрема алгоритм молі і полум’я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак.

6. Розроблено метод виявлення атак типу відмова в обслуговуванні у мережах на основі статистичних показників з урахуванням його імплементації в компоненти

обманних систем з приманками і пастками для підвищення достовірності виявлення атак відмова в обслуговуванні.

7. Розроблено обманну систему з приманками і пастками для виявлення двоцільових КА, зокрема КА типу відмова в обслуговуванні у мережах на основі статистичних показників, для експериментального дослідження таких систем з метою покращення її характеристик, здійснено постановку експерименту, проведено експериментальні дослідження, які дозволили встановити покращення її характеристик.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Honeypot allocation for cyber deception under uncertainty / A. H. Anwar et al. *IEEE transactions on network and service management*. 2022. P. 1. URL: https://doi.org/10.1109/tnsm.2022.3179965

2. Honeypot-Based cyber deception against malicious reconnaissance via hypergame theory / A. H. Anwar et al. *GLOBECOM 2022 – 2022 IEEE global communications conference*, Rio de Janeiro, Brazil, 4–8 December 2022. 2022. P. 3393–3398. URL: https://doi.org/10.1109/globecom48099.2022.10000813

3. Biswas J. Analysis of client honeypots. *(IJCSIT) international journal of computer science and information technologies*. 2014. Vol. 5, no. 4. P. 5776–5780. URL: https://ijcsit.com/docs/Volume%205/vol5issue04/ijcsit20140504209.pdf

4. Strategic honeypot game model for distributed denial of service attacks in the smart grid / K. Wang et al. *IEEE transactions on smart grid*. 2017. Vol. 8, no. 5. P. 2474–2482. URL: https://doi.org/10.1109/tsg.2017.2670144

5. Viola V. From honeypots to distributed deception platforms: theory and testing of emerging technologies for IT security. 2019. URL: https://api.semanticscholar.org/CorpusID:226826721

6. Gnatyuk S., Sydorenko V., Yudin O., Zhyharevych O., Polozhentsev A. Method for calculating the criticality level of sectoral information and telecommunication systems. *CEUR Workshop Proc*. 2022. Vol. 3347 Pp. 234–245. URL: https://ceur-ws.org/Vol-3347/Paper_20.pdf

7. Svanadze V., Gnatyuk S. Challenges and solutions for cybersecurity and information security management in organizations. *CEUR-WS*. 2024. Vol. 3654. P. 497–504. URL: https://ceur-ws.org/Vol-3654/short20.pdf

8. Дудикевич В., Микитин Г., Ребець А. Квінтесенція інформаційної безпеки кіберфізичної системи. *Вісник Національного університету «Львівська політехніка». Інформаційні системи та мережі*. 2018. № 887. С. 58–68. URL: https://science.lpnu.ua/sites/default/files/journal-paper/2019/feb/15470/180937ism-58-68.pdf

9. An adaptive honeypot deployment algorithm based on learning automata / Y. Zhang et al. *2017 IEEE second international conference on data science in cyberspace*

(DSC), Shenzhen, China, 26–29 June 2017. 2017. URL: https://doi.org/10.1109/dsc.2017.52

10. Nasr M., Zolfaghari H., Houmansadr A. The waterfall of liberty: decoy routing circumvention that resists routing attacks. *Proceedings of the 2017 ACM SIGSAC conference on computer and communications security*. 2017. URL: https://dl.acm.org/doi/pdf/10.1145/3133956.3134075

11. A survey on deception techniques for securing web application / A. I. M. Efendi et al. 2019 IEEE 5th intl conference on big data security on cloud (bigdatasecurity), IEEE intl conference on high performance and smart computing, (HPSC) and IEEE intl conference on intelligent data and security (IDS), Washington, DC, USA, 27–29 May 2019. 2019. URL: https://doi.org/10.1109/bigdatasecurity-hpsc-ids.2019.00066

12. Karpinski M., Kuznetsov O., Oliynykov R. Security, privacy, confidentiality, and trust in the blockchain: from theory to applications. *Electronics*. 2025. Vol. 14, no. 3. P. 581. URL: https://doi.org/10.3390/electronics14030581

13. Deceptive routing in relay networks / A. Clark et al. *Lecture notes in computer science*. Berlin, Heidelberg, 2012. P. 171–185. URL: https://doi.org/10.1007/978-3-642-34266-0_10

14. Campbell R. M., Padayachee K., Masombuka T. A survey of honeypot research: trends and opportunities. *2015 10th international conference for internet technology and secured transactions (ICITST)*, London, United Kingdom, 14–16 December 2015. 2015. URL: https://doi.org/10.1109/icitst.2015.7412090

15. Ткачов В.М., Коваленко А.А., Кучук Г.А., Ні Я.С. Метод забезпечення живучості високомобільної комп'ютерної мережі. *Сучасні інформаційні системи*. Харків: НТУ «ХПІ», 2021. Т. 5., No 2. С. 159-165. DOI: https://doi.org/10.20998/2522-9052.2021.2.24

16. Ткачов В.М., Коваленко А.А., Фесенко Т.Г. Оптимізація мережного алгоритму функціонування комп'ютерних мереж підвищеної живучості на мобільній платформі на етапі їх проектування. *Системи управління, навігації та зв'язку. Збірник наукових праць*. Полтава: ПНТУ, 2021. Т. 3 (65). С. 143-147. DOI: https://doi.org/10.26906/SUNZ.2021.3.143

17. Yevseiev S., Melenti Y., Voitko O., Hrebeniuk V., Korchenko A., Mykus S., Milov O., Prokopenko O., Sievierinov O., Chopenko D. Development of a concept for

building a critical infrastructure facilities security system, *Eastern-European Journal of Enterprise Technologies*, 2021, vol. 3, pp. 63–83. DOI: https://doi.org/10.15587/1729-4061.2021.233533

18. Корченко А.О. Методи ідентифікації аномальних станів для систем виявлення вторгнень: автореф. Дис. … д-ра техн. Наук : 05.13.21. Київ, 2019. 40 с.

19. Lobanchykova N., Pilkevychb I., Korchenko O. Analysis of attacks on components of IoT systems and cybersecurity technologies. *QuaInT+ doors, CEUR-WS*. 2021. Vol. 2850. P. 83–96. URL: https://ceur-ws.org/Vol-2850/paper6.pdf

20. Letychevskyi O., Peschanenko V. Applying algebraic virtual machine to cybersecurity tasks. *2022 IEEE 9th international conference on sciences of electronics, technologies of information and telecommunications (SETIT)*, Hammamet, Tunisia, 28–30 May 2022. 2022. DOI: https://doi.org/10.1109/setit54465.2022.9875895

21. Летичевський О. О. Сучасні наукові проблеми кібербезпеки. *Visnik Nacional noi academii nauk Ukraini*. 2023. № 2. С. 12–20. DOI: https://doi.org/10.15407/visn2023.02.012

22. Mathematical model for heterogeneous databases parameters estimation in distributed systems with dynamic structure / V. Mukhin et al. *2020 IEEE 2nd international conference on advanced trends in information theory (ATIT)*, Kyiv, Ukraine, 25–27 November 2020. 2020. P. 158–161. DOI: https://doi.org/10.1109/atit50783.2020.9349331

23. Mukhin V., Kornaga Y., Zavgorodnii V., Bazaka Y., Zavgorodnya A., Mukhin O., Method of Data Processing System Synthesis for Heterogeneous Distributed Databases Based on Network-Centric Control. *2023 IEEE 12th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, Dortmund, Germany, 2023, pp. 607-612, DOI: https://doi.org/10.1109/IDAACS58523.2023.10348667

24. Secure cyber deception architecture and decoy injection to mitigate the insider threat / K. Park et al. *Symmetry*. 2018. Vol. 10, no. 1. P. 14. DOI: https://doi.org/10.3390/sym10010014

25. Rowe N. C. Honeypot deception tactics. *Autonomous cyber deception*. Cham, 2019. P. 35–45. DOI: https://doi.org/10.1007/978-3-030-02110-8_3

26. Joshi R.C., Sardana A. Science Honeypots. A new Paradigm to Information Security. *Publishers, P.O.* Box 699, Enfi eld, NH 03748, USA, 2011. – 323p. DOI: https://doi.org/10.1201/b10738

27. Savenko O., Sachenko A., Lysenko S., Markowsky G., Vasylkiv N. BOTNET DETECTION APPROACH BASED ON THE DISTRIBUTED SYSTEMS. *International Journal of Computing*, 2020. Vol. 19(2), Pp. 190-198. DOI: https://doi.org/10.47839/ijc.19.2.1761

28. Kashtalian A., Lysenko S., Kysil T., Sachenko A., Savenko O., Savenko, B. Method and Rules for Determining the Next Centralization Option in Multicomputer System Architecture. *International Journal of Computing*. 2025. Vol. 24(1), 35-51. DOI: https://doi.org/10.47839/ijc.24.1.3875

29. Kashtalian A., Ścisło Ł., Rucki R., Lysenko S., Sachenko A., Savenko B., Savenko O., Nicheporuk A. Control and Decision-Making in Deceptive Multi-Computer Systems Based on Previous Experience for Cybersecurity of Critical Infrastructure. *Applied Sciences*, 2025. Vol. 15(22), 12286. DOI: https://doi.org/10.3390/app152212286

30. Kashtalian A., Lysenko S., Sachenko A., Savenko B., Savenko O., Nicheporuk A. Evaluation criteria of centralization options in the architecture of multicomputer systems with traps and baits. *Radioelectronic and Computer Systems*. 2025. Vol. 1. Pp. 264-297. DOI: https://doi.org/10.32620/reks.2025.1.18

31. Kashtalian A., Savenko O., Sachenko A. Agglomerative clustering of data collected by honeypots. *2021 11th IEEE international conference on intelligent data acquisition and advanced computing systems: technology and applications (IDAACS)*, Cracow, Poland, 22–25 September 2021. 2021. DOI: https://doi.org/10.1109/idaacs53288.2021.9661027

32. Kashtalian A., Sochor T. K-Means clustering of honeynet data with unsupervised representation learning. International workshop on intelligent information technologies & systems of information security. *CEUR workshop proceedings*. Vol. 2853. 2021. P. 539–549. URL: https://ceur-ws.org/Vol-2853/paper48.pdf

33. Kashtalian A., Lysenko S., Savenko O., Nicheporuk A., Sochor T., Avsiyevych V. Multi-computer malware detection systems with metamorphic functionality. *Radioelectronic and Computer Systems*, 2024. Vol. 1. Pp. 152-175. DOI: https://doi.org/10.32620/reks.2024.1.13

34. Sochor T., Zuzcak M. High-Interaction linux honeypot architecture in recent perspective. *Computer networks*. Cham, 2016. P. 118–131. DOI: https://doi.org/10.1007/978-3-319-39207-3_11

35. Security and availability models for smart building automation systems / V. Kharchenko et al. *International Journal of Computing*. 2017. Vol. 16(4). P. 194–202. DOI: https://doi.org/10.47839/ijc.16.4.907

36. Moskalenko V., Kharchenko V., Moskalenko A., Kuzikov B. Resilience and resilient systems of artificial intelligence: Taxonomy, models and methods, *Algorithms*, 2023, vol. 16, 165. DOI: 10.3390/a16030165

37. Resilience and resilient systems of artificial intelligence: taxonomy, models and methods / V. Moskalenko et al. *Algorithms*. 2023. Vol. 16, no. 3. P. 165. DOI: https://doi.org/10.3390/a16030165

38. Model-driven deception for control system environments / W. Hofer et al. *2019 IEEE international symposium on technologies for homeland security (HST)*, Woburn, MA, USA, 5–6 November 2019. 2019. DOI: 10.1109/HST47167.2019.9032927

39. Khoroshko V., Khokhlachova Y., Vyshnevska N. Decomposition of computer network technology in their design. *Ukrainian scientific journal of information security*. 2023. Vol. 29, no. 3. P. 130–137. DOI: https://doi.org/10.18372/2225-5036.29.18072

40. Khoroshko V., Khokhlachova Y., Vyshnevska N. Choice of indicators for forecasting cyber protection of computer systems. *Ukrainian scientific journal of information security*. 2023. Vol. 29, no. 1. P. 41–47. DOI: https://orcid.org/0000-0001-9036-6556

41. A novel deception defense-based honeypot system for power grid network / M. Feng et al. *Lecture notes in computer science*. Cham, 2022. P. 297–307. DOI: https://doi.org/10.1007/978-3-030-97774-0_27

42. Ferguson-Walter, K., Fugate, S., Mauger, J., & Major, M. Game theory for adaptive defensive cyber deception. 2019. DOI: https://doi.org/10.1145/3314058.3314063

43. Walter E. C., Ferguson-Walter K. J., Ridley A. D. Incorporating deception into cyberbattlesim for autonomous defense. 2021. DOI: https://doi.org/10.48550/arXiv.2108.13980

44. A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems / J. Franco et al. *IEEE communications surveys & tutorials*. 2021. P. 1. DOI: 10.1109/COMST.2021.3106669

45. Shafronenko A. Y., Bodyanskiy Y. V., Holovin O. O. CLUSTERIZATION OF DATA ARRAYS BASED ON THE MODIFIED GRAY WOLF ALGORITHM . *Radio Electronics, Computer Science, Control*, 2023. Vol. 1, 73. DOI: https://doi.org/10.15588/1607-3274-2023-1-7

46. Bodyanskiy, Y. V., Pliss, I. P., & Shafronenko, A. Y. CLUSTERIZATION OF DATA ARRAYS BASED ON COMBINED OPTIMIZATION OF DISTRIBUTION DENSITY FUNCTIONS AND THE EVOLUTIONARY METHOD OF CAT SWARM. *Radio Electronics, Computer Science, Control*, 2021. Vol. 4. 61. DOI: https://doi.org/10.15588/1607-3274-2022-4-5

47. Bodyanskiy Y. V., Shafronenko A. Y. , Pliss I. P. Credibilistic fuzzy clustering based on evolutionary method of crazy cats, System Research and Information Technologies, 2021 (3), pp. 110–119. DOI: https://doi.org/10.20535/SRIT.2308-8893.2021.3.09

48. Прикладні методи комбінаторної оптимізації: навч. Посіб. / Л. Ф. Гуляницький, О. Ю. Мулеса. – К. : *Видавничо-поліграфічний центр «Київський університет»*, 2016. – 142 с.

49. Mirjalili S. The ant lion optimizer, Advances in Engineering Software, 2015, Vol. 83, pp. 80–98. DOI: https://doi.org/10.1016/j.advengsoft.2015.01.010

50. Mirjalili S. M., Lewis A. Grey wolf optimizer, Advances in Engineering Software, 2014, Vol. 69, pp. 46–61. DOI: https://doi.org/10.1016/j.advengsoft.2013.12.007

51. Mirjalili S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. Knowledge-Based Systems (2015), DOI: http://dx.doi.org/10.1016/j.knosys.2015.07.006

52. Povkhan I., Leheza A., Mulesa O., Melnyk O., Kintonova A. Method of hybrid logical classification trees based on group selection of discrete features. *Informatyka, Automatyka, Pomiary W Gospodarce I Ochronie Środowiska*. 2025. № 15(3). Pp. 15–21. DOI: https://doi.org/10.35784/iapgos.6957

53. Mulesa O., Bohdan Y. Development of a fuzzy production model for assessing the degree of information security in international cooperation. *Technology Audit and*

Production Reserves. 2024. № 6(2(80). Pp. 6–10. DOI: https://doi.org/10.15587/2706-5448.2024.318446

54. Syberfeldt A., Lidberg S. Real-world simulation-based manufacturing optimizations using cuckoo search, *Proceedings of the 2012 Winter Simulation Conference (WSC).* Berlin, Germany, December 2012, proceedings, pp. 1–12. DOI: https://doi.org/10.1109/WSC.2012.6465158

55. Sahoo, S.K., Saha, A.K., Ezugwu, A.E. *et al.* Moth Flame Optimization: Theory, Modifications, Hybridizations, and Applications. *Arch Computat Methods Eng* 30, Pp. 391–426 2023. DOI: https://doi.org/10.1007/s11831-022-09801-z

56. Rotshtein A., Shtovba S. Genetic Optimization of Multidimensional Technological Process Reliability. *In: Levitin, G. (eds) Computational Intelligence in Reliability Engineering. Studies in Computational Intelligence*, 2007. Vol 39. Springer, Berlin, Heidelberg. DOI: https://doi.org/10.1007/978-3-540-37368-1_9

57. Shtovba S.D. Ant Algorithms: Theory and Applications. *Program Comput Soft* 2005. Vol. 31. Pp. 167–178. DOI: https://doi.org/10.1007/s11086-005-0029-1

58. Rotshtein A.P., Zelinska O.V., Kaminskyi V.P. Optimization of Product Quality Indicators in the “Producer–Consumer” System Based on Fuzzy Cognitive Maps and Genetic Algorithm. *Cybern Syst Anal*. 2024. Vol. 60. Pp. 600–612. DOI: https://doi.org/10.1007/s10559-024-00699-y

59. Штовба С. Д. Генетичний алгоритм вибору правил нечіткої бази знань, збалансованої за критеріями точності та компактності [Електронний ресурс] / С. Д. Штовб,а В. В. Мазуренко, Д. А. Савчук // *Наукові праці Вінницького національного технічного університету.* – 2012. - № 3. URL : http://praci.vntu.edu.ua/index.php/praci/article/view/331

60. Про захист інформації в інформаційно-комунікаційних системах : Закон від 05.07.1994 № 80/94-ВР : станом на 16 груд. 2020 р. URL: https://zakon.rada.gov.ua/laws/show/80/94-вр#Text

61. Методичні рекомендації щодо забезпечення кіберзахисту автоматизованих систем управління технологічними процесами : Наказ від 29.05.2023 № 463. URL: https://zakon.rada.gov.ua/rada/show/v0463519-23#Text

62. ДСТУ ISO/IEC 27034-1:2017. Інформаційні технології. Методи захисту. Безпека прикладних програм. Частина 1. Огляд і загальні поняття (ISO/IEC 27034-1:2011; Cor 1:2014, IDT). Вид. офіц.

63. ДСТУ ISO/IEC 30111:2016. Інформаційні технології. Методи захисту. Процеси оброблення вразливостей (ISO/IEC 30111:2013, IDT). Вид. офіц.

64. *Bitdefender Total Security*. URL: https://www.bitdefender.com/ (date of access: 16.11.2023).

65. *AhnLab V3 Endpoint Security*. URL: https://www.ahnlab.com/ (date of access: 16.11.2023).

66. *AV-TEST*. URL: https://www.av-test.org/en/antivirus/home-windows/

67. *Sophos Home Premium*. URL – https://www.sophos.com/en-us (date of access: 16.11.2023).

68. *AVG Internet Security*. URL – https://www.avg.com/en-ww/homepage (date of access: 16.11.2023).

69. *TotalAV Pro Antivirus*. URL – https://www.totalav.com/ (date of access: 16.11.2023).

70. *Surfshark One Antivirus*. URL – https://surfshark.com/one (date of access: 16.11.2023).

71. *PC Protect*. URL – https://www.pcprotect.com/ (date of access: 16.11.2023).

72. *Scanguard*. URL – https://www.scanguard.com/ (date of access: 16.11.2023).

73. *Guardio*. URL – https://www.guard.io/ (date of access: 16.11.2023).

74. *Malwarebytes Premium*. URL – https://www.malwarebytes.com/ (date of access: 16.11.2023).

75. *Panda Dome*. URL – https://www.pandasecurity.com/ (date of access: 16.11.2023).

76. *Comodo Internet Security*. URL – https://www.comodo.com/ (date of access: 16.11.2023).

77. *ZoneAlarm Extreme Security*. URL – https://www.zonealarm.com/ (date of access: 16.11.2023).

78. *Emsisoft Anti-Malware*. URL – https://www.emsisoft.com/ (date of access: 16.11.2023).

79. *SentinelOne Singularity*. URL – https://www.sentinelone.com/ (date of access: 16.11.2023).

80. *Seqrite Endpoint Security*. URL – https://www.seqrite.com/ (date of access: 16.11.2023).

81. *Symantec Endpoint Security Complete*. URL – https://www.broadcom.com/ (date of access: 16.11.2023).

82. *WithSecure Elements Endpoint Protection*. URL – https://www.withsecure.com/ (date of access: 16.11.2023).

83. *Intego Antivirus*. URL – https://www.intego.com/ (date of access: 16.11.2023).

84. *Norton Antivirus*. URL – https://us.norton.com/ (date of access: 16.11.2023).

85. *McAfee*. URL – https://www.mcafee.com/ (date of access: 16.11.2023).

86. *Fortinet Network Firewall*. URL – https://www.fortinet.com/ (date of access: 16.11.2023).

87. *Avira Antivirus*. URL – https://www.avira.com/ (date of access: 16.11.2023).

88. *Avast Antivirus*. URL – https://www.avast.com/ (date of access: 16.11.2023).

89. *Zeek*. URL – https://zeek.org/ (date of access: 16.11.2023).

90. *OSSEC*. URL – https://www.ossec.net/ (date of access: 16.11.2023).

91. *Security Onion*. URL – https://securityonion.net/ (date of access: 16.11.2023).

92. *Wazuh*. URL – https://wazuh.com/ (date of access: 16.11.2023).

93. *MISP*. URL – https://www.misp-project.org/ (date of access: 16.11.2023).

94. *Osquery*. URL: https://osquery.io/ (date of access: 16.11.2023).

95. *Datasheet*. *Labyrinth Deception Platform*. URL: https://labyrinth.tech/assets/media/pdf/labyrinth-data-sheet.pdf (date of access: 16.11.2023).

96. *SentinelOne*. URL: https://www.sentinelone.com/surfaces/identity/ (date of access: 16.11.2023).

97. *Counter Craft Security*. URL: https://www.countercraftsec.com/ (дата звернення: 16.11.2023).

98. *Fidelis Security*. URL: https://fidelissecurity.com/fidelis-elevate/ (дата звернення: 16.11.2023).

99. *The Commvault Data Protection Platform*. URL: https://www.commvault.com/ (дата звернення: 16.11.2023).

100. *Proofpoint Identity Threat Defense*. URL: https://www.proofpoint.com/us/illusive-is-now-proofpoint (date of access: 16.11.2023).

101. Каштальян А. С. Принцип синтезу мультикомп'ютерних систем з комбінованих антивірусних приманок і пасток та контролеру прийняття рішень для виявлення та протидії зловмисному програмному забезпеченню та комп'ютерним атакам. *Measuring and computing devices in technological processes*, 2023. № 4, С. 265–272. DOI: https://doi.org/10.31891/2219-9365-2023-76-35

102. Каштальян А. С. Мультикомп'ютерна система з комбінованих антивірусних приманок і пасток для виявлення зловмисного програмного забезпечення та комп'ютерних атак на основі мультиагентних технологій. *Вісник Хмельницького національного університету. Серія: Технічні науки* 2025, № *347* (1). С. 535-542. DOI: https://doi.org/10.31891/

103. Савенко О.С., Дрозд А.І., Медзатий Д.М. Концептуальна архітектура обманних систем з приманками і пастками на основі популяційних алгоритмів. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №84(4). С. 127-151. DOI: https://doi.org/10.31891/2219-9365-2025-84-15

104. Vladov, S., Mulesa, O., Vysotska, V., Horvat, P., Paziura, N., Kolobylina, O., Mieshkov, O., Ilnytskyi, O., & Koropatov, O. Method for Detecting Low-Intensity DdoS Attacks Based on a Combined Neural Network and Its Application in Law Enforcement Activities. *Data*, 2025. № 10(11). Pp. 173. DOI: https://doi.org/10.3390/data10110173

105. Radivilova, T. *Et al.* Statistical and Signature Analysis Methods of Intrusion Detection. *In: Oliynykov, R., Kuznetsov, O., Lemeshko, O., Mulesa, O., Radivilova, T. (eds) Information Security Technologies in the Decentralized Distributed Networks. Lecture Notes on Data Engineering and Communications Technologies*. 2022. Vol. 115. Springer, Cham. DOI: https://doi.org/10.1007/978-3-030-95161-0_5

106. Tkachuk R., Polotai O., Balatska V., Brych T., Kukharska, N. (2025). МОДЕЛЮВАННЯ ЗАХИСТУ ОПЕРАЦІЙНИХ СИСТЕМ ВІД РЕАЛІЗАЦІЇ КІБЕРАТАК З ВИКОРИСТАННЯМ КРИТЕРІЮ ПІРСОНА. *Вісник Львівського державного університету безпеки життєдіяльності*, *31*, 117-125. https://doi.org/https://doi.org/10.32447/20784643.31.2025.12

107. Yashchuk V., Ivanusa A., Maslova N., Tkachuk R., Brych, T. (2025). КОНЦЕПТУАЛІЗАЦІЯ ІНТЕГРАТИВНОГО ВИКОРИСТАННЯ БАЗ ДАНИХ ВРАЗЛИВОСТЕЙ У КОНТЕКСТІ СИСТЕМНОГО МЕНЕДЖМЕНТУ ІНФОРМАЦІЙНОЇ БЕЗПЕКИ. *Вісник Львівського державного університету безпеки життєдіяльності*, *31*, 126-139. https://doi.org/https://doi.org/10.32447/20784643.31.2025.13

108. Sierhieiev Y., Savenko O., Paiuk V., Drozd A. Effectiveness and improvement of Static Application Security Testing (SAST) in the context of SQL Injection vulnerabilities // *Proceedings of 2024 IEEE 14th International Conference on Dependable Systems, Services and Technologies* (DeSSerT-2024, Athens, Greece, October 11-13, 2024) DOI: 10.1109/DESSERT65323.2024.11122171

109. Ivanusa, A., Tkachuk, R., Brych, T., Balatska, V., & Tkachenko, A. (2024). МЕТОДИ ТА МОДЕЛІ ПРОЄКТУВАННЯ СИСТЕМИ АВТОМАТИЗОВАНОГО ПОШУКУ ВРАЗЛИВОСТЕЙ У WEB-ДОДАТКАХ. *Вісник Львівського державного університету безпеки життєдіяльності*, *30*, 110-122. https://doi.org/https://doi.org/10.32447/20784643.30.2024.11

110. Semeniuk B., Kashtalian A., Martiniuk D., Drozd A., Abdel-Badeeh M. Salem. Detection of computer attacks based on sonification of network traffic. *Intelitsis'25: The 6th International Workshop on Intelligent Information Technologies & Systems of Information Security*, April 04, 2025, Khmelnytskyi, Ukraine https://ceur-ws.org/Vol-3963/paper21.pdf

111. Каштальян А. С. Метод виявлення зловмисного програмного забезпечення та комп'ютерних атак мультикомп'ютерними системами антивірусних комбінованих приманок та пасток. *Herald of Khmelnytskyi National University. Technical Sciences*, 2025, № *349*(2), С. 346-352. DOI: https://doi.org/10.31891/2307-5732-2025-349-50

112. Балацька В., Ткачук, Р., & Маслова, Н. (2025). ЕВОЛЮЦІЯ КСЗІ ТА ІНТЕГРАЦІЯ БЛОКЧЕЙН-ТЕХНОЛОГІЙ У КІБЕРЗАХИСТІ ДЕРЖАВНИХ ІНФОРМАЦІЙНИХ СИСТЕМ УКРАЇНИ. *Електронне фахове наукове видання «Кібербезпека: освіта, наука, техніка»*, *2*(30), 316–332. https://doi.org/10.28925/2663-4023.2025.30.975

113. Denysiuk D., Sochor T., Kapustian M., Kashtalian A., Drozd A. A method for detecting botnets in IT infrastructure using a neural network. (2024) *CEUR Workshop Proceedings*, 3736, pp. 282-292. *The 1th Proceedings of the 1st International Workshop on*

Intelligent & CyberPhysical Systems (IcyberPhyS 2024). Khmelnytskyi, Ukraine, June 28, 2024 : CEUR-Workshop Proceedings. Vol. 3736. (Khmelnytskyi, Ukraine, June 28, 2024). Khmelnytskyi, 2024. Pp. 282-292. URL: https://ceur-ws.org/Vol-3736/

114. Продеус М.С., Нічепорук А. О., Каштальян А. С. Система моніторингу, виявлення, реагування та захисту інформації на основі Honeypot-файлів. *Central Ukrainian Scientific Bulletin. Technical Sciences.* 2025. Issue 11(42), Part I. Pp. 56-67. DOI: https://mapiea.kntu.kr.ua/pdf/11(42)_I/8.pdf

115. Каштальян А. С. Методи організації функціонування мультикомп'ютерних систем антивірусних комбінованих приманок та пасток в корпоративних мережах. *Measuring and computing devices in technological processes*. 2025. № 1. С. 160–171. DOI: https://doi.org/10.31891/2219-9365-2025-81-19

116. Каштальян А., Савенко О., Грибинчук В. Моделі та типи приманок для зловмисних атак в корпоративних комп'ютерних мережах. *Вісник Хмельницького національного університету, технічні науки*. 2020. № 5. С. 45–50. DOI: https://www.doi.org/10.31891/2307-5732-2020-289-5-45-50

117. Dahiya M., Nitin N., Dahiya D. Intelligent Cyber Security Framework Based on SC-AJSO Feature Selection and HT-RLSTM Attack Detection. *Appl. Sci.* 2022, *12*, 6314. https://doi.org/10.3390/app12136314

118. Aslan O., Ozkan-Okay M., Gupta D. Intelligent behavior-based malware detection system on cloud computing environment. *IEEE Access.* 2021, *9*, 83252–83271.

119. Connor M., Michail O., Spirakis P. On the Distributed Construction of Stable Networks in Polylogarithmic Parallel Time. *Information.* 2021, *12*, 254. https://doi.org/10.3390/info12060254

120. Dai F., Hossain M.A., Wang Y. State of the Art in Parallel and Distributed Systems: Emerging Trends and Challenges. *Electronics.* 2025, *14*, 677. https://doi.org/10.3390/electronics14040677

121. Palko D., Babenko T., Bigdan A., Kiktev N., Hutsol T., Kuboń M. Hnatiienko H. Tabor S., Gorbovy O., Borusiewicz A. Cyber Security Risk Modeling in Distributed Information Systems. *Appl. Sci.* 2023, *13*, 2393. https://doi.org/10.3390/app13042393

122. Pinto S.J., Siano P., Parente M. Review of Cybersecurity Analysis in Smart Distribution Systems and Future Directions for Using Unsupervised Learning Methods for Cyber Detection. *Energies* 2023, *16*, 1651. https://doi.org/10.3390/en16041651

123. Ayele, E.D.; Gonzalez, J.F.; Teeuw, W.B. Enhancing Cybersecurity in Distributed Microgrids: A Review of Communication Protocols and Standards. *Sensors*. 2024, *24*, 854. https://doi.org/10.3390/s24030854

124. Zhou B., Sun B., Zang T., Cai Y., Wu J., Luo H. Security Risk Assessment Approach for Distribution Network Cyber Physical Systems Considering Cyber Attack Vulnerabilities. *Entropy* 2023, *25*, 47. https://doi.org/10.3390/e25010047

125. Wang X., Shi B., Fang Y. Distributed Systems for Emerging Computing: Platform and Application. *Future Internet.* 2023, *15*, 151. https://doi.org/10.3390/fi15040151

126. Acevedo-Iles M., Romero-Quete D., Cortes C.A. A Distributed Coordination Approach for Enhancing Protection System Adaptability in Active Distribution Networks. *Energies.* 2024, *17*, 4338. https://doi.org/10.3390/en17174338

127. Kurttisi A., Dogan K.M., Gruenwald B.C. A Novel Distributed Adaptive Controller for Multi-Agent Systems with Double-Integrator Dynamics: A Hedging-Based Approach. *Electronics.* 2024, *13*, 1142. https://doi.org/10.3390/electronics13061142

128. Li Q., Zeng, F. Enhancing Software Architecture Adaptability: A Comprehensive Evaluation Method. *Symmetry* 2024, *16*, 894. https://doi.org/10.3390/sym16070894

129. Ayedh M.A.T., Wahab A.W.A., Idris M.Y.I. Enhanced Adaptable and Distributed Access Control Decision Making Model Based on Machine Learning for Policy Conflict Resolution in BYOD Environment. *Appl. Sci.* 2023, *13*, 7102. https://doi.org/10.3390/app13127102

130. Brouillet M., Georgiev G.Y. Modeling and Predicting Self-Organization in Dynamic Systems out of Thermodynamic Equilibrium: Part 1: Attractor, Mechanism and Power Law Scaling. *Processes* 2024, *12*, 2937. https://doi.org/10.3390/pr12122937

131. Liu Q., Hagemeyer V., Keller H.B. A Review of Rule Learning-Based Intrusion Detection Systems and their Prospects in Smart Grids. *IEEE Access* 2021, *9*, 57542–57564. https://doi.org/10.1109/ACCESS.2021.3071263

132. Каштальян А., Савенко О. Покращення безпеки та модель антивірусних інтелектуальних приманок в корпоративних комп'ютерних мережах. *Вісник Хмельницького національного університету, технічні науки*. 2020. Т. 1, № 4. С. 33–38. DOI: https://doi.org/10.31891/2307-5732-2020-287-4-33-38

133. Каштальян А., Савенко О., Бельфер Р. Моделі приманок в корпоративних комп'ютерних мережах з врахуванням типів зловмисних атак. *Вимірювальна та обчислювальна техніка в технологічних процесах*. 2020. № 1. С. 104–110. DOI: https://doi.org/10.31891/2219-9365-2020-65-1-16

134. Savenko O., Rusyn B., Lysenko S., Ciszewski T., Savenko B., Drozd A., Nicheporuk A., Sachenko A. Synthesis of a Moth and Flame Algorithm for Incorporation into the Architecture of Deceptive Systems with Baits and Traps. *Applied Sciences*. 2026. 16(5). 2415. DOI: https://doi.org/10.3390/app16052415

135. Rehida, P., Savenko, O., Sachenko, A., Drozd, A., Vizhevski, P. A trust model that ensures the correctness of computing in grid computing system. (2024) *CEUR Workshop Proceedings*, 3675, pp. 388-401. *The 5th International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS-2024)*: CEUR-Workshop Proceedings. Vol. 3675. (Khmelnytskyi, March 2024). Khmelnytskyi, 2024. Pp. 388-401. URL: https://ceur-ws.org/Vol-3675/paper28.pdf

136. Kozelskyi O., Drozd A., Savenko B., Gaj P. A model for probabilistic monitoring and proactive restart of real-time operating systems under intensive state changes in cyber-physical systems. *Proceedings of the 2nd International Workshop on Intelligent & CyberPhysical Systems (IcyberPhyS 2025)* Khmelnytskyi, Ukraine on July 4, 2025. Pp. 198-210. URL: https://ceur-ws.org/Vol-4013/paper16.pdf

137. Каштальян А. С. Концептуальна модель архітектури мультикомп'ютерних систем із приманками та пастками для виявлення та протидії зловмисному програмному забезпеченню та комп'ютерним атакам. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2023, №3, С. 22-31. DOI: https://doi.org/10.32782/IT/2023-3-3

138. Каштальян А. С. Критерій оперативності щодо варіантів централізації в архітектурі мультикомп'ютерних систем з комбінованих антивірусних приманок і пасток для виявлення зловмисного програмного забезпечення та комп'ютерних атак. *Herald of Khmelnytskyi National University. Technical sciences, [S. l.]*, v. 345, n. 6(2), p. 172–178, 2024. DOI: 10.31891/. Disponível em: https://heraldts.khmnu.edu.ua/index.php/heraldts/article/view/995.

139. Каштальян А. С. Особливості правил формування варіантів централізації для мультикомп'ютерних систем в корпоративних мережах.

Measuring and computing devices in technological processes, [S. l.], 2024, n. 4, Pp. 386–393, DOI: https://doi.org/10.31891/2219-9365-2024-80-46

140. Каштальян А. С. Характерні властивості централізації в архітектурі мультикомп'ютерних систем антивірусних комбінованих приманок і пасток. *Information Technology: Computer Science, Software Engineering and Cyber Security*, 2024, № 4, С. 112-124. DOI: https://doi.org/10.32782/IT/2024-4-14

141. Lallie H.S., Thompson A., Titis E., Stephens P. Analysing Cyber Attacks and Cyber Security Vulnerabilities in the University Sector. *Computers*. 2025, 14, 49. DOI: https://doi.org/10.3390/computers14020049

142. Dawood M., Tu S., Xiao C., Alasmary H., Waqas M. Rehman S.U. Cyberattacks and Security of Cloud Computing: A Complete Guideline. *Symmetry* 2023, 15, 1981. DOI: https://doi.org/10.3390/sym15111981

143. Aslan Ö., Aktuğ S.S., Ozkan-Okay M., Yilmaz A.A., Akin E. A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions. *Electronics* 2023, 12, 1333. DOI: https://doi.org/10.3390/electronics12061333

144. Shah S.S.H., Ahmad A.R., Jamil N. Khan A.u.R. Memory Forensics-Based Malware Detection Using Computer Vision and Machine Learning. *Electronics* 2022, 11, 2579. DOI: https://doi.org/10.3390/electronics11162579

145. Malik E.F., Khaw K.W., Belaton B., Wong W.P. Chew X. Credit Card Fraud Detection Using a New Hybrid Machine Learning Architecture. *Mathematics* 2022, 10, 1480. DOI: https://doi.org/10.3390/math10091480

146. Gadal S., Mokhtar R., Abdelhaq M., Alsaqour R., Ali E.S. Saeed R. Machine Learning-Based Anomaly Detection Using K-Mean Array and Sequential Minimal Optimization. *Electronics*. 2022, 11, 2158. DOI: https://doi.org/10.3390/electronics11142158

147. Akhtar M.S., Feng T. Malware Analysis and Detection Using Machine Learning Algorithms. *Symmetry*. 2022, 14, 2304. DOI: https://doi.org/10.3390/sym14112304

148. Alashhab A.A., Zahid M.S.M., Azim M.A., Daha M.Y., Isyaku B. Ali S. A Survey of Low Rate DdoS Detection Techniques Based on Machine Learning in Software-Defined Networks. *Symmetry* 2022, 14, 1563. DOI: https://doi.org/10.3390/sym14081563

149. Saghezchi F.B., Mantas G., Violas M.A., de Oliveira Duarte A.M., Rodriguez J. Machine Learning for DdoS Attack Detection in Industry 4.0 CPPSs. *Electronics* 2022, 11, 602. DOI: https://doi.org/10.3390/electronics11040602

150. Najafi Mohsenabad H., Tut M.A. Optimizing Cybersecurity Attack Detection in Computer Networks: A Comparative Analysis of Bio-Inspired Optimization Algorithms Using the CSE-CIC-IDS 2018 *Dataset. Appl*. Sci. 2024, 14, 1044. DOI: https://doi.org/10.3390/app14031044

151. Ahmad I., Ul Haq Q.E., Imran M., Alassafi M.O., AlGhamdi R.A. An Efficient Network Intrusion Detection and Classification System. *Mathematics.* 2022, 10, 530. DOI: https://doi.org/10.3390/math10030530

152. Alnajim A.M., Habib S., Islam M., Thwin S.M., Alotaibi F. A Comprehensive Survey of Cybersecurity Threats, Attacks, and Effective Countermeasures in Industrial Internet of Things. *Technologies* 2023, 11, 161. DOI: https://doi.org/10.3390/technologies11060161

153. Aslam M.M., Tufail A., Kim K.-H., Apong R.A.A.H.M., Raza M.T. A Comprehensive Study on Cyber Attacks in Communication Networks in Water Purification and Distribution Plants: Challenges, Vulnerabilities, and Future Prospects. *Sensors.* 2023, 23, 7999. DOI: https://doi.org/10.3390/s23187999

154. Genuario F., Santoro G., Giliberti M., Bello S., Zazzera E. Impedovo D. Machine Learning-Based Methodologies for Cyber-Attacks and Network Traffic Monitoring: A Review and Insights. *Information*. 2024, 15, 741. DOI: https://doi.org/10.3390/info15110741

155. Ding S., Liu G., Yin L., Wang J., Li Z. Detection of Cyber-Attacks in a Discrete Event System Based on Deep Learning. *Mathematics*. 2024, 12, 2635. DOI: https://doi.org/10.3390/math12172635

156. Prabakar D., Sundarrajan M., Manikandan R., Jhanjhi N.Z., Masud M., Alqhatani A. Energy Analysis-Based Cyber Attack Detection by IoT with Artificial Intelligence in a Sustainable Smart City. *Sustainability* 2023, 15, 6031. DOI: https://doi.org/10.3390/su15076031

157. Xing W., Shen J. Security Control of Cyber–Physical Systems under Cyber Attacks: A Survey. *Sensors* 2024, 24, 3815. DOI: https://doi.org/10.3390/s24123815

158. Mozo A., Karamchandani A., de la Cal L., Gómez-Canaval S., Pastor A. Gifre L. A Machine-Learning-Based Cyberattack Detector for a Cloud-Based SDN Controller. *Appl. Sci*. 2023, 13, 4914. DOI: https://doi.org/10.3390/app13084914

159. Guo W., Shu H., Gu Y., Huang Y., Zhao H., Li Y. A Novel Virus Capable of Intelligent Program Infection through Software Framework Function Recognition. *Electronics* 2023, *12*, 460. DOI: https://doi.org/10.3390/electronics12020460

160. Šijan A., Viduka D., Ilić L., Predić B., Karabašević D. Modeling Cybersecurity Risk: The Integration of Decision Theory and Pivot Pairwise Relative Criteria Importance Assessment with Scale for Cybersecurity Threat Evaluation. *Electronics* 2024, *13*, 4209. DOI: https://doi.org/10.3390/electronics13214209

161. Aldea C.L., Bocu R., Solca R.N. Real-Time Monitoring and Management of Hardware and Software Resources in Heterogeneous Computer Networks through an Integrated System Architecture. *Symmetry*. 2023, *15*, 1134. DOI: https://doi.org/10.3390/sym15061134

162. Lee S,. Jeong T. Large-Scale Distributed System and Design Methodology for Real-Time Cluster Services and Environments. *Electronics* 2022, *11*, 4037. DOI: https://doi.org/10.3390/electronics11234037

163. Yang Y., Huang Y., Qin X., Lu S. Coded Distributed Computing Under Combination Networks. *Entropy* 2025, *27*, 311. DOI: https://doi.org/10.3390/e27030311

164. Huang L., Wang J., Cheng M., Deng Q., Zhong B. Coded Caching for Combination Networks with Multiaccess. *Information* 2022, *13*, 191. DOI: https://doi.org/10.3390/info13040191

165. Kodituwakku A., Gregor J. InDepth: A Distributed Data Collection System for Modern Computer Networks. *Electronics* 2025, *14*, 1974. DOI: https://doi.org/10.3390/electronics14101974

166. Golightly L., Modesti P., Chang V. Deploying Secure Distributed Systems: Comparative Analysis of GNS3 and SEED Internet Emulator. *J. Cybersecur. Priv.* 2023, *3*, 464-492. DOI: https://doi.org/10.3390/jcp3030024

167. Wang J.-J., Chen Y.-C., Chen M.-C. An Authentication Approach in a Distributed System Through Synergetic Computing. *Computers* 2025, *14*, 16. DOI: https://doi.org/10.3390/computers14010016

168. Дрозд А. Метод виявлення комп'ютерних атак типу відмови в обслуговуванні на основі статистичних показників мережного трафіку. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2025. № 4, С. 79–89. DOI: https://doi.org/10.32782/IT/2025-4-10

169. Савенко О.С., Дрозд А.І., Коробчинський М.В. Метод синтезу популяційних алгоритмів в архітектурі обманних систем з приманками і пастками. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №82(2). С. 459–474. DOI: https://doi.org/10.31891/2219-9365-2025-82-64

170. RAMSKYI I., DROZD A., LYHUN O., PONOCHOVNA O. SYSTEM FOR CYBERSECURITY EVALUATION OF CORPORATE NETWORKS. *Computer Systems and Information Technologies*. 2025. № 2. С. 123–131. DOI: https://doi.org/10.31891/csit-2025-2-14

171. Дрозд А.І. Метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах. *Вісник Хмельницького національного університету. Технічні науки*. 2025. № 359 (6.2). С. 445–457. DOI: https://doi.org/10.31891/2307-5732-2025-359-135

172. СЛОВНИКUA. URL: https://slovnyk.ua/index.php?swrd=%D0%BE%D0%BF%D1%82%D0%B8%D0%BC%D1%96%D0%B7%D0%B0%D1%86%D1%96%D1%8F Дата звернення: 09.12.2025.

173. Sahoo SK, Saha AK (2022) A hybrid moth flame optimization algorithm for global optimization. *J Bionic Eng*. DOI: https://doi.org/10.1007/s42235-022-00207-y

174. Wu Y., Chen R., Li C., Zhang L., Dai W. An adaptive sine-cosine moth-flame optimization algorithm for parameter identification of hybrid active power filters in power systems. *IEEE Access* 8:156378–156393. 2020. DOI: https://doi.org/https://doi.org/10.1109/ACCESS.2020.3005717

175. Wu Y., Chen R., Li C., Zhang L., Cui Z. Hybrid symbiotic differential evolution moth-flame optimization algorithm for estimating parameters of photovoltaic models. *IEEE Access* 8:156328–156346. 2020. DOI: https://doi.org/https://doi.org/10.1109/ACCESS.2020.3005711

176. Zhang Z., Qin H., Yao L., Liu Y., Jiang Z., Feng Z., Ouyang S. Improved multi-objective moth-flame optimization algorithm based on R-domination for cascade reservoirs operation. *J Hydrol* 581:124431. 2020. DOI: https://doi.org/10.1016/j.jhydrol.2019.124431

177. Wang M., Chen H., Yang B., Zhao X., Hu L., Cai Z., Huang H., Tong C. Toward an optimal kernel extreme learning machine using a chaotic moth-flame optimization strategy with applications in medical diagnoses. *Neurocomputing* 267:69–84. 2017. DOI: https://doi.org/10.1016/j.neucom.2017.04.060

178. Ma L, Wang C, Xie N, Shi M, Ye Y, Wang L (2021) Moth-flame optimization algorithm based on diversity and mutation strategy. Appl Intell. https://doi.org/10.1007/s10489-020-02081-9

179. Zhao X, Fang Y, Liu L, Li J, Xu M (2020) An improved moth-flame optimization algorithm with orthogonal opposition-based learning and modified position updating mechanism of moths for global optimization problems. Appl Intell 50(12):4434–4458. https://doi.org/10.1007/s10489-020-01793-2

180. Dabba, A., Tari, A. & Meftali, S. Hybridization of Moth flame optimization algorithm and quantum computing for gene selection in microarray data. *J Ambient Intell Human Comput* **12**, 2731–2750 (2021). https://doi.org/10.1007/s12652-020-02434-9

181. Свідоцтво про реєстрацію авторського права на твір № 142624 Україна. Комп'ютерна програма «Програмне забезпечення функціонування обманних систем в корпоративних мережах з прийняттям рішень на основі популяційних алгоритмів» / Дрозд А. І., Савенко О. С., Нічепорук А. О., Регіда П. Г. 13.02.2026.

ДОДАТКИ

ДОДАТОК А

Список публікацій здобувача за темою дисертації

Наукові праці, в яких опубліковані основні наукові результати дисертації

1. Савенко О.С., Дрозд А.І., Медзатий Д.М. Концептуальна архітектура обманних систем з приманками і пастками на основі популяційних алгоритмів. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №84(4). С. 127-151. DOI: https://doi.org/10.31891/2219-9365-2025-84-15

2. Дрозд А. Метод виявлення комп'ютерних атак типу відмови в обслуговуванні на основі статистичних показників мережного трафіку. *Information Technology: Computer Science, Software Engineering and Cyber Security*. 2025. № 4, С. 79–89. DOI: https://doi.org/10.32782/IT/2025-4-10

3. Савенко О.С., Дрозд А.І., Коробчинський М.В. Метод синтезу популяційних алгоритмів в архітектурі обманних систем з приманками і пастками. *Вимірювальна та обчислювальна техніка в технологічних процесах. Measuring and computing devices in technological processes*. 2025. №82(2). С. 459–474. DOI: https://doi.org/10.31891/2219-9365-2025-82-64

4. RAMSKYI I., DROZD A., LYHUN O., PONOCHOVNA O. SYSTEM FOR CYBERSECURITY EVALUATION OF CORPORATE NETWORKS. *Computer Systems and Information Technologies*. 2025. № 2. С. 123–131. DOI: https://doi.org/10.31891/csit-2025-2-14

5. Дрозд А.І. Метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах. *Вісник Хмельницького національного університету. Технічні науки*. 2025. № 359 (6.2). С. 445–457. DOI: https://doi.org/10.31891/2307-5732-2025-359-135

6. Savenko O., Rusyn B., Lysenko S., Ciszewski T., Savenko B., Drozd A., Nicheporuk A., Sachenko A. Synthesis of a Moth and Flame Algorithm for Incorporation into the Architecture of Deceptive Systems with Baits and Traps. *Applied Sciences*. 2026. 16(5). 2415. DOI: https://doi.org/10.3390/app16052415

Праці, які засвідчують апробацію матеріалів дисертації

7. Rehida, P., Savenko, O., Sachenko, A., Drozd, A., Vizhevski, P. A trust model that ensures the correctness of computing in grid computing system. (2024) *CEUR Workshop Proceedings*, 3675, pp. 388-401. *The 5th International Workshop on Intelligent Information Technologies & Systems of Information Security (IntelITSIS-2024)*: CEUR-Workshop Proceedings. Vol. 3675. (Khmelnytskyi, March 2024). Khmelnytskyi, 2024. Pp. 388-401. URL: https://ceur-ws.org/Vol-3675/paper28.pdf *(Scopus)*

8. Denysiuk D., Sochor T., Kapustian M., Kashtalian A., Drozd A. A method for detecting botnets in IT infrastructure using a neural network. (2024) *CEUR Workshop Proceedings*, 3736, pp. 282-292. *The 1th Proceedings of the 1st International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS 2024)*. Khmelnytskyi, Ukraine, June 28, 2024 : CEUR-Workshop Proceedings. Vol. 3736. (Khmelnytskyi, Ukraine, June 28, 2024). Khmelnytskyi, 2024. Pp. 282-292. URL: https://ceur-ws.org/Vol-3736/paper21.pdf *(Scopus)*

9. Sierhieiev Y., Savenko O., Paiuk V., Drozd A. Effectiveness and improvement of Static Application Security Testing (SAST) in the context of SQL Injection vulnerabilities // *Proceedings of 2024 IEEE 14th International Conference on Dependable Systems, Services and Technologies* (DeSSerT-2024, Athens, Greece, October 11-13, 2024) DOI: 10.1109/DESSERT65323.2024.11122171 *(Scopus)*

10. Semeniuk B., Kashtalian A., Martiniuk D., Drozd A., Abdel-Badeeh M. Salem. Detection of computer attacks based on sonification of network traffic. *Intelitsis'25: The 6th International Workshop on Intelligent Information Technologies & Systems of Information Security*, April 04, 2025, Khmelnytskyi, Ukraine. URL: https://ceur-ws.org/Vol-3963/paper21.pdf *(Scopus)*

11. Kozelskyi O., Drozd A., Savenko B., Gaj P. A model for probabilistic monitoring and proactive restart of real-time operating systems under intensive state changes in cyber-physical systems. *Proceedings of the 2nd International Workshop on Intelligent & CyberPhysical Systems (ICyberPhyS 2025)* Khmelnytskyi, Ukraine on July 4, 2025. Pp. 198-210. URL: https://ceur-ws.org/Vol-4013/paper16.pdf *(Scopus)*

Публікації, які додатково відображають наукові результати дисертації

12. Свідоцтво про реєстрацію авторського права на твір № 142624 Україна. Комп’ютерна програма «Програмне забезпечення функціонування обманних систем в корпоративних мережах з прийняттям рішень на основі популяційних алгоритмів» / Дрозд А. І., Савенко О. С., Нічепорук А. О., Регіда П. Г. 13.02.2026.

ДОДАТОК Б

Акти впровадження

ГОЛОВНЕ УПРАВЛІННЯ РОЗВІДКИ
МІНІСТЕРСТВА ОБОРОНИ УКРАЇНИ
ВОЄННА АКАДЕМІЯ
ІМЕНІ ЄВГЕНІЯ БЕРЕЗНЯКА
Код 14303342
«18» 12 2025р.
№ 222/ВА/2167
04050, м. Київ

ЗАТВЕРДЖУЮ
Тимчасово виконуючий обов'язки
начальника Воєнної академії
імені Євгенія Березняка
полковник
Вячеслав КОЛОТОВ
18.12.2025

АКТ

впровадження результатів дисертаційної роботи
"Методи та системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів"
Дрозда Андрія Ігоровича

Комісія в складі: голова комісії – начальник Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка, кандидат технічних наук, доцент, полковник Чумак Олександр Ілліч; члени комісії: заступник начальника другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка, кандидат технічних наук, доцент, полковник Руденко Михайло Миколайович; доцент другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка, кандидат технічних наук, доцент, полковник Гончарук Андрій Вікторович; старший викладач другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка, кандидат технічних наук, старший науковий співробітник, полковник Волков Олександр Віталійович, склала цей акт про те, що результати дисертаційної роботи здобувача наукового ступеня доктора філософії Дрозда А.І. впроваджені в освітній процес у блоці військово-спеціальних дисциплін та використані при удосконаленні навчально-лабораторного комплексу другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка.

До переліку впроваджених та використаних результатів дисертаційної роботи Дрозда А.І. належать:

1) архітектура обманних систем з приманками і пастками, особливістю якої є синтез в ній популяційних алгоритмів, зокрема алгоритму молі і полум'я,

для оптимізації формування послідовності наступних кроків при здійсненні комп'ютерних атак, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміни послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових комп'ютерних атак;

2) метод синтезу алгоритму дискретної оптимізації молі і полум'я в архітектурі обманних систем з приманками і пастками, який характеризується формуванням дискретного простору пошуку з координатним поданням об'єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі і полум'я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале та адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

3) метод виявлення атак у мережах на основі статистичних показників, який базується на обчисленні статистичних ознак мережного IP-трафіку при розбитті потоку пакетів на часові вікна і встановлює динамічні зміни трафіку на рівні всього аналізованого періоду, що дозволяє підвищити достовірність виявлення атак.

Впровадження та використання зазначених результатів дозволили підвищити якість викладання військово-спеціальних дисциплін та удосконалити навчально-лабораторний комплекс другої кафедри Другого навчально-наукового інституту Воєнної академії імені Євгенія Березняка.

Голова комісії: полковник Олександр ЧУМАК

Члени комісії: полковник Михайло РУДЕНКО

полковник Андрій ГОНЧАРУК

полковник Олександр ВОЛКОВ

«Затверджую»
Проректор з науково-педагогічної роботи
Віктор ЛОПАТОВСЬКИЙ
«25» січня 2026 р.

АКТ

про впровадження в навчальний процес Хмельницького національного університету результатів дисертаційної роботи аспіранта кафедри комп'ютерної інженерії та інформаційних систем Дрозда Андрія Ігоровича «Методи та системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів»

Ми, комісія в складі: декана факультету інформаційних технологій, професора кафедри комп'ютерної інженерії та інформаційних систем, д.т.н., професора Говорущенко Т. О. (голова комісії), завідувача кафедри комп'ютерної інженерії та інформаційних систем, д.ф., доцента Павлової О. О., доцента кафедри комп'ютерної інженерії та інформаційних систем, к.т.н., доцента Капустян М. В. склала акт про те, що результати дисертаційної роботи Дрозда А.І. впроваджені та використовуються в освітньому процесі Хмельницького національного університету при викладанні дисциплін на кафедрі комп'ютерної інженерії та інформаційних систем для здобувачів спеціальності F7 Комп'ютерна інженерія, зокрема в курсах «Безпека та захист комп'ютерних систем», «Моделювання та методи оптимізації в наукових та експериментальних дослідженнях», «Методології забезпечення якості, надійності, гарантоздатності та безпеки комп'ютерних систем та мереж».

При викладанні цих дисциплін викладачами кафедр використовувалися наступні матеріали досліджень, отримані Дроздом А.І. особисто:

1) архітектура обманних систем з приманками і пастками, в якій на відміну від відомих варіантів архітектури, здійснено синтез популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових КА;

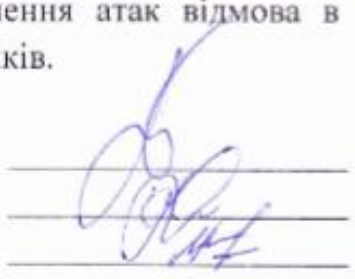
2) метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, який, на відміну від відомих, характеризується формуванням дискретного простору пошуку з

координатним поданням об'єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі й полум'я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале й адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

3) метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах, в якому на відміну від відомих, в архітектурі обманних систем синтезовано популяційні алгоритми, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак, що дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж та ускладнює дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп'ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах;

4) метод виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників, який на відміну від відомих, базується на обчисленні статистичних ознак мережного IP-трафіку при розбитті потоку пакетів на часові вікна, і встановлює динамічні зміни трафіку на рівні всього аналізованого періоду, що дозволяє підвищити достовірність виявлення атак відмова в обслуговуванні.

Отримані матеріали досліджень дозволили розробити лабораторні практикуми з використанням архітектури обманних систем з приманками і пастками та методу створення таких систем на основі синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, методу виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників.



Говорущенко Т. О.
Павлова О. О.
Капустян М. В.

«Затверджую»
Технічний директор ПП «Нолт Технолоджис»
Мельниченко О.В.
16 лютого 2026 р.

АКТ
про впровадження результатів дисертаційної роботи
аспіранта кафедри комп'ютерної інженерії та інформаційних систем
Хмельницького національного університету Дрозда Андрія Ігоровича
«Методи та системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів»

Результати дисертаційної роботи аспіранта кафедри комп'ютерної інженерії та інформаційних систем Хмельницького національного університету Дрозда А.І. впроваджені на ПП «Нолт Технолоджис».

При розробленні обманної системи з приманками і пастками в корпоративній мережі, включаючи підсистему синтезованих популяційних алгоритмів в архітектуру системи, були використані на ПП «Нолт Технолоджис» такі результати, які одержані Дроздом А.І. особисто:

1) архітектура обманних систем з приманками і пастками, в якій на відміну від відомих варіантів архітектури, здійснено синтез популяційних алгоритмів, зокрема алгоритму молі і полум'я, для оптимізації формування послідовності наступних кроків при здійсненні КА та дій ЗПЗ, уникнення повного перебору варіантів, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж, а також врахування потенційної спроможності зловмисників до здійснення двоцільових КА;

2) метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, який, на відміну від відомих, характеризується формуванням дискретного простору пошуку з координатним поданням об'єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі й полум'я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале й адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

3) метод організації функціонування обманних систем з приманками і

пастками в корпоративних мережах, в якому на відміну від відомих, в архітектурі обманних систем синтезовано популяційні алгоритми, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак, що дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж та ускладнює дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп'ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах.

Отриманні результати дозволили покращити протидію зловмисним діям у корпоративній мережі шляхом удосконалення методів функціонування розробленої обманної системи з приманками і пастками та реалізованого в ній методу виявлення атак та підвищити точність виявлення приблизно на 5-6,5 %.

Цей акт не є підставою для фінансових розрахунків.

Технічний директор Олександр МЕЛЬНИЧЕНКО

«Затверджую»
Директор ТОВ «ІТТ»
Сімогук В.С.
«17» лютого 2026 р.

АКТ

про впровадження результатів дисертаційної роботи
аспіранта кафедри комп'ютерної інженерії та інформаційних систем
Хмельницького національного університету Дрозда Андрія Ігоровича
«Методи та системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів»

Результати дисертаційної роботи Дрозда А.І. аспіранта кафедри комп'ютерної інженерії та інформаційних систем Хмельницького національного університету впроваджені на ТОВ «ІТТ».

При розробленні системи виявлення комп'ютерних атак в корпоративних мережах на основі популяційних алгоритмів були отримані на ТОВ «ІТТ» такі результати, які одержані Дроздом А.І. особисто:

1) розроблено новий метод синтезу алгоритму дискретної оптимізації молі й полум'я в архітектурі обманних систем з приманками і пастками, який, на відміну від відомих, характеризується формуванням дискретного простору пошуку з координатним поданням об'єктів, синтезом спірального сліду на основі секторного оцінювання потенційних кроків і кутових характеристик, урахуванням часу як параметра зміни кроків та динамічним переміщенням молі й полум'я для уникнення передчасної збіжності до локальних оптимумів, що дало змогу розробляти обманні системи, які забезпечують довготривале й адаптивне функціонування у процесі протидії зловмисникам у корпоративних мережах за рахунок зміни кроків для опрацювання подій;

2) розроблено новий метод організації функціонування обманних систем з приманками і пастками в корпоративних мережах, в якому на відміну від відомих, в архітектурі обманних систем синтезовано популяційні алгоритми, зокрема алгоритм молі і полум'я, для здійснення ними вибору наступних кроків для уникнення реалізації зловмисниками двоцільових атак, що дає змогу уникати повного перебору варіантів з можливих кроків, швидкої збіжності обраних кроків при триваючих впливах та зміну послідовності кроків з врахуванням поточних змін в оточуючому середовищі корпоративних мереж та ускладнює дії за рахунок прийняття рішень на основі популяційних алгоритмів з можливістю самостійно блокувати або активувати сервери чи комп'ютерні станції, приманки чи пастки під час встановлення потенційно зловмисних впливів в корпоративних мережах;

3) розроблено новий метод виявлення атак відмова в обслуговуванні у мережах на основі статистичних показників, який на відміну від відомих, базується на обчисленні статистичних ознак мережного IP-трафіку при розбитті потоку пакетів на часові вікна, і встановлює динамічні зміни трафіку на рівні всього аналізованого періоду, що дозволяє підвищити достовірність виявлення атак відмова в обслуговуванні.

Отриманні результати дозволили покращити протидію зловмисним діям у корпоративній мережі шляхом удосконалення архітектури та методів функціонування розробленої обманної системи з приманками і пастками та сумісно з реалізованим в ній методом виявлення атак підвищити точність виявлення приблизно на 4-6%.

Цей акт не є підставою для фінансових розрахунків.

Заступник директора Іванов О.В.

Системний адміністратор Веремеєнко В.А.

ДОДАТОК В

Результати експериментів

Таблиця В.1 - Детальні відомості про вузли корпоративної мережі та встановлені компоненти обманної системи з приманками і пастками

Номер компоненти системи	Тип вузла: комп'ютерна станція; сервер	Номер сегменту: 1..25	IP-адреса вузла	Наявність та тип активної приманки та/чи пастки	Належність вузла демілітаризованій зоні	Тип публічного сервісу	Операційна система	Наявність центру системи	Номер будівлі
1	2	3	4	5	6	7	8	9	10
1	КС	1	10.0.1.10	-	-	-	Windows 10	-	1
2	КС	1	10.0.1.11	-	-	-	Windows 10	-	1
3	сервер	1	10.0.1.12	-	-	-	Windows 10	-	1
4	КС	1	10.0.1.13	-	-	-	Ubuntu 22.04	-	1
5	КС	1	10.0.1.14	-	-	-	Ubuntu 22.04	-	1
6	КС	1	10.0.1.15	-	-	-	Ubuntu 22.04	-	1
7	сервер	1	10.0.1.16	+	-	-	Windows 10	+	1
8	КС	1	10.0.1.17	-	-	-	Windows 10	-	1
9	КС	2	10.0.2.10	-	-	-	Windows 10	-	1
10	КС	2	10.0.2.11	-	-	-	Windows 10	-	1
11	КС	2	10.0.2.12	-	-	-	Windows 10	-	1
12	КС	2	10.0.2.13	-	-	-	Windows 10	-	1
13	КС	2	10.0.2.14	-	-	-	Windows 10	-	1
14	КС	2	10.0.2.15	-	-	-	Windows 10	-	1
15	КС	2	10.0.2.16	-	-	-	Windows 10	-	1

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
16	КС	2	10.0.2.17	-	-	-	Windows 10	-	1
17	КС	2	10.0.2.18	-	-	-	Windows 10	-	1
18	КС	2	10.0.2.19	-	-	-	Windows 10	-	1
19	КС	2	10.0.2.20	-	-	-	Windows 10	-	1
20	КС	2	10.0.2.21	+	-	-	Windows 10	-	1
21	КС	2	10.0.2.22	+	-	-	Windows 10	-	1
22	КС	2	10.0.2.23	-	-	-	Windows 10	-	1
23	КС	3	10.0.3.10	-	-	-	Windows 10	-	1
24	сервер	3	10.0.3.11	-	-	-	Windows 10	-	1
25	КС	3	10.0.3.12	-	-	-	Windows 10	-	1
26	КС	3	10.0.3.13	-	-	-	Windows 10	-	1
27	КС	3	10.0.3.14	-	-	-	Windows 10	-	1
28	КС	3	10.0.3.15	-	-	-	Windows 10	-	1
29	сервер	3	10.0.3.16	-	-	-	Windows 10	-	1
30	КС	3	10.0.3.17	-	-	-	Windows 10	-	1
31	КС	3	10.0.3.18	-	-	-	Windows 10	-	1
32	КС	3	10.0.3.19	-	-	-	Windows 10	-	1
33	КС	3	10.0.3.20	-	-	-	Windows 10	-	1
34	КС	3	10.0.3.21	-	-	-	Windows 10	-	1
35	КС	3	10.0.3.22	-	-	-	Windows 10	-	1
36	КС	3	10.0.3.23	-	-	-	Windows 10	-	1
37	КС	4	10.0.4.10	-	-	-	Windows 10	-	1
38	КС	4	10.0.4.11	-	-	-	Windows 10	-	1
39	КС	4	10.0.4.12	-	-	-	Windows 10	-	1

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
40	КС	4	10.0.4.13	-	-	-	Windows 10	-	1
41	сервер	4	10.0.4.14	-	-	-	Windows 10	-	1
42	КС	4	10.0.4.15	-	-	-	Windows 10	-	1
43	КС	4	10.0.4.16	-	-	-	Windows 10	-	1
44	КС	4	10.0.4.17	+	-	-	Windows 10	-	1
45	КС	4	10.0.4.18	-	-	-	Windows 10	-	1
46	КС	4	10.0.4.19	-	-	-	Windows 10	-	1
47	КС	4	10.0.4.20	-	-	-	Windows 10	-	1
48	КС	4	10.0.4.21	-	-	-	Windows 10	-	1
49	КС	4	10.0.4.22	-	-	-	Windows 10	-	1
50	КС	4	10.0.4.23	-	-	-	Windows 10	-	1
51	КС	5	10.0.5.10	-	-	-	Windows 10	-	1
52	КС	5	10.0.5.11	-	-	-	Windows 10	-	1
53	КС	5	10.0.5.12	-	-	-	Windows 10	-	1
54	КС	5	10.0.5.13	-	-	-	Windows 10	-	1
55	КС	5	10.0.5.14	+	-	-	Windows 10	-	1
56	КС	5	10.0.5.15	-	-	-	Windows 10	-	1
57	КС	5	10.0.5.16	-	-	-	Windows 10	-	1
58	КС	5	10.0.5.17	-	-	-	Windows 10	-	1
59	КС	5	10.0.5.18	-	-	-	Windows 10	-	1
60	КС	5	10.0.5.19	-	-	-	Windows 10	-	1
61	КС	5	10.0.5.20	-	-	-	Windows 10	-	1
62	КС	5	10.0.5.21	-	-	-	Windows 10	-	1
63	КС	5	10.0.5.22	-	-	-	Windows 10	-	1

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
64	КС	5	10.0.5.23	-	-	-	Windows 10	-	1
65	КС	6	10.0.6.10	-	-	-	Windows 10	-	1
66	КС	6	10.0.6.11	-	-	-	Windows 10	-	1
67	КС	6	10.0.6.12	-	-	-	Windows 10	-	1
68	КС	6	10.0.6.13	-	-	-	Windows 10	-	1
69	КС	6	10.0.6.14	-	-	-	Windows 10	-	1
70	КС	6	10.0.6.15	-	-	-	Windows 10	-	1
71	КС	6	10.0.6.16	-	-	-	Windows 10	-	1
72	КС	6	10.0.6.17	-	-	-	Windows 10	-	1
73	КС	6	10.0.6.18	-	-	-	Windows 10	-	1
74	КС	7	10.0.7.10	-	-	-	Windows 10	-	1
75	КС	7	10.0.7.11	-	-	-	Windows 10	-	1
76	КС	7	10.0.7.12	-	-	-	Windows 10	-	1
77	КС	8	10.0.8.10	-	-	-	Windows 10	-	1
78	КС	8	10.0.8.11	-	-	-	Windows 10	-	1
79	КС	8	10.0.8.12	-	-	-	Windows 10	-	1
80	КС	8	10.0.8.13	-	-	-	Windows 10	-	1
81	КС	8	10.0.8.14	-	-	-	Windows 10	-	1
82	КС	9	10.0.9.10	-	-	-	Windows 10	-	1
83	КС	9	10.0.9.11	-	-	-	Windows 10	-	1
84	КС	9	10.0.9.12	-	-	-	Windows 10	-	1
85	КС	9	10.0.9.13	-	-	-	Windows 10	-	1
86	КС	9	10.0.9.14	-	-	-	Windows 10	-	1
87	КС	9	10.0.9.15	-	-	-	Windows 10	-	1

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
88	КС	9	10.0.9.16	-	-	-	Windows 10	-	1
89	КС	9	10.0.9.17	-	-	-	Windows 10	-	1
90	КС	9	10.0.9.18	-	-	-	Windows 10	-	1
91	КС	9	10.0.9.19	-	-	-	Windows 10	-	1
92	КС	9	10.0.9.20	-	-	-	Windows 10	-	1
93	сервер	10	10.0.10.10	-	+	Web-сервер	Windows Server 2022	-	1
94	сервер	10	10.0.10.11	-	+	Mail-сервер	Windows Server 2022	-	1
95	сервер	10	10.0.10.12	-	+	FTP-сервер	Windows Server 2022	-	1
96	КС	10	10.0.10.13	-	+	-	Ubuntu 22.04	-	1
97	КС	10	10.0.10.14	-	+	-	Windows 10	-	1
98	КС	10	10.0.10.15	-	+	-	Ubuntu 22.04	-	1
99	КС	10	10.0.10.16	-	+	-	Windows 10	-	1
100	КС	10	10.0.10.17	-	+	-	Ubuntu 22.04	-	1
101	КС	10	10.0.10.18	-	+	-	Windows 10	-	1
102	КС	10	10.0.10.19	-	+	-	Windows 10	-	1
103	КС	10	10.0.10.20	-	+	-	Ubuntu 22.04	-	1
104	КС	10	10.0.10.21	-	+	-	Windows 10	-	1
105	КС	10	10.0.10.22	-	+	-	Windows 10	-	1
106	КС	10	10.0.10.23	-	+	-	Windows 10	-	1
107	КС	11	10.0.11.10	-	-	-	Windows 10	-	2
108	сервер	11	10.0.11.11	-	-	-	Windows 10	-	2
109	сервер	11	10.0.11.12	-	-	-	Windows 10	-	2
110	КС	11	10.0.11.13	-	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
111	КС	11	10.0.11.14	-	-	-	Windows 10	-	2
112	КС	11	10.0.11.15	-	-	-	Windows 10	-	2
113	КС	11	10.0.11.16	-	-	-	Windows 10	-	2
114	КС	11	10.0.11.17	-	-	-	Windows 10	-	2
115	КС	12	10.0.12.10	-	-	-	Windows 10	-	2
116	КС	12	10.0.12.11	-	-	-	Windows 10	-	2
117	КС	12	10.0.12.12	-	-	-	Windows 10	-	2
118	КС	12	10.0.12.13	-	-	-	Windows 10	-	2
119	КС	12	10.0.12.14	-	-	-	Windows 10	-	2
120	КС	12	10.0.12.15	-	-	-	Windows 10	-	2
121	КС	12	10.0.12.16	-	-	-	Windows 10	-	2
122	КС	12	10.0.12.17	-	-	-	Windows 10	-	2
123	КС	12	10.0.12.18	-	-	-	Windows 10	-	2
124	КС	12	10.0.12.19	-	-	-	Windows 10	-	2
125	КС	12	10.0.12.20	-	-	-	Windows 10	-	2
126	КС	12	10.0.12.21	-	-	-	Windows 10	-	2
127	КС	13	10.0.13.10	-	-	-	Windows 10	-	2
128	КС	13	10.0.13.11	-	-	-	Windows 10	-	2
129	КС	13	10.0.13.12	-	-	-	Windows 10	-	2
130	КС	13	10.0.13.13	-	-	-	Windows 10	-	2
131	КС	13	10.0.13.14	-	-	-	Windows 10	-	2
132	КС	13	10.0.13.15	-	-	-	Windows 10	-	2
133	КС	13	10.0.13.16	-	-	-	Windows 10	-	2
134	КС	13	10.0.13.17	+	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
135	КС	13	10.0.13.18	-	-	-	Windows 10	-	2
136	КС	13	10.0.13.19	-	-	-	Windows 10	-	2
137	КС	13	10.0.13.20	-	-	-	Windows 10	-	2
138	КС	13	10.0.13.21	-	-	-	Windows 10	-	2
139	КС	13	10.0.13.22	-	-	-	Windows 10	-	2
140	КС	13	10.0.13.23	-	-	-	Windows 10	-	2
141	КС	14	10.0.14.10	-	-	-	Windows 10	-	2
142	КС	14	10.0.14.11	-	-	-	Windows 10	-	2
143	КС	14	10.0.14.12	-	-	-	Windows 10	-	2
144	КС	14	10.0.14.13	-	-	-	Windows 10	-	2
145	КС	14	10.0.14.14	-	-	-	Windows 10	-	2
146	КС	14	10.0.14.15	-	-	-	Windows 10	-	2
147	КС	14	10.0.14.16	-	-	-	Windows 10	-	2
148	КС	14	10.0.14.17	-	-	-	Windows 10	-	2
149	КС	14	10.0.14.18	-	-	-	Windows 10	-	2
150	КС	14	10.0.14.19	-	-	-	Windows 10	-	2
151	КС	14	10.0.14.20	-	-	-	Windows 10	-	2
152	КС	14	10.0.14.21	-	-	-	Windows 10	-	2
153	КС	14	10.0.14.22	-	-	-	Windows 10	-	2
154	КС	14	10.0.14.23	+	-	-	Windows 10	-	2
155	КС	15	10.0.15.10	-	-	-	Windows 10	-	2
156	КС	15	10.0.15.11	-	-	-	Windows 10	-	2
157	КС	15	10.0.15.12	-	-	-	Windows 10	-	2
158	КС	15	10.0.15.13	-	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
159	КС	15	10.0.15.14	-	-	-	Windows 10	-	2
160	КС	15	10.0.15.15	-	-	-	Windows 10	-	2
161	КС	15	10.0.15.16	-	-	-	Windows 10	-	2
162	КС	15	10.0.15.17	-	-	-	Windows 10	-	2
163	КС	15	10.0.15.18	-	-	-	Windows 10	-	2
164	КС	15	10.0.15.19	-	-	-	Windows 10	-	2
165	КС	15	10.0.15.20	-	-	-	Windows 10	-	2
166	КС	15	10.0.15.21	-	-	-	Windows 10	-	2
167	КС	15	10.0.15.22	-	-	-	Windows 10	-	2
168	КС	15	10.0.15.23	-	-	-	Windows 10	-	2
169	КС	15	10.0.15.24	-	-	-	Windows 10	-	2
170	КС	16	10.0.16.10	-	-	-	Windows 10	-	2
171	КС	16	10.0.16.11	-	-	-	Windows 10	-	2
172	КС	16	10.0.16.12	-	-	-	Windows 10	-	2
173	КС	16	10.0.16.13	-	-	-	Windows 10	-	2
174	КС	16	10.0.16.14	-	-	-	Windows 10	-	2
175	КС	16	10.0.16.15	-	-	-	Windows 10	-	2
176	КС	16	10.0.16.16	-	-	-	Windows 10	-	2
177	КС	16	10.0.16.17	-	-	-	Windows 10	-	2
178	КС	16	10.0.16.18	-	-	-	Windows 10	-	2
179	КС	16	10.0.16.19	-	-	-	Windows 10	-	2
180	КС	16	10.0.16.20	-	-	-	Windows 10	-	2
181	КС	16	10.0.16.21	-	-	-	Windows 10	-	2
182	КС	16	10.0.16.22	-	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
183	КС	16	10.0.16.23	-	-	-	Windows 10	-	2
184	КС	17	10.0.17.10	-	-	-	Windows 10	-	2
185	КС	17	10.0.17.11	-	-	-	Windows 10	-	2
186	КС	17	10.0.17.12	-	-	-	Windows 10	-	2
187	КС	17	10.0.17.13	-	-	-	Windows 10	-	2
188	КС	17	10.0.17.14	-	-	-	Windows 10	-	2
189	КС	17	10.0.17.15	-	-	-	Windows 10	-	2
190	КС	17	10.0.17.16	-	-	-	Windows 10	-	2
191	КС	17	10.0.17.17	-	-	-	Windows 10	-	2
192	КС	17	10.0.17.18	-	-	-	Windows 10	-	2
193	КС	17	10.0.17.19	-	-	-	Windows 10	-	2
194	КС	17	10.0.17.20	-	-	-	Windows 10	-	2
195	КС	17	10.0.17.21	-	-	-	Windows 10	-	2
196	КС	17	10.0.17.22	-	-	-	Windows 10	-	2
197	КС	17	10.0.17.23	-	-	-	Windows 10	-	2
198	КС	18	10.0.18.10	-	-	-	Windows 10	-	2
199	КС	18	10.0.18.11	+	-	-	Windows 10	-	2
200	КС	18	10.0.18.12	-	-	-	Windows 10	-	2
201	КС	18	10.0.18.13	-	-	-	Windows 10	-	2
202	КС	18	10.0.18.14	-	-	-	Windows 10	-	2
203	КС	18	10.0.18.15	-	-	-	Windows 10	-	2
204	КС	18	10.0.18.16	-	-	-	Windows 10	-	2
205	КС	18	10.0.18.17	-	-	-	Windows 10	-	2
206	КС	18	10.0.18.18	-	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
207	КС	18	10.0.18.19	-	-	-	Windows 10	-	2
208	КС	18	10.0.18.20	-	-	-	Windows 10	-	2
209	КС	18	10.0.18.21	-	-	-	Windows 10	-	2
210	КС	18	10.0.18.22	-	-	-	Windows 10	-	2
211	КС	18	10.0.18.23	-	-	-	Windows 10	-	2
212	КС	18	10.0.18.24	-	-	-	Windows 10	-	2
213	КС	19	10.0.19.10	-	-	-	Windows 10	-	2
214	КС	19	10.0.19.11	-	-	-	Windows 10	-	2
215	КС	19	10.0.19.12	-	-	-	Windows 10	-	2
216	КС	19	10.0.19.13	-	-	-	Windows 10	-	2
217	КС	19	10.0.19.14	-	-	-	Windows 10	-	2
218	КС	19	10.0.19.15	-	-	-	Windows 10	-	2
219	КС	19	10.0.19.16	-	-	-	Windows 10	-	2
220	КС	20	10.0.20.10	-	-	-	Windows 10	-	2
221	КС	20	10.0.20.11	-	-	-	Windows 10	-	2
222	КС	20	10.0.20.12	-	-	-	Windows 10	-	2
223	КС	20	10.0.20.13	-	-	-	Windows 10	-	2
224	КС	20	10.0.20.14	-	-	-	Windows 10	-	2
225	КС	20	10.0.20.15	-	-	-	Windows 10	-	2
226	КС	20	10.0.20.16	-	-	-	Windows 10	-	2
227	КС	20	10.0.20.17	-	-	-	Windows 10	-	2
228	КС	21	10.0.21.10	-	-	-	Windows 10	-	2
229	КС	21	10.0.21.11	+	-	-	Windows 10	-	2
230	КС	21	10.0.21.12	-	-	-	Windows 10	-	2

Продовження таблиці В.1

1	2	3	4	5	6	7	8	9	10
231	КС	21	10.0.21.13	-	-	-	Windows 10	-	2
232	КС	21	10.0.21.14	-	-	-	Windows 10	-	2
233	КС	21	10.0.21.15	-	-	-	Windows 10	-	2
234	КС	21	10.0.21.16	-	-	-	Windows 10	-	2
235	КС	21	10.0.21.17	-	-	-	Windows 10	-	2
236	КС	21	10.0.21.18	-	-	-	Windows 10	-	2
237	КС	22	10.0.22.10	-	-	-	Windows 10	-	3
238	КС	22	10.0.22.11	-	-	-	Windows 10	-	3
239	КС	22	10.0.22.12	-	-	-	Windows 10	-	3
240	КС	22	10.0.22.13	-	-	-	Windows 10	-	3
241	КС	22	10.0.22.14	-	-	-	Windows 10	-	3
242	КС	22	10.0.22.15	-	-	-	Windows 10	-	3
243	КС	22	10.0.22.16	-	-	-	Windows 10	-	3
244	КС	22	10.0.22.17	-	-	-	Windows 10	-	3
245	КС	22	10.0.22.18	-	-	-	Windows 10	-	3
246	КС	23	10.0.23.10	-	-	-	Windows 10	-	3
247	КС	23	10.0.23.11	-	-	-	Windows 10	-	3
248	КС	23	10.0.23.12	-	-	-	Windows 10	-	3
249	КС	23	10.0.23.13	-	-	-	Windows 10	-	3
250	КС	23	10.0.23.14	-	-	-	Windows 10	-	3
251	КС	23	10.0.23.15	-	-	-	Windows 10	-	3
252	КС	23	10.0.23.16	-	-	-	Windows 10	-	3
253	КС	23	10.0.23.17	-	-	-	Windows 10	-	3
254	КС	23	10.0.23.18	-	-	-	Windows 10	-	3

Кінець таблиці В.1

1	2	3	4	5	6	7	8	9	10
255	КС	23	10.0.23.19	-	-	-	Windows 10	-	3
256	КС	23	10.0.23.20	-	-	-	Windows 10	-	3
257	КС	23	10.0.23.21	-	-	-	Windows 10	-	3
258	КС	23	10.0.23.22	-	-	-	Windows 10	-	3
259	КС	23	10.0.23.23	-	-	-	Windows 10	-	3
260	КС	24	10.0.24.10	-	-	-	Windows 10	-	3
261	КС	24	10.0.24.11	-	-	-	Windows 10	-	3
262	КС	24	10.0.24.12	-	-	-	Windows 10	-	3
263	КС	24	10.0.24.13	-	-	-	Windows 10	-	3
264	КС	24	10.0.24.14	-	-	-	Windows 10	-	3
265	КС	24	10.0.24.15	-	-	-	Windows 10	-	3
266	КС	25	10.0.25.10	-	-	-	Windows 10	-	3
267	КС	25	10.0.25.11	-	-	-	Windows 10	-	3
268	КС	25	10.0.25.12	-	-	-	Windows 10	-	3
269	КС	25	10.0.25.13	-	-	-	Windows 10	-	3
270	КС	25	10.0.25.14	-	-	-	Windows 10	-	3
271	КС	25	10.0.25.15	-	-	-	Windows 10	-	3
272	КС	25	10.0.25.16	-	-	-	Windows 10	-	3
273	КС	25	10.0.25.17	-	-	-	Windows 10	-	3
274	КС	25	10.0.25.18	-	-	-	Windows 10	-	3
275	КС	25	10.0.25.19	-	-	-	Windows 10	-	3

Таблиця В.2 - Координати розміщення вузлів корпоративних мереж (фрагмент)

Номер компоненти системи	Номер сегменту: 1..25	IP-адреса вузла	x_{1,R_m}	x_{2,R_m}	R_m	Номер дії: 1..11
1	2	3	4	5	6	7
1	1	10.0.1.10	0,540302306	0,841470985	1	3
2	1	10.0.1.11	-0,432177945	0,901788348	1	6
3	1	10.0.1.12	-0,994367461	0,105987512	1	5
4	1	10.0.1.13	-0,612548239	-0,790433207	1	7
5	1	10.0.1.14	0,350797342	-0,9364514	1	3
6	1	10.0.1.15	0,981111354	-0,193443817	1	9
7	1	10.0.1.16	0,680001394	0,733210819	1	4
8	1	10.0.1.17	-0,266671985	0,963787348	1	7
9	2	10.0.2.10	-0,832293673	1,818594854	2	7
10	2	10.0.2.11	-1,694459048	1,062454016	2	9
11	2	10.0.2.12	-1,999537926	-0,042989324	2	11
12	2	10.0.2.13	-1,64722967	-1,134299084	2	5
13	2	10.0.2.14	-0,753362499	-1,852685873	2	7
14	2	10.0.2.15	0,388187304	-1,961966008	2	8
15	2	10.0.2.16	1,402112951	-1,42621151	2	10
16	2	10.0.2.17	1,955066589	-0,421562132	2	11
17	2	10.0.2.18	1,865253914	0,721684029	2	8
18	2	10.0.2.19	1,162202595	1,627662474	2	10
19	2	10.0.2.20	0,077054481	1,9985151	2	7
20	2	10.0.2.21	-1,033426798	1,712316867	2	6
21	2	10.0.2.22	-1,804148839	0,863161031	2	11
22	2	10.0.2.23	-1,981721724	-0,269775847	2	7
23	3	10.0.3.10	-2,96997749	0,423360024	3	11
24	3	10.0.3.11	-2,714316536	-1,277687653	3	3
25	3	10.0.3.12	-1,566271025	-2,558670568	3	9
26	3	10.0.3.13	0,096716667	-2,998440576	3	10
27	3	10.0.3.14	1,727906867	-2,4524147	3	1
28	3	10.0.3.15	2,791013947	-1,100109608	3	9
29	3	10.0.3.16	2,936520697	0,613877997	3	1
30	3	10.0.3.17	2,116588933	2,126041225	3	3
31	3	10.0.3.18	0,600787231	2,939226889	3	3
32	3	10.0.3.19	-1,112535004	2,786084325	3	9
33	3	10.0.3.20	-2,460089634	1,716962141	3	10
34	3	10.0.3.21	-2,998841763	0,08335513	3	7
35	3	10.0.3.22	-2,551666111	-1,577656508	3	10
36	3	10.0.3.23	-1,265580404	-2,719982765	3	8
37	4	10.0.4.10	-2,614574483	-2,614574483	4	1
38	4	10.0.4.11	-0,521885861	-0,521885861	4	1
39	4	10.0.4.12	1,742382929	1,742382929	4	1
40	4	10.0.4.13	3,43380931	3,43380931	4	8
41	4	10.0.4.14	3,996303836	3,996303836	4	7

Продовження таблиці В.2

1	2	3	4	5	6	7
42	4	10.0.4.15	3,244935448	3,244935448	4	4
43	4	10.0.4.16	1,426731175	1,426731175	4	10
44	4	10.0.4.17	-0,860538829	-0,860538829	4	4
45	4	10.0.4.18	-2,864889891	-2,864889891	4	2
46	4	10.0.4.19	-3,927352465	-3,927352465	4	11
47	4	10.0.4.20	-3,698621234	-3,698621234	4	6
48	4	10.0.4.21	-2,253896057	-2,253896057	4	11
49	4	10.0.4.22	-0,068158544	-0,068158544	4	11
50	4	10.0.4.23	2,139987421	2,139987421	4	8
51	5	10.0.5.10	1,418310927	-4,794621373	5	4
52	5	10.0.5.11	3,818920559	-3,227358945	5	11
53	5	10.0.5.12	4,963985493	-0,59903925	5	4
54	5	10.0.5.13	4,477043275	2,226226295	5	7
55	5	10.0.5.14	2,518185666	4,319576479	5	4
56	5	10.0.5.15	-0,268574649	4,992781555	5	1
57	5	10.0.5.16	-2,967035803	4,024512212	5	11
58	5	10.0.5.17	-4,690026006	1,7331059	5	5
59	5	10.0.5.18	-4,871078587	-1,128092814	5	1
60	5	10.0.5.19	-3,450668975	-3,618408991	5	11
61	5	10.0.5.20	-0,895784563	-4,919102562	5	3
62	5	10.0.5.21	1,953606514	-4,602545121	5	10
63	5	10.0.5.22	4,160711305	-2,772811107	5	5
64	5	10.0.5.23	4,999901018	-0,031461302	5	5
65	6	10.0.6.10	5,76102172	-1,676492989	6	11
66	6	10.0.6.11	-2,593067669	5,410730086	6	1
67	6	10.0.6.12	-2,30219067	-5,540750682	6	1
68	6	10.0.6.13	5,663904835	1,979944954	6	6
69	6	10.0.6.14	-5,252756754	2,899749382	6	7
70	6	10.0.6.15	1,342621981	-5,847851419	6	3
71	6	10.0.6.16	3,461865351	4,900559998	6	9
72	6	10.0.6.17	-5,960321515	-0,688888553	6	10
73	6	10.0.6.18	4,488465205	-3,981668005	6	6
74	7	10.0.7.10	5,27731578	4,598906191	7	10
75	7	10.0.7.11	1,626513714	6,808410471	7	3
76	7	10.0.7.12	-2,628762752	6,487650299	7	9
77	8	10.0.8.10	-1,16400027	7,914865973	8	3
78	8	10.0.8.11	-0,883097951	-7,951109231	8	4
79	8	10.0.8.12	2,872354295	7,46656419	8	9
80	8	10.0.8.13	-4,673474813	-6,492967979	8	2
81	8	10.0.8.14	6,168488229	5,094090005	8	9
82	9	10.0.9.10	-8,200172357	3,709066367	9	5
83	9	10.0.9.11	-5,688675415	-6,974164611	9	10
84	9	10.0.9.12	5,281322906	-7,287498086	9	11
85	9	10.0.9.13	8,398513003	3,234962032	9	7
86	9	10.0.9.14	-0,972060508	8,947351472	9	8
87	9	10.0.9.15	-8,897275557	1,35590843	9	5

Продовження таблиці В.2

1	2	3	4	5	6	7
88	9	10.0.9.16	-3,593116089	-8,251637218	9	3
89	9	10.0.9.17	7,053653947	-5,589809119	9	11
90	9	10.0.9.18	7,212333648	5,383515891	9	11
91	9	10.0.9.19	-3,35301811	8,35208175	9	4
92	9	10.0.9.20	-8,932761402	-1,098077287	9	8
93	10	10.0.10.10	-8,390715291	-5,440211109	10	8
94	10	10.0.10.11	-4,023018967	-9,155070638	10	11
95	10	10.0.10.12	1,667323391	-9,860021943	10	8
96	10	10.0.10.13	6,80950063	-7,323298517	10	10
97	10	10.0.10.14	9,712921582	-2,378897717	10	1
98	10	10.0.10.15	9,423029919	3,347612155	10	4
99	10	10.0.10.16	6,035133187	7,97352917	10	1
100	10	10.0.10.17	0,663068584	9,977992787	10	5
101	10	10.0.10.18	-4,926992763	8,701996456	10	11
102	10	10.0.10.19	-8,897209039	4,565048884	10	4
103	10	10.0.10.20	-9,942294127	-1,072747641	10	11
104	10	10.0.10.21	-7,718655897	-6,357857434	10	3
105	10	10.0.10.22	-2,957358835	-9,552697458	10	6
106	10	10.0.10.23	2,776227687	-9,606901677	10	4
107	11	10.0.11.10	0,048682678	-10,99989227	11	4
108	11	10.0.11.11	9,385421185	-5,737061023	11	7
109	11	10.0.11.12	9,812053568	4,972283659	11	6
110	11	10.0.11.13	0,923553804	10,9611609	11	9
111	11	10.0.11.14	-8,841727325	6,543994033	11	10
112	11	10.0.11.15	-10,21306137	-4,085752984	11	5
113	11	10.0.11.16	-1,888564097	-10,8366658	11	1
114	11	10.0.11.17	8,228852876	-7,299724676	11	10
115	12	10.0.12.10	10,1262475	-6,438875016	12	11
116	12	10.0.12.11	-4,79982378	-10,99825857	12	1
117	12	10.0.12.12	-11,60700706	3,045880353	12	10
118	12	10.0.12.13	1,219028444	11,9379215	12	10
119	12	10.0.12.14	11,98308087	0,637003046	12	1
120	12	10.0.12.15	2,477786752	-11,74140421	12	5
121	12	10.0.12.16	-11,21867647	-4,259260292	12	2
122	12	10.0.12.17	-5,938780975	10,42741006	12	8
123	12	10.0.12.18	9,386545319	7,476146532	12	8
124	12	10.0.12.19	8,834557425	-8,120997174	12	1
125	12	10.0.12.20	-6,661058735	-9,981497709	12	6
126	12	10.0.12.21	-10,88951336	5,041676187	12	8
127	13	10.0.13.10	11,79680816	5,462171479	13	9
128	13	10.0.13.11	6,857138904	11,0444396	13	7
129	13	10.0.13.12	-0,336948651	12,99563256	13	5
130	13	10.0.13.13	-7,420257758	10,67425758	13	5
131	13	10.0.13.14	-12,06401225	4,843512001	13	1
132	13	10.0.13.15	-12,74148716	-2,579632714	13	10
133	13	10.0.13.16	-9,229949393	-9,154672807	13	1

Продовження таблиці В.2

1	2	3	4	5	6	7
134	13	10.0.13.17	-2,683885548	-12,71993547	13	1
135	13	10.0.13.18	4,744558073	-12,10327099	13	6
136	13	10.0.13.19	10,61313559	-7,507419859	13	7
137	13	10.0.13.20	12,99243756	-0,443357865	13	4
138	13	10.0.13.21	11,10022201	6,766466683	13	5
139	13	10.0.13.22	5,558591738	11,75168319	13	7
140	13	10.0.13.23	-1,810534097	12,8733044	13	2
141	14	10.0.14.10	1,914321055	13,86850298	14	2
142	14	10.0.14.11	-6,018543467	12,64029804	14	3
143	14	10.0.14.12	-11,97269431	7,256348319	14	2
144	14	10.0.14.13	-13,99058814	-0,513267592	14	4
145	14	10.0.14.14	-11,40880294	-8,114136761	14	5
146	14	10.0.14.15	-5,076151029	-13,04732504	14	10
147	14	10.0.14.16	2,925384649	-13,69095047	14	6
148	14	10.0.14.17	9,965143005	-9,83340861	14	11
149	14	10.0.14.18	13,72866603	-2,742941676	14	3
150	14	10.0.14.19	12,97862205	5,249320889	14	11
151	14	10.0.14.20	7,961602657	11,51576672	14	8
152	14	10.0.14.21	0,327050945	13,9961794	14	1
153	14	10.0.14.22	-7,415025151	11,87507482	14	6
154	14	10.0.14.23	-12,71926695	5,849807539	14	8
155	15	10.0.15.10	-11,39531869	9,754317602	15	7
156	15	10.0.15.11	3,999643985	14,45693079	15	11
157	15	10.0.15.12	14,78844872	2,510335505	15	11
158	15	10.0.15.13	8,546255014	-12,32726755	15	1
159	15	10.0.15.14	-7,538164787	-12,96827173	15	9
160	15	10.0.15.15	-14,94131753	1,325530292	15	9
161	15	10.0.15.16	-5,137421687	14,09279598	15	1
162	15	10.0.15.17	10,58294466	10,63020612	15	9
163	15	10.0.15.18	14,11554761	-5,074575413	15	10
164	15	10.0.15.19	1,392093283	-14,93526285	15	3
165	15	10.0.15.20	-12,93455412	-7,595874524	15	1
166	15	10.0.15.21	-12,36522594	8,491241816	15	10
167	15	10.0.15.22	2,444415592	14,79948757	15	1
168	15	10.0.15.23	14,43896549	4,06402208	15	8
169	15	10.0.15.24	9,804996508	-11,35174187	15	7
170	16	10.0.16.10	-15,32255169	-4,606453067	16	6
171	16	10.0.16.11	-10,27336304	-12,26613272	16	7
172	16	10.0.16.12	-1,846605705	-15,89308175	16	9
173	16	10.0.16.13	7,187259309	-14,29486984	16	10
174	16	10.0.16.14	13,85817249	-7,996940361	16	8
175	16	10.0.16.15	15,97294089	0,930139397	16	9
176	16	10.0.16.16	12,8362931	9,551417668	16	8
177	16	10.0.16.17	5,479463324	15,03248089	16	6
178	16	10.0.16.18	-3,678847006	15,57132251	16	8
179	16	10.0.16.19	-11,62766455	10,99078783	16	2

Продовження таблиці В.2

1	2	3	4	5	6	7
180	16	10.0.16.20	-15,75366035	2,79681705	16	4
181	16	10.0.16.21	-14,70033272	-6,316661939	16	2
182	16	10.0.16.22	-8,813983679	-13,35341498	16	10
183	16	10.0.16.23	-0,029865415	-15,99997213	16	6
184	17	10.0.17.10	-4,677776747	-16,34375736	17	5
185	17	10.0.17.11	5,069052606	-16,2266665	17	1
186	17	10.0.17.12	13,14933194	-10,7747422	17	9
187	17	10.0.17.13	16,90651166	-1,780411026	17	2
188	17	10.0.17.14	15,10534557	7,799265035	17	3
189	17	10.0.17.15	8,338002198	14,81478044	17	6
190	17	10.0.17.16	-1,1706222	16,95964751	17	5
191	17	10.0.17.17	-10,29438169	13,52869933	17	9
192	17	10.0.17.18	-16,03366221	5,649927097	17	7
193	17	10.0.17.19	-16,50156322	-4,086368982	17	7
194	17	10.0.17.20	-11,54425312	-12,47919147	17	2
195	17	10.0.17.21	-2,791544416	-16,76923611	17	3
196	17	10.0.17.22	6,878939022	-15,54606696	17	8
197	17	10.0.17.23	14,28783695	-9,211824759	17	3
198	18	10.0.18.10	11,88570075	-13,51777044	18	11
199	18	10.0.18.11	-7,19973567	-16,49738786	18	2
200	18	10.0.18.12	-17,99365421	-0,477920772	18	4
201	18	10.0.18.13	-8,06532509	16,09193994	18	5
202	18	10.0.18.14	11,15137103	14,129647	18	3
203	18	10.0.18.15	17,52568007	-4,104940671	18	8
204	18	10.0.18.16	3,716680128	-17,61210631	18	5
205	18	10.0.18.17	-14,3726047	-10,83643088	18	4
206	18	10.0.18.18	-15,90979452	8,41893333	18	2
207	18	10.0.18.19	0,87540301	17,97870044	18	1
208	18	10.0.18.20	16,65244968	6,833441283	18	3
209	18	10.0.18.21	13,25183614	-12,18149576	18	4
210	18	10.0.18.22	-5,41014828	-17,16771084	18	2
211	18	10.0.18.23	-17,84157879	-2,382869326	18	4
212	18	10.0.18.24	-9,725898082	15,14618455	18	7
213	19	10.0.19.10	18,78538775	2,847666984	19	1
214	19	10.0.19.11	4,834262169	18,37470841	19	3
215	19	10.0.19.12	-14,95176321	11,72368444	19	3
216	19	10.0.19.13	-16,69117936	-9,077694169	19	10
217	19	10.0.19.14	1,715469356	-18,9223985	19	7
218	19	10.0.19.15	18,05156595	-5,927981676	19	1
219	19	10.0.19.16	12,59962644	14,22144204	19	1
220	20	10.0.20.10	8,161641236	18,25890501	20	8
221	20	10.0.20.11	-11,24907702	16,53657359	20	2
222	20	10.0.20.12	-19,98041627	-0,884853562	20	4
223	20	10.0.20.13	-9,743226996	-17,46623966	20	1
224	20	10.0.20.14	9,7437535	-17,46594594	20	8
225	20	10.0.20.15	19,98044293	-0,884251265	20	1

Продовження таблиці В.2

1	2	3	4	5	6	7
226	20	10.0.20.16	11,24857854	16,53691268	20	3
227	20	10.0.20.17	-8,162191635	18,25865898	20	7
228	21	10.0.21.10	-11,50231446	17,56976841	21	11
229	21	10.0.21.11	-5,420134355	-0,557574235	21	5
230	21	10.0.21.12	18,73210227	9,492541531	21	8
231	21	10.0.21.13	-19,5661696	7,626598673	21	1
232	21	10.0.21.14	7,366744184	-19,6654794	21	7
233	21	10.0.21.15	9,739845423	18,60471476	21	9
234	21	10.0.21.16	-20,35849144	-5,150905391	21	5
235	21	10.0.21.17	17,41586021	-11,73404504	21	11
236	21	10.0.21.18	-2,872108667	20,8026679	21	3
237	22	10.0.22.10	-21,99913818	-0,194728804	22	11
238	22	10.0.22.11	14,81715757	-16,26197533	22	4
239	22	10.0.22.12	2,234885481	21,88618941	22	9
240	22	10.0.22.13	-17,79821795	-12,93149016	22	9
241	22	10.0.22.14	21,50573248	-4,63718345	22	6
242	22	10.0.22.15	-10,88776512	19,11691844	22	2
243	22	10.0.22.16	-6,982802553	-20,86241761	22	6
244	22	10.0.22.17	20,20195872	8,710962283	22	4
245	22	10.0.22.18	-19,96410783	9,243073009	22	11
246	23	10.0.23.10	-12,25515947	-19,4630693	23	4
247	23	10.0.23.11	0,450761563	-22,99558249	23	6
248	23	10.0.23.12	13,00848593	-18,96784895	23	8
249	23	10.0.23.13	21,28941656	-8,704064725	23	3
250	23	10.0.23.14	22,57103581	4,421350727	23	2
251	23	10.0.23.15	16,43198635	16,0931608	23	7
252	23	10.0.23.16	4,890600542	22,47403004	23	5
253	23	10.0.23.17	-8,258665685	21,4661231	23	1
254	23	10.0.23.18	-18,69273432	13,40080907	23	11
255	23	10.0.23.19	-22,98120157	0,929717405	23	5
256	23	10.0.23.20	-19,7141501	-11,84703701	23	9
257	23	10.0.23.21	-9,965686864	-20,7288467	23	11
258	23	10.0.23.22	3,059190505	-22,7956433	23	7
259	23	10.0.23.23	15,07829927	-17,36792709	23	9
260	24	10.0.24.10	10,18029618	-21,73388069	24	3
261	24	10.0.24.11	-16,32056389	17,59656768	24	9
262	24	10.0.24.12	20,90753762	-11,78451825	24	2
263	24	10.0.24.13	-23,50465655	4,850888624	24	2
264	24	10.0.24.14	23,86474237	2,54441967	24	3
265	24	10.0.24.15	-21,95352427	-9,697565267	24	8
266	25	10.0.25.10	24,7800703	-3,308793752	25	8
267	25	10.0.25.11	-6,452540898	-24,15294425	25	5
268	25	10.0.25.12	-23,12868841	9,490193476	25	8
269	25	10.0.25.13	12,37180551	21,72414391	25	8
270	25	10.0.25.14	19,96240465	-15,04999669	25	5
271	25	10.0.25.15	-17,48073142	-17,87243769	25	10

Кінець таблиці В.2

1	2	3	4	5	6	7
272	25	10.0.25.16	-15,48860688	19,62404283	25	10
273	25	10.0.25.17	21,44468998	12,85010784	25	4
274	25	10.0.25.18	10,00032357	-22,91273726	25	11
275	25	10.0.25.19	-24,00404659	-6,986111045	25	4

Таблиця В.3 - Координати слідів спіралей

Номер компоненти системи	Номер сегменту: 1..25	IP-адреса вузла	x_{1,R_m}	x_{2,R_m}	t, мс	R_m	Номер дії: 1..11
1	2	3	4	5	6	7	8
Слід 1							
249	23	10.0.23.13	21,28941656	-8,704064725	18432	23	3
237	21	10.0.21.17	17,41586021	-11,73404504	39785	21	11
219	19	10.0.19.14	1,715469356	-18,9223985	61204	19	7
198	17	10.0.17.22	6,878939022	-15,54606696	75519	17	8
174	16	10.0.16.12	-1,846605705	-15,89308175	104883	16	9
153	14	10.0.14.20	7,961602657	11,51576672	129447	14	8
131	13	10.0.13.12	-0,336948651	12,99563256	171902	13	5
Слід 2							
216	19	10.0.19.13	-16,69117936	-9,077694169	236926	19	10
207	18	10.0.18.19	0,87540301	17,97870044	301949	18	1
152	14	10.0.14.21	0,327050945	13,9961794	366973	14	1
129	13	10.0.13.12	-0,336948651	12,99563256	431996	13	5
112	11	10.0.11.15	-10,21306137	-4,085752984	497020	11	5
Слід 3							
215	19	10.0.19.12	-14,95176321	11,72368444	562043	19	3
181	16	10.0.16.21	-14,70033272	-6,316661939	627067	16	2
142	14	10.0.14.11	-6,018543467	12,64029804	692090	14	3
Слід 4							
265	24	10.0.24.15	-21,95352427	-9,697565267	757114	24	8
242	22	10.0.22.15	-10,88776512	19,11691844	822137	22	2
224	20	10.0.20.14	9,7437535	-17,46594594	887161	20	8
185	17	10.0.17.11	5,069052606	-16,2266665	952184	17	1
171	16	10.0.16.11	-10,27336304	-12,26613272	1017208	16	7
30	3	10.0.3.17	2,116588933	2,126041225	1082231	3	3
Слід 5							
224	20	10.0.20.14	9,7437535	-17,46594594	1147255	20	8
204	18	10.0.18.16	3,716680128	-17,61210631	1212278	18	5
174	16	10.0.16.14	13,85817249	-7,996940361	1277302	16	8
122	12	10.0.12.17	-5,938780975	10,42741006	1342325	12	8

Продовження таблиці В.3

1	2	3	4	5	6	7	8
71	6	10.0.6.16	3,461865351	4,900559998	1407349	6	9
57	5	10.0.5.16	-2,967035803	4,024512212	1472372	5	11
12	2	10.0.2.13	-1,64722967	-1,134299084	1537396	2	5
7	1	10.0.1.16	0,680001394	0,733210819	1602419	1	4
Слід 6							
226	20	10.0.20.16	11,24857854	16,53691268	1667443	20	3
211	18	10.0.18.23	-17,84157879	-2,382869326	1732466	18	4
145	14	10.0.14.14	-11,40880294	-8,114136761	1797490	14	5
63	5	10.0.5.22	4,160711305	-2,772811107	1862513	5	5
2	1	10.0.1.11	-0,432177945	0,901788348	1927537	1	6
Слід 7							
269	25	10.0.25.13	12,37180551	21,72414391	1992560	25	8
258	23	10.0.23.22	3,059190505	-22,7956433	2057584	23	7
192	17	10.0.17.18	-16,03366221	5,649927097	2122607	17	7
185	17	10.0.17.11	5,069052606	-16,2266665	2187631	17	1
180	16	10.0.16.20	-15,75366035	2,79681705	2252654	16	4
126	12	10.0.12.21	-10,88951336	5,041676187	2317678	12	8
77	8	10.0.8.10	-1,16400027	7,914865973	2382701	8	3
Слід 8							
221	20	10.0.20.11	-11,24907702	16,53657359	2447725	20	2
202	18	10.0.18.14	11,15137103	14,129647	2512748	18	3
173	16	10.0.16.13	7,187259309	-14,29486984	2577772	16	10
168	15	10.0.15.23	14,43896549	4,06402208	2642795	15	8
137	13	10.0.13.20	12,99243756	-0,443357865	2707819	13	4
129	13	10.0.13.12	-0,336948651	12,99563256	2772842	13	5
91	9	10.0.9.19	-3,35301811	8,35208175	2837866	9	4
73	6	10.0.6.18	4,488465205	-3,981668005	2902889	6	6
27	3	10.0.3.14	1,727906867	-2,4524147	2967913	3	1
Слід 9							
236	21	10.0.21.18	-2,872108667	20,8026679	3032936	21	3
226	20	10.0.20.16	11,24857854	16,53691268	3097960	20	3
221	20	10.0.20.11	-11,24907702	16,53657359	3162983	20	2
193	17	10.0.17.19	-16,50156322	-4,086368982	3228007	17	7
83	9	10.0.9.11	-5,688675415	-6,974164611	3293030	9	10
80	8	10.0.8.13	-4,673474813	-6,492967979	3358054	8	2
69	6	10.0.6.14	-5,252756754	2,899749382	3423077	6	7
Слід 10							
255	23	10.0.23.19	-22,98120157	0,929717405	3488101	23	5
226	20	10.0.20.16	11,24857854	16,53691268	3553124	20	3
183	16	10.0.16.23	-0,029865415	-15,99997213	3618148	16	6
145	14	10.0.14.14	-11,40880294	-8,114136761	3683171	14	5
126	12	10.0.12.21	-10,88951336	5,041676187	3748195	12	8

Продовження таблиці В.3

1	2	3	4	5	6	7	8
Слід 11							
273	25	10.0.25.17	21,44468998	12,85010784	3813218	25	4
249	23	10.0.23.13	21,28941656	-8,704064725	3878242	23	3
218	19	10.0.19.15	18,05156595	-5,927981676	3943265	19	1
197	17	10.0.17.23	14,28783695	-9,211824759	4008289	17	3
193	17	10.0.17.19	-16,50156322	-4,086368982	4073312	17	7
156	15	10.0.15.11	3,999643985	14,45693079	4138336	15	11
94	10	10.0.10.11	-4,023018967	-9,155070638	4203359	10	11
Слід 12							
209	18	10.0.18.21	13,25183614	-12,18149576	4268383	18	4
186	17	10.0.17.12	13,14933194	-10,7747422	4333406	17	9
183	16	10.0.16.23	-0,029865415	-15,99997213	4398430	16	6
150	14	10.0.14.19	12,97862205	5,249320889	4463453	14	11
135	13	10.0.13.18	4,744558073	-12,10327099	4528477	13	6
53	5	10.0.5.12	4,963985493	-0,59903925	4593500	5	4
41	4	10.0.4.14	3,996303836	3,996303836	4658524	4	7
36	3	10.0.3.23	-1,265580404	-2,719982765	4723547	3	8
Слід 13							
268	25	10.0.25.12	-23,12868841	9,490193476	4788571	25	8
251	23	10.0.23.15	16,43198635	16,0931608	4853594	23	7
173	16	10.0.16.13	7,187259309	-14,29486984	4918618	16	10
152	14	10.0.14.21	0,327050945	13,9961794	4983641	14	1
134	13	10.0.13.17	-2,683885548	-12,71993547	5048665	13	1
107	11	10.0.11.10	0,048682678	-10,99989227	5113688	11	4
40	4	10.0.4.13	3,43380931	3,43380931	5178712	4	8
34	3	10.0.3.21	-2,998841763	0,08335513	5243735	3	7
31	3	10.0.3.18	0,600787231	2,939226889	5308759	3	3
26	3	10.0.3.13	0,096716667	-2,998440576	5373782	3	10
Слід 14							
214	19	10.0.19.11	4,834262169	18,37470841	5438806	19	3
193	17	10.0.17.19	-16,50156322	-4,086368982	5503829	17	7
176	16	10.0.16.16	12,8362931	9,551417668	5568853	16	8
162	15	10.0.15.17	10,58294466	10,63020612	5633876	15	9
144	14	10.0.14.13	-13,99058814	-0,513267592	5698900	14	4
115	12	10.0.12.10	10,1262475	-6,438875016	5763923	12	11
75	7	10.0.7.11	1,626513714	6,808410471	5828947	7	3
34	3	10.0.3.21	-2,998841763	0,08335513	5893970	3	7
Слід 15							
269	25	10.0.25.13	12,37180551	21,72414391	5958994	25	8
243	22	10.0.22.16	-6,982802553	-20,86241761	6024017	22	6
203	18	10.0.18.15	17,52568007	-4,104940671	6089041	18	8
199	18	10.0.18.11	-7,19973567	-16,49738786	6154064	18	2

Кінець таблиці В.3

1	2	3	4	5	6	7	8
101	10	10.0.10.18	-4,926992763	8,701996456	6219088	10	11
73	6	10.0.6.18	4,488465205	-3,981668005	6284111	6	6
41	4	10.0.4.14	3,996303836	3,996303836	6349135	4	7
Слід 16							
272	25	10.0.25.16	-15,48860688	19,62404283	6414158	25	10
206	18	10.0.18.18	-15,90979452	8,41893333	6479182	18	2
167	15	10.0.15.22	2,444415592	14,79948757	6544209	15	1
106	10	10.0.10.23	2,776227687	-9,606901677	6609233	10	4
81	8	10.0.8.14	6,168488229	5,094090005	6674256	8	9
72	6	10.0.6.17	-5,960321515	-0,688888553	6739280	6	10
38	4	10.0.4.11	-0,521885861	-0,521885861	6804303	4	1
Слід 17							
221	18	10.0.18.23	-17,84158	-2,382869	6869327	18	4
171	16	10.0.16.11	-10,27336	-12,26613	6931140	16	7
170	16	10.0.16.10	-15,32255	-4,606453	6984023	16	6
115	12	10.0.12.10	10,126248	-6,438875	7029918	12	11
83	9	10.0.9.11	-5,688675	-6,974165	7083376	9	10
32	9	10.0.3.19	-1,112535	2,7860843	7198642	3	9

ДОДАТОК Г

Керівництво користувача

1. ЗАГАЛЬНИЙ ОГЛЯД СИСТЕМИ (*Комп'ютерна програма «Програмне забезпечення функціонування обманних систем в корпоративних мережах з прийняттям рішень на основі популяційних алгоритмів»*)

Обманна система корпоративної мережі - це програмний комплекс для захисту корпоративних мереж від кібератак шляхом створення розподіленої системи обманних станцій.

Основні можливості системи:

1) керування розподіленими станціями обманної системи;
2) автоматична перевірка цілісності станцій;
3) виконання команд на віддалених станціях;
4) динамічне переміщення центру управління;
5) моніторинг подій у реальному часі;
6) генерація детальних звітів про топологію системи.

2. ГОЛОВНЕ ВІКНО ПРОГРАМИ

Головне вікно програми складається з кількох основних областей:

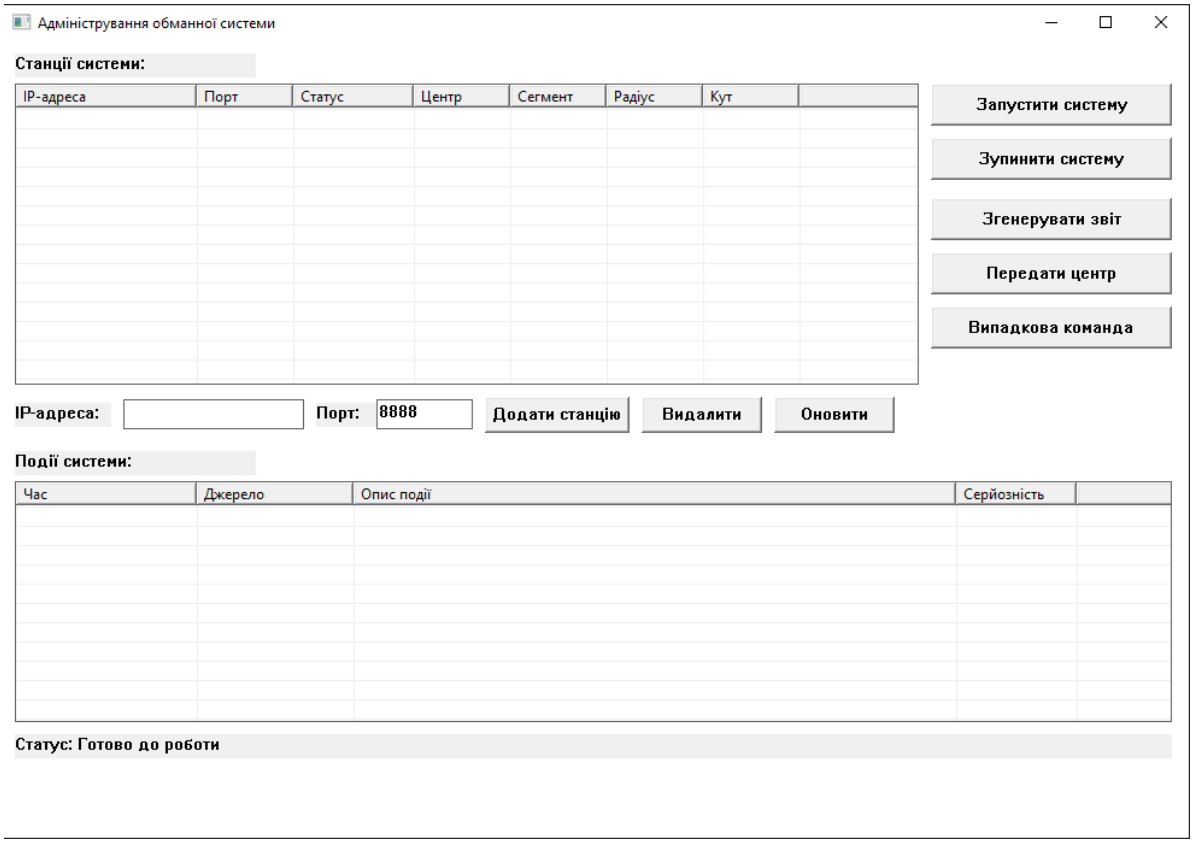


Рисунок Г.1 - Головне вікно програми

Елементи інтерфейсу:

1. Таблиця "Станції системи" - відображає список всіх зареєстрованих станцій.
2. Поля введення IP-адреси та порту - для додавання нових станцій.
3. Кнопки управління станціями - для додавання, видалення та оновлення.
4. Таблиця "Події системи" - журнал подій та дій системи.
5. Рядок стану - відображає поточний статус системи.
6. Панель управління системою (праворуч) - кнопки керування.

3. КЕРУВАННЯ СТАНЦІЯМИ

3.1. Додавання станції

Для додавання нової станції до системи:

1. Введіть IP-адресу станції в поле "IP-адреса" (наприклад, 192.168.1.100)
2. Введіть порт станції в поле "Порт" (за замовчуванням 8888)
3. Натисніть кнопку "Додати станцію"
4. Система додасть станцію до списку та автоматично призначить їй координати

Рисунок Г.2 – Додавання нової станції

3.2. Видалення станції

Для видалення станції з системи:

1. Виберіть станцію в таблиці "Станції системи" (клікніть на рядок)
2. Натисніть кнопку "Видалити"
3. Підтвердіть видалення у діалоговому вікні
4. Станція буде видалена зі списку та бази даних

3.3. Оновлення списку станцій

Кнопка "Оновити" дозволяє оновити відображення списку станцій та подій. Використовуйте цю кнопку для перевірки актуальності даних.

4. КЕРУВАННЯ СИСТЕМОЮ

Рисунок Г.3 – Панель управління системою

4.1. Запуск системи

Кнопка "Запустити систему" ініціалізує роботу обманної системи. При запуску відбувається:

1. Реєстрація поточного комп'ютера як центральної станції.
2. Ініціалізація мережного модуля на порту 8888.
3. Запуск потоків перевірки цілісності станцій.
4. Запуск потоку автоматичного виконання команд.
5. Запуск потоку прийому мережних повідомлень.

4.2. Зупинка системи

Кнопка "Зупинити систему" безпечно завершує роботу системи. При зупинці відбувається:

1. Зупинка всіх робочих потоків.
2. Закриття всіх мережних з'єднань.

3. Збереження поточного стану бази даних станцій.
4. Звільнення системних ресурсів.

4.3. Відправка команд

Кнопка "Надіслати команду" відправляє випадкову команду всім активним станціям. Система випадковим чином обирає одну з 11 доступних функцій:

1. Сканування мережі (SCAN_NETWORK).
2. Логування активності (LOG_ACTIVITY).
3. Перехоплення трафіку (CAPTURE_TRAFFIC).
4. Аналіз патернів (ANALYZE_PATTERNS).
5. Генерація обманки (GENERATE_DECOY).
6. Оновлення правил (UPDATE_RULES).
7. Відправка сповіщення (SEND_ALERT).
8. Ізоляція сегмента (ISOLATE_SEGMENT).
9. Дзеркалювання трафіку (MIRROR_TRAFFIC).
10. Ротація облікових даних (ROTATE_CREDENTIALS).
11. Резервне копіювання стану (BACKUP_STATE).

4.4. Передача центру управління

Функція передачі центру дозволяє динамічно змінювати центральну станцію системи. Це підвищує безпеку системи, роблячи її стійкішою до атак.

Для передачі центру:

1. Виберіть станцію в таблиці "Станції системи".
2. Натисніть кнопку "Передати центр".
3. Підтвердіть дію у діалоговому вікні.
4. Система відправить повідомлення про передачу центру.
5. Нова станція стане центром управління.

5. ГЕНЕРАЦІЯ ЗВІТІВ

Кнопка "Згенерувати звіт" створює детальний звіт про топологію системи у форматі текстового файлу.

Звіт містить:

1. Загальну інформацію про систему.
2. Таблицю всіх зареєстрованих станцій.
3. Координати кожної станції (полярні та декартові).
4. Візуалізацію розміщення станцій на координатній площині.
5. Детальну інформацію по кожному сегменту (квадранту).
6. Легенду та пояснення.

Процес генерації звіту:

1. Натисніть кнопку "Згенерувати звіт".
2. У діалоговому вікні виберіть місце збереження файлу.
3. Введіть ім'я файлу (за замовчуванням: DeceptionSystem_AppendixB.txt).
4. Натисніть "Зберегти".
5. Система згенерує звіт та покаже повідомлення про успішне завершення.

ПРИКЛАД змісту звіту:

═══

ДОДАТОК В

Топологія обманної системи корпоративної мережі

═══

Дата формування: …

Кількість станцій: 4

IP-адреса	Порт	Сегмент	Радіус	Кут	Статус
192.168.1.10	8888	0	1.00	45.0	Активна
192.168.1.20	8888	1	2.00	135.0	Активна
192.168.1.30	8888	2	3.00	225.0	Активна
192.168.1.40	8888	3	1.00	315.0	Активна

Рисунок Г.4 – Таблиця з інформацією про станції

6. МОНІТОРИНГ ПОДІЙ

Таблиця "Події системи" відображає журнал всіх важливих подій, що відбуваються в системі.

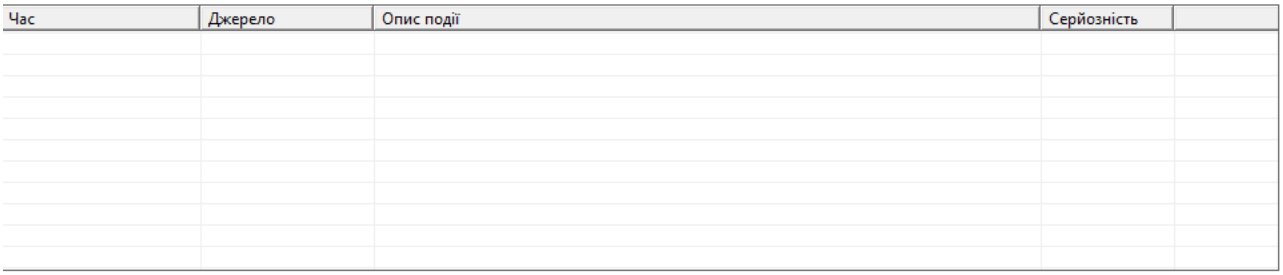


Рисунок Г.5 – Журнал подій системи

Стовпці таблиці подій:

1. Час - дата та час події у форматі YYYY-MM-DD HH:MM:SS.
2. Джерело - IP-адреса станції, що згенерувала подію.
3. Опис - текстовий опис події.
4. Важливість - рівень важливості події (1-10, де 10 - найбільш критична).

Типи подій:

1. Запуск/зупинка системи.
2. Додавання/видалення станцій.
3. Перевірка цілісності станцій.
4. Виконання команд.
5. Передача центру управління.
6. Помилки та попередження.

7. РЯДОК СТАНУ

Рядок стану розташований внизу головного вікна та відображає поточний статус системи та результати останньої виконаної операції.

Рисунок Г.6 – Рядок стану

Приклади повідомлень у рядку стану:

1. "Готово до роботи" - система ініціалізована, але не запущена.
2. "Систему запущено" - система активна та працює.
3. "Станцію додано: 192.168.1.100" - успішно додано нову станцію.
4. "Випадкову команду надіслано всім станціям" - команду відправлено.
5. "Звіт згенеровано" - звіт створено.
6. "Дані оновлено" - список станцій та подій оновлено.

ДОДАТКОВА ІНФОРМАЦІЯ

Структура таблиці станцій

Таблиця "Станції системи" містить наступні колонки:

1. IP-адреса - мережна адреса станції.
2. Порт - мережний порт станції.
3. Центр - позначка "Так" для центральної станції.
4. Статус - "Активна" або "Неактивна".
5. Сегмент - номер квадранта (0-3).
6. Радіус - відстань від центру координат (1-3).
7. Кут - полярний кут розміщення (0-360°).

Автоматичні процеси

Після запуску системи автоматично виконуються наступні процеси:

1. Перевірка цілісності станцій - кожні 30 секунд система перевіряє доступність всіх зареєстрованих станцій.

2. Відправка випадкових команд - система автоматично генерує та відправляє команди станціям з інтервалом 30-120 секунд.

3. Прийом повідомлень - система постійно очікує вхідні повідомлення від станцій.

4. Збереження стану - база даних станцій автоматично зберігається після кожної зміни.

ДОДАТОК Д

Лістинг програмного коду

Файл: DeceptionSystem.h

```
#pragma once
#ifndef DECEPTION_SYSTEM_H
#define DECEPTION_SYSTEM_H

#define NOMINMAX

#include <string>
#include <vector>
#include <map>
#include <memory>
#include <mutex>
#include <chrono>
#include <winsock2.h>
#include <ws2tcpip.h>

#pragma comment(lib, "ws2_32.lib")

// Константи системи
constexpr int DEFAULT_PORT = 8888;
constexpr int MAX_BUFFER_SIZE = 4096;
constexpr int INTEGRITY_CHECK_INTERVAL = 30; // секунди
constexpr int MAX_STATIONS = 100;

// Типи повідомлень
enum class MessageType {
    HANDSHAKE = 1,
    INTEGRITY_CHECK = 2,
    COMMAND_EXECUTE = 3,
    RESPONSE_SUCCESS = 4,
    RESPONSE_FAILURE = 5,
    CENTER_TRANSFER = 6,
    EVENT_REPORT = 7,
    STATION_REGISTER = 8
};

// Функції-дії (1-11)
enum class SystemFunction {
    SCAN_NETWORK = 1,
    LOG_ACTIVITY = 2,
    CAPTURE_TRAFFIC = 3,
    ANALYZE_PATTERNS = 4,
    GENERATE_DECOY = 5,
    UPDATE_RULES = 6,
    SEND_ALERT = 7,
    ISOLATE_SEGMENT = 8,
    MIRROR_TRAFFIC = 9,
    ROTATE_CREDENTIALS = 10,
    BACKUP_STATE = 11
};

// Структура станції
struct Station {
    std::string ipAddress;
    int port;
    bool isCenter;
    bool isActive;
    std::chrono::system_clock::time_point lastSeen;
```

```
    int segmentId; // Для візуалізації на координатній площині
    double angle;  // Кут на колі (0-360)
    double radius; // Радіус кола

    Station() : port(DEFAULT_PORT), isCenter(false), isActive(false),
                segmentId(0), angle(0.0), radius(1.0) {}
};

// Структура повідомлення
struct Message {
    MessageType type;
    std::string senderId;
    std::string receiverId;
    SystemFunction function;
    std::string payload;
    std::chrono::system_clock::time_point timestamp;

    std::string serialize() const;
    static Message deserialize(const std::string& data);
};

// Структура події
struct Event {
    std::string eventId;
    std::string description;
    std::string sourceIp;
    std::chrono::system_clock::time_point timestamp;
    int severity; // 1-10

    std::string toJson() const;
};

// База даних станцій
class StationDatabase {
private:
    std::map<std::string, Station> stations;
    std::mutex dbMutex;
    std::string dbFilePath;

public:
    StationDatabase(const std::string& filePath = "stations.db");

    bool addStation(const std::string& ip, const Station& station);
    bool removeStation(const std::string& ip);
    Station* getStation(const std::string& ip);
    std::vector<Station> getAllStations();
    bool updateStation(const std::string& ip, const Station& station);

    bool saveToFile();
    bool loadFromFile();

    void assignCoordinates(); // Розміщення станцій на координатній площині
};

// Клас для роботи з сокетами
class NetworkManager {
private:
    SOCKET serverSocket;
    bool isInitialized;
    int port;

public:
    NetworkManager(int port = DEFAULT_PORT);
```

```
    ~NetworkManager();

    bool initialize();
    void cleanup();

    bool sendMessage(const std::string& ip, int port, const Message& msg);
    Message receiveMessage(SOCKET socket);

    SOCKET acceptConnection();
    bool connectToStation(const std::string& ip, int port, SOCKET& outSocket);
};

// Генератор звітів
class ReportGenerator {
public:
    static bool generateAppendixB(const std::vector<Station>& stations,
                                  const std::string& outputPath);
    static std::string generateCoordinateVisualization(const std::vector<Station>& stations);
};

#endif // DECEPTION_SYSTEM_H

            Файл: main.cpp
            #define _CRT_SECURE_NO_WARNINGS
#include "DeceptionSystem.h"
#include <iostream>

// Оголошення класу AdminGUI (визначення в AdminGUI.cpp)
class AdminGUI {
private:
    std::string myIpAddress;
public:
    AdminGUI(const std::string& myIp);
    ~AdminGUI();
    bool Create(HINSTANCE hInstance);
    void Run();
};

// Функція для отримання локальної IP-адреси
std::string getLocalIPAddress() {
    WSADATA wsaData;
    WSAStartup(MAKEWORD(2, 2), &wsaData);

    char hostName[256];
    if (gethostname(hostName, sizeof(hostName)) == SOCKET_ERROR) {
        WSACleanup();
        return "127.0.0.1";
    }

    struct addrinfo hints, *result;
    ZeroMemory(&hints, sizeof(hints));
    hints.ai_family = AF_INET;
    hints.ai_socktype = SOCK_STREAM;
    hints.ai_protocol = IPPROTO_TCP;

    if (getaddrinfo(hostName, NULL, &hints, &result) != 0) {
        WSACleanup();
        return "127.0.0.1";
    }

    char ipStr[INET_ADDRSTRLEN];
    struct sockaddr_in* addr = (struct sockaddr_in*)result->ai_addr;
    inet_ntop(AF_INET, &addr->sin_addr, ipStr, INET_ADDRSTRLEN);
```

```
    freeaddrinfo(result);
    WSACleanup();
    return std::string(ipStr);
}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow) {
    // Ініціалізація Winsock
    WSADATA wsaData;
    if (WSAStartup(MAKEWORD(2, 2), &wsaData) != 0) {
        MessageBox(NULL, L"Не вдалось ініціалізувати Winsock", L"Помилка", MB_OK | MB_ICONERROR);
        return 1;
    }

    // Отримання локальної IP-адреси
    std::string myIp = getLocalIPAddress();

    // Створення і запуск GUI
    AdminGUI gui(myIp);

    if (!gui.Create(hInstance)) {
        MessageBox(NULL, L"Не вдалось створити вікно програми", L"Помилка", MB_OK | MB_ICONERROR);
        WSACleanup();
        return 1;
    }

    gui.Run();

    WSACleanup();
    return 0;
}
```

Файл: Message.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include "DeceptionSystem.h"
#include <sstream>
#include <ctime>

// Реалізація Message
std::string Message::serialize() const {
    std::stringstream ss;

    // Формат: TYPE|SENDER|RECEIVER|FUNCTION|PAYLOAD|TIMESTAMP
    ss << static_cast<int>(type) << "|"
       << senderId << "|"
       << receiverId << "|"
       << static_cast<int>(function) << "|"
       << payload << "|"
       << std::chrono::system_clock::to_time_t(timestamp);

    return ss.str();
}

Message Message::deserialize(const std::string& data) {
    Message msg;
    std::stringstream ss(data);
    std::string token;

    try {
        // TYPE
        std::getline(ss, token, '|');
        msg.type = static_cast<MessageType>(std::stoi(token));
```

```
        // SENDER
        std::getline(ss, msg.senderId, '|');

        // RECEIVER
        std::getline(ss, msg.receiverId, '|');

        // FUNCTION
        std::getline(ss, token, '|');
        msg.function = static_cast<SystemFunction>(std::stoi(token));

        // PAYLOAD
        std::getline(ss, msg.payload, '|');

        // TIMESTAMP
        std::getline(ss, token, '|');
        time_t tt = std::stoll(token);
        msg.timestamp = std::chrono::system_clock::from_time_t(tt);

    } catch (...) {
        // Помилка десеріалізації
        msg.type = MessageType::HANDSHAKE;
    }

    return msg;
}

// Реалізація Event
std::string Event::toJson() const {
    std::stringstream ss;

    ss << "{\n"
       << "  \"eventId\": \"" << eventId << "\",\n"
       << "  \"description\": \"" << description << "\",\n"
       << "  \"sourceIp\": \"" << sourceIp << "\",\n"
       << "  \"timestamp\": " << std::chrono::system_clock::to_time_t(timestamp) << ",\n"
       << "  \"severity\": " << severity << "\n"
       << "}";

    return ss.str();
}
```

Файл: NetworkManager.cpp

```
#define _CRT_SECURE_NO_WARNINGS
#include "DeceptionSystem.h"
#include <iostream>

// Реалізація NetworkManager
NetworkManager::NetworkManager(int port) {
    this->port = port;
    serverSocket = INVALID_SOCKET;
    isInitialized = false;
}

NetworkManager::~NetworkManager() {
    cleanup();
}

bool NetworkManager::initialize() {
    WSADATA wsaData;
    int result = WSAStartup(MAKEWORD(2, 2), &wsaData);
    if (result != 0) {
        std::cerr << "WSAStartup failed: " << result << std::endl;
        return false;
```

```
    }

    // Створення сокета
    serverSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (serverSocket == INVALID_SOCKET) {
        std::cerr << "Socket creation failed: " << WSAGetLastError() << std::endl;
        WSACleanup();
        return false;
    }

    // Прив'язка до порту
    sockaddr_in serverAddr;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_addr.s_addr = INADDR_ANY;
    serverAddr.sin_port = htons(static_cast<u_short>(port));

    if (bind(serverSocket, (sockaddr*)&serverAddr, sizeof(serverAddr)) == SOCKET_ERROR) {
        std::cerr << "Bind failed: " << WSAGetLastError() << std::endl;
        closesocket(serverSocket);
        WSACleanup();
        return false;
    }

    // Прослуховування
    if (listen(serverSocket, SOMAXCONN) == SOCKET_ERROR) {
        std::cerr << "Listen failed: " << WSAGetLastError() << std::endl;
        closesocket(serverSocket);
        WSACleanup();
        return false;
    }

    isInitialized = true;
    std::cout << "Network Manager initialized on port " << port << std::endl;
    return true;
}

void NetworkManager::cleanup() {
    if (serverSocket != INVALID_SOCKET) {
        closesocket(serverSocket);
        serverSocket = INVALID_SOCKET;
    }
    if (isInitialized) {
        WSACleanup();
        isInitialized = false;
    }
}

bool NetworkManager::sendMessage(const std::string& ip, int port, const Message& msg) {
    SOCKET clientSocket = INVALID_SOCKET;

    // Створення сокета для з'єднання
    clientSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (clientSocket == INVALID_SOCKET) {
        return false;
    }

    // Підключення до станції
    sockaddr_in serverAddr;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(static_cast<u_short>(port));
    inet_pton(AF_INET, ip.c_str(), &serverAddr.sin_addr);

    if (connect(clientSocket, (sockaddr*)&serverAddr, sizeof(serverAddr)) == SOCKET_ERROR) {
```

```
        closesocket(clientSocket);
        return false;
    }

    // Серіалізація та відправка повідомлення
    std::string serialized = msg.serialize();
    int result = send(clientSocket, serialized.c_str(), static_cast<int>(serialized.length()), 0);

    closesocket(clientSocket);
    return (result != SOCKET_ERROR);
}

Message NetworkManager::receiveMessage(SOCKET socket) {
    char buffer[MAX_BUFFER_SIZE];
    int bytesReceived = recv(socket, buffer, MAX_BUFFER_SIZE - 1, 0);

    if (bytesReceived > 0) {
        buffer[bytesReceived] = '\0';
        return Message::deserialize(std::string(buffer));
    }

    return Message(); // Порожнє повідомлення при помилці
}

SOCKET NetworkManager::acceptConnection() {
    if (!isInitialized || serverSocket == INVALID_SOCKET) {
        return INVALID_SOCKET;
    }

    sockaddr_in clientAddr;
    int clientAddrSize = sizeof(clientAddr);

    SOCKET clientSocket = accept(serverSocket, (sockaddr*)&clientAddr, &clientAddrSize);

    if (clientSocket == INVALID_SOCKET) {
        std::cerr << "Accept failed: " << WSAGetLastError() << std::endl;
    }

    return clientSocket;
}

bool NetworkManager::connectToStation(const std::string& ip, int port, SOCKET& outSocket) {
    outSocket = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);
    if (outSocket == INVALID_SOCKET) {
        return false;
    }

    sockaddr_in serverAddr;
    serverAddr.sin_family = AF_INET;
    serverAddr.sin_port = htons(static_cast<u_short>(port));
    inet_pton(AF_INET, ip.c_str(), &serverAddr.sin_addr);

    if (connect(outSocket, (sockaddr*)&serverAddr, sizeof(serverAddr)) == SOCKET_ERROR) {
        closesocket(outSocket);
        outSocket = INVALID_SOCKET;
        return false;
    }

    return true;
}
```

```
        Файл: StationDatabase.cpp
        #define _CRT_SECURE_NO_WARNINGS
#define NOMINMAX
#include "DeceptionSystem.h"
#include <fstream>
#include <sstream>
#include <algorithm>
#include <cmath>
#include <iostream>

// Реалізація StationDatabase
StationDatabase::StationDatabase(const std::string& filePath) {
    dbFilePath = filePath;
    loadFromFile();
}

bool StationDatabase::addStation(const std::string& ip, const Station& station) {
    std::lock_guard<std::mutex> lock(dbMutex);

    if (stations.find(ip) != stations.end()) {
        return false; // Станція вже існує
    }

    stations[ip] = station;
    assignCoordinates();
    saveToFile();
    return true;
}

bool StationDatabase::removeStation(const std::string& ip) {
    std::lock_guard<std::mutex> lock(dbMutex);

    auto it = stations.find(ip);
    if (it == stations.end()) {
        return false;
    }

    stations.erase(it);
    assignCoordinates();
    saveToFile();
    return true;
}

Station* StationDatabase::getStation(const std::string& ip) {
    std::lock_guard<std::mutex> lock(dbMutex);

    auto it = stations.find(ip);
    if (it == stations.end()) {
        return nullptr;
    }

    return &(it->second);
}

std::vector<Station> StationDatabase::getAllStations() {
    std::lock_guard<std::mutex> lock(dbMutex);

    std::vector<Station> result;
    for (const auto& pair : stations) {
        result.push_back(pair.second);
    }
    return result;
}
```

```
bool StationDatabase::updateStation(const std::string& ip, const Station& station) {
    std::lock_guard<std::mutex> lock(dbMutex);

    auto it = stations.find(ip);
    if (it == stations.end()) {
        return false;
    }

    stations[ip] = station;
    saveToFile();
    return true;
}

bool StationDatabase::saveToFile() {
    std::ofstream file(dbFilePath);
    if (!file.is_open()) {
        return false;
    }

    for (const auto& pair : stations) {
        file << pair.first << "|"
             << pair.second.port << "|"
             << pair.second.isCenter << "|"
             << pair.second.isActive << "|"
             << pair.second.segmentId << "|"
             << pair.second.angle << "|"
             << pair.second.radius << "\n";
    }

    file.close();
    return true;
}

bool StationDatabase::loadFromFile() {
    std::ifstream file(dbFilePath);
    if (!file.is_open()) {
        return false;
    }

    std::lock_guard<std::mutex> lock(dbMutex);
    stations.clear();

    std::string line;
    while (std::getline(file, line)) {
        std::stringstream ss(line);
        std::string ip, token;
        Station station;

        std::getline(ss, ip, '|');
        std::getline(ss, token, '|'); station.port = std::stoi(token);
        std::getline(ss, token, '|'); station.isCenter = (token == "1");
        std::getline(ss, token, '|'); station.isActive = (token == "1");
        std::getline(ss, token, '|'); station.segmentId = std::stoi(token);
        std::getline(ss, token, '|'); station.angle = std::stod(token);
        std::getline(ss, token, '|'); station.radius = std::stod(token);

        station.ipAddress = ip;
        stations[ip] = station;
    }

    file.close();
    return true;
```

```
}

void StationDatabase::assignCoordinates() {
    // Розподіл станцій по сегментах та колах
    int numStations = static_cast<int>(stations.size());
    if (numStations == 0) return;

    // Визначаємо кількість сегментів (4 квадранти)
    const int numSegments = 4;

    // Розподіляємо станції між сегментами
    int stationsPerSegment = (numStations + numSegments - 1) / numSegments;

    int index = 0;
    for (auto& pair : stations) {
        Station& station = pair.second;

        // Визначаємо сегмент (0-3)
        station.segmentId = index / stationsPerSegment;
        if (station.segmentId >= numSegments) {
            station.segmentId = numSegments - 1;
        }

        // Визначаємо радіус (1-3 кола)
        int posInSegment = index % stationsPerSegment;
        station.radius = 1.0 + (posInSegment % 3);

        // Визначаємо кут в межах сегменту
        double segmentStart = station.segmentId * 90.0; // 0, 90, 180, 270
        double segmentRange = 90.0;

        // Уникаємо std::min через конфлікт з макросами Windows
        int remaining = numStations - station.segmentId * stationsPerSegment;
        int stationsInThisSegment = (stationsPerSegment < remaining) ? stationsPerSegment : remaining;
        double angleStep = segmentRange / (stationsInThisSegment + 1);

        station.angle = segmentStart + angleStep * (posInSegment + 1);

        index++;
    }
}

            Файл: ReportGenerator.cpp
            #define _CRT_SECURE_NO_WARNINGS
#include "DeceptionSystem.h"
#include <fstream>
#include <sstream>
#include <iomanip>
#include <cmath>
#include <ctime>

// Константи для візуалізації
const double PI = 3.14159265358979323846;

// Реалізація ReportGenerator
bool ReportGenerator::generateAppendixB(const std::vector<Station>& stations,
                                        const std::string& outputPath) {
    std::ofstream file(outputPath);
    if (!file.is_open()) {
        return false;
    }
```

```
    // Заголовок звіту
    file <<
"═══════════════════════════════════════════════════════════════════════════════\n";
    file << "                    ДОДАТОК В\n";
    file << "      Топологія обманної системи корпоративної мережі\n";
    file <<
"═══════════════════════════════════════════════════════════════════════════════\n\n";

    time_t now = time(0);
    struct tm timeinfo;
    localtime_s(&timeinfo, &now);
    char dateStr[100];
    strftime(dateStr, sizeof(dateStr), "%Y-%m-%d %H:%M:%S", &timeinfo);

    file << "Дата формування: " << dateStr << "\n";
    file << "Кількість станцій: " << stations.size() << "\n\n";

    // Таблиця станцій
    file <<
" ┌──────────────────────┬──────┬─────────┬─────────┬──────┬──────────┐\n";
    file << " │  IP-адреса       │ Порт │ Сегмент │ Радіус │ Кут  │ Статус │\n";
    file <<
" ├──────────────────────┼──────┼─────────┼─────────┼──────┼──────────┤\n";

    for (const auto& station : stations) {
        file << " │ " << std::setw(19) << std::left << station.ipAddress
             << " │ " << std::setw(4) << station.port
             << " │ " << std::setw(7) << station.segmentId
             << " │ " << std::setw(6) << std::fixed << std::setprecision(2) << station.radius
             << " │ " << std::setw(5) << std::fixed << std::setprecision(1) << station.angle
             << " │ " << std::setw(7) << (station.isActive ? "Активна" : "Неактивна")
             << " │\n";
    }

    file <<
" └──────────────────────┴──────┴─────────┴─────────┴──────┴──────────┘\n\n";

    // Координатна площина
    file << "\nКООРДИНАТНА ПЛОЩИНА:\n";
    file << "(Розташування станцій по сегментах)\n\n";
    file << generateCoordinateVisualization(stations);

    // Детальна інформація по сегментах
    file << "\n\nДЕТАЛЬНА ІНФОРМАЦІЯ ПО СЕГМЕНТАХ:\n";
    file << "═══════════════════════════════════════════════════════\n";

    for (int seg = 0; seg < 4; seg++) {
        file << "\nСегмент " << seg << " (";
        switch(seg) {
            case 0: file << "I квадрант, 0°-90°"; break;
            case 1: file << "II квадрант, 90°-180°"; break;
            case 2: file << "III квадрант, 180°-270°"; break;
            case 3: file << "IV квадрант, 270°-360°"; break;
        }
        file << "):\n";
        file << "──────────────────────────────────────────────────\n";

        int count = 0;
        for (const auto& station : stations) {
            if (station.segmentId == seg) {
                count++;
```

```
            // Переведення в декартові координати
            double angleRad = station.angle * PI / 180.0;
            double x = station.radius * cos(angleRad);
            double y = station.radius * sin(angleRad);

            file << "  " << station.ipAddress
                << " - Координати: ("
                << std::fixed << std::setprecision(2) << x << ", " << y
                << "), Радіус: " << station.radius
                << ", Кут: " << station.angle << "°";

            if (station.isCenter) {
               file << " [ЦЕНТР СИСТЕМИ]";
            }
            file << "\n";
         }
      }

      if (count == 0) {
         file << "  (Немає станцій)\n";
      }
   }

   // Легенда
   file <<
"\n\n═══════════════════════════════════════════════════════════════════════════════\n";
   file << "ЛЕГЕНДА:\n";
   file << "  • Сегмент - квадрант координатної площини (0-3)\n";
   file << "  • Радіус - відстань від центру координат (1-3)\n";
   file << "  • Кут - полярний кут розміщення станції (0-360°)\n";
   file << "  • Станції розподілені рівномірно по квадрантах\n";
   file << "  • Центр системи може переміщуватись між станціями\n";
   file <<
"═══════════════════════════════════════════════════════════════════════════════\n";

   file.close();
   return true;
}

std::string ReportGenerator::generateCoordinateVisualization(const std::vector<Station>& stations) {
   std::stringstream ss;

   // Розмір ASCII-графіку
   const int width = 61;
   const int height = 31;
   const int centerX = width / 2;
   const int centerY = height / 2;

   // Ініціалізація поля
   char field[31][61];
   for (int y = 0; y < height; y++) {
      for (int x = 0; x < width; x++) {
         field[y][x] = ' ';
      }
   }

   // Рисування осей
   for (int x = 0; x < width; x++) {
      field[centerY][x] = '-';
   }
   for (int y = 0; y < height; y++) {
      field[y][centerX] = '|';
   }
```

```
    field[centerY][centerX] = '+';

    // Рисування кіл
    for (int r = 1; r <= 3; r++) {
        for (int angle = 0; angle < 360; angle += 5) {
            double rad = angle * PI / 180.0;
            int x = centerX + static_cast<int>(r * 8 * cos(rad));
            int y = centerY - static_cast<int>(r * 4 * sin(rad));

            if (x >= 0 && x < width && y >= 0 && y < height) {
                if (field[y][x] == ' ') {
                    field[y][x] = '.';
                }
            }
        }
    }

    // Розміщення станцій
    for (const auto& station : stations) {
        double angleRad = station.angle * PI / 180.0;
        int x = centerX + static_cast<int>(station.radius * 8 * cos(angleRad));
        int y = centerY - static_cast<int>(station.radius * 4 * sin(angleRad));

        if (x >= 0 && x < width && y >= 0 && y < height) {
            if (station.isCenter) {
                field[y][x] = 'C'; // Центр
            } else if (station.isActive) {
                field[y][x] = 'A'; // Активна
            } else {
                field[y][x] = 'O'; // Неактивна
            }
        }
    }

    // Виведення поля
    ss << "    ";
    for (int i = 0; i < width; i++) ss << "-";
    ss << "\n";

    for (int y = 0; y < height; y++) {
        ss << "    |";
        for (int x = 0; x < width; x++) {
            ss << field[y][x];
        }
        ss << "|\n";
    }

    ss << "    ";
    for (int i = 0; i < width; i++) ss << "-";
    ss << "\n\n";

    ss << "    Позначення: C - Центр системи, A - Активна станція, O - Неактивна станція\n";
    ss << "    Кола показують радіуси 1, 2 та 3\n";

    return ss.str();
}

        Файл: SystemCenter.cpp
        #define _CRT_SECURE_NO_WARNINGS
#define NOMINMAX
#include "DeceptionSystem.h"
#include <thread>
#include <atomic>
```

```
#include <random>
#include <queue>
#include <mutex>
#include <iostream>

// Клас центру системи - РЕАЛІЗАЦІЯ
class SystemCenter {
private:
    std::string myIpAddress;
    StationDatabase database;
    NetworkManager networkManager;
    std::atomic<bool> isRunning;
    std::atomic<bool> isCenter;

    std::thread integrityThread;
    std::thread commandThread;
    std::thread listenerThread;

    std::queue<Event> eventQueue;
    std::mutex eventMutex;

    std::random_device rd;
    std::mt19937 gen;

    // Виконання функцій системи
    bool executeFunction(SystemFunction func);
    std::string getFunctionName(SystemFunction func);

    // Потоки
    void integrityCheckLoop();
    void commandExecutionLoop();
    void messageListenerLoop();

    // Обробка повідомлень
    void handleMessage(const Message& msg, SOCKET clientSocket);
    void handleIntegrityCheck(const Message& msg);
    void handleCommandExecute(const Message& msg);
    void handleEventReport(const Message& msg);

public:
    SystemCenter(const std::string& myIp, int port = DEFAULT_PORT);
    ~SystemCenter();

    bool start();
    void stop();

    // Управління станціями
    bool registerStation(const std::string& ip, int port);
    bool unregisterStation(const std::string& ip);
    std::vector<Station> getStations();

    // Перевірка цілісності
    bool checkStationIntegrity(const std::string& ip);
    void checkAllStationsIntegrity();

    // Відправка команд
    bool sendRandomCommand();
    bool sendCommandToStation(const std::string& ip, SystemFunction func);
    bool broadcastCommand(SystemFunction func);

            // Переміщення центру
    bool transferCenterTo(const std::string& newCenterIp);
```

```
    // Робота з подіями
    void reportEvent(const Event& event);
    std::vector<Event> getRecentEvents(int count = 10);

    // Генерація звітів
    bool generateReport(const std::string& outputPath);

    // Статус
    bool isCenterNode() const { return isCenter.load(); }
    std::string getMyIp() const { return myIpAddress; }
};

// Реалізація SystemCenter
SystemCenter::SystemCenter(const std::string& myIp, int port)
    : database("stations.db"),
      networkManager(port),
      gen(rd()) {
    myIpAddress = myIp;
    isRunning = false;
    isCenter = false;
}

SystemCenter::~SystemCenter() {
    stop();
}

bool SystemCenter::start() {
    if (!networkManager.initialize()) {
        return false;
    }

    isRunning = true;
    isCenter = true;  // Автоматично стає центром при запуску

    // Реєструє себе як центральну станцію
    Station myStation;
    myStation.ipAddress = myIpAddress;
    myStation.port = DEFAULT_PORT;
    myStation.isActive = true;
    myStation.isCenter = true;
    myStation.lastSeen = std::chrono::system_clock::now();
    database.addStation(myIpAddress, myStation);

    std::cout << "System started as CENTER at " << myIpAddress << std::endl;

    // Запуск потоків
    listenerThread = std::thread(&SystemCenter::messageListenerLoop, this);
    integrityThread = std::thread(&SystemCenter::integrityCheckLoop, this);
    commandThread = std::thread(&SystemCenter::commandExecutionLoop, this);

    return true;
}

void SystemCenter::stop() {
    isRunning = false;

    if (listenerThread.joinable()) {
        listenerThread.join();
    }
    if (integrityThread.joinable()) {
        integrityThread.join();
    }
    if (commandThread.joinable()) {
```

```
        commandThread.join();
    }

    networkManager.cleanup();
}

bool SystemCenter::registerStation(const std::string& ip, int port) {
    Station station;
    station.ipAddress = ip;
    station.port = port;
    station.isActive = false;
    station.isCenter = false;
    station.lastSeen = std::chrono::system_clock::now();

    return database.addStation(ip, station);
}

bool SystemCenter::unregisterStation(const std::string& ip) {
    return database.removeStation(ip);
}

std::vector<Station> SystemCenter::getStations() {
    return database.getAllStations();
}

bool SystemCenter::checkStationIntegrity(const std::string& ip) {
    Station* station = database.getStation(ip);
    if (!station) {
        return false;
    }

    Message msg;
    msg.type = MessageType::INTEGRITY_CHECK;
    msg.senderId = myIpAddress;
    msg.receiverId = ip;
    msg.timestamp = std::chrono::system_clock::now();

    bool success = networkManager.sendMessage(ip, station->port, msg);

    if (success) {
        station->lastSeen = std::chrono::system_clock::now();
        station->isActive = true;
        database.updateStation(ip, *station);
    } else {
        station->isActive = false;
        database.updateStation(ip, *station);
    }

    return success;
}

void SystemCenter::checkAllStationsIntegrity() {
    std::vector<Station> stations = database.getAllStations();

    for (auto& station : stations) {
        checkStationIntegrity(station.ipAddress);
    }
}

bool SystemCenter::sendRandomCommand() {
    if (!isCenter) {
        std::cout << "ERROR: This system is not the center!" << std::endl;
        return false;
```

```
    }

    std::vector<Station> stations = database.getAllStations();
    if (stations.empty()) {
        std::cout << "ERROR: No stations registered!" << std::endl;
        return false;
    }

    // Обчислює активні станції (крім себе)
    int activeCount = 0;
    for (const auto& station : stations) {
        if (station.ipAddress != myIpAddress && station.isActive) {
            activeCount++;
        }
    }

    // Генерація випадкової функції (1-11)
    std::uniform_int_distribution<> dis(1, 11);
    SystemFunction randomFunc = static_cast<SystemFunction>(dis(gen));

    std::cout << "Generated command: " << getFunctionName(randomFunc) << std::endl;

    if (activeCount == 0) {
        std::cout << "WARNING: No other active stations. Executing locally..." << std::endl;
        // Виконує локально для тесту
        executeFunction(randomFunc);

        // Створюємо подію
        Event event;
        event.eventId = myIpAddress + "_local";
        event.description = "Executed " + getFunctionName(randomFunc) + " locally (no remote stations)";
        event.sourceIp = myIpAddress;
        event.timestamp = std::chrono::system_clock::now();
        event.severity = 3;
        reportEvent(event);

        return true;
    }

    return broadcastCommand(randomFunc);
}

bool SystemCenter::sendCommandToStation(const std::string& ip, SystemFunction func) {
    Station* station = database.getStation(ip);
    if (!station) {
        return false;
    }

    Message msg;
    msg.type = MessageType::COMMAND_EXECUTE;
    msg.senderId = myIpAddress;
    msg.receiverId = ip;
    msg.function = func;
    msg.payload = getFunctionName(func);
    msg.timestamp = std::chrono::system_clock::now();

    return networkManager.sendMessage(ip, station->port, msg);
}

bool SystemCenter::broadcastCommand(SystemFunction func) {
    std::vector<Station> stations = database.getAllStations();
    bool allSuccess = true;
```

```
    for (const auto& station : stations) {
        if (station.ipAddress != myIpAddress && station.isActive) {
            bool success = sendCommandToStation(station.ipAddress, func);
            allSuccess = allSuccess && success;
        }
    }

    return allSuccess;
}

bool SystemCenter::transferCenterTo(const std::string& newCenterIp) {
    Station* newCenter = database.getStation(newCenterIp);
    if (!newCenter) {
        return false;
    }

    // Повідомлення про передачу центру
    Message msg;
    msg.type = MessageType::CENTER_TRANSFER;
    msg.senderId = myIpAddress;
    msg.receiverId = newCenterIp;
    msg.payload = "CENTER_TRANSFER";
    msg.timestamp = std::chrono::system_clock::now();

    bool success = networkManager.sendMessage(newCenterIp, newCenter->port, msg);

    if (success) {
        // Оновлення статусів
        Station* oldCenter = database.getStation(myIpAddress);
        if (oldCenter) {
            oldCenter->isCenter = false;
            database.updateStation(myIpAddress, *oldCenter);
        }

        newCenter->isCenter = true;
        database.updateStation(newCenterIp, *newCenter);

        isCenter = false;
    }

    return success;
}

void SystemCenter::reportEvent(const Event& event) {
    std::lock_guard<std::mutex> lock(eventMutex);
    eventQueue.push(event);

    // Обмеження розміру черги
    while (eventQueue.size() > 1000) {
        eventQueue.pop();
    }
}

std::vector<Event> SystemCenter::getRecentEvents(int count) {
    std::lock_guard<std::mutex> lock(eventMutex);

    std::vector<Event> events;
    std::queue<Event> tempQueue = eventQueue;

    while (!tempQueue.empty() && events.size() < static_cast<size_t>(count)) {
        events.push_back(tempQueue.front());
        tempQueue.pop();
    }
```

```
    return events;
}

bool SystemCenter::generateReport(const std::string& outputPath) {
    std::vector<Station> stations = database.getAllStations();
    return ReportGenerator::generateAppendixB(stations, outputPath);
}

// Приватні методи
void SystemCenter::integrityCheckLoop() {
    while (isRunning) {
        if (isCenter) {
            checkAllStationsIntegrity();
        }
                std::this_thread::sleep_for(std::chrono::seconds(INTEGRITY_CHECK_INTERVAL));
    }
}

void SystemCenter::commandExecutionLoop() {
    while (isRunning) {
        if (isCenter) {
            // Випадкове виконання команд
            std::uniform_int_distribution<> dis(30, 120); // 30-120 секунд
            int sleepTime = dis(gen);

            std::this_thread::sleep_for(std::chrono::seconds(sleepTime));

            if (isRunning && isCenter) {
                sendRandomCommand();
            }
        } else {
            std::this_thread::sleep_for(std::chrono::seconds(10));
        }
    }
}

void SystemCenter::messageListenerLoop() {
    while (isRunning) {
        SOCKET clientSocket = networkManager.acceptConnection();

        if (clientSocket != INVALID_SOCKET) {
            Message msg = networkManager.receiveMessage(clientSocket);
            handleMessage(msg, clientSocket);
            closesocket(clientSocket);
        }
    }
}

void SystemCenter::handleMessage(const Message& msg, SOCKET clientSocket) {
    switch (msg.type) {
        case MessageType::INTEGRITY_CHECK:
            handleIntegrityCheck(msg);
            break;

        case MessageType::COMMAND_EXECUTE:
            handleCommandExecute(msg);
            break;

        case MessageType::EVENT_REPORT:
            handleEventReport(msg);
            break;
```

```
        case MessageType::CENTER_TRANSFER:
            isCenter = true;
            Station* myStation = database.getStation(myIpAddress);
            if (myStation) {
                myStation->isCenter = true;
                database.updateStation(myIpAddress, *myStation);
            }
            break;
    }
}

void SystemCenter::handleIntegrityCheck(const Message& msg) {
    // Відповідь на перевірку цілісності
    Message response;
    response.type = MessageType::RESPONSE_SUCCESS;
    response.senderId = myIpAddress;
    response.receiverId = msg.senderId;
    response.timestamp = std::chrono::system_clock::now();

    Station* sender = database.getStation(msg.senderId);
    if (sender) {
        networkManager.sendMessage(msg.senderId, sender->port, response);
    }
}

void SystemCenter::handleCommandExecute(const Message& msg) {
    // Виконання команди
    bool success = executeFunction(msg.function);

    // Надсилання відповіді
    Message response;
    response.type = success ? MessageType::RESPONSE_SUCCESS : MessageType::RESPONSE_FAILURE;
    response.senderId = myIpAddress;
    response.receiverId = msg.senderId;
    response.function = msg.function;
    response.payload = success ? "Success" : "Failed";
    response.timestamp = std::chrono::system_clock::now();

    Station* sender = database.getStation(msg.senderId);
    if (sender) {
        networkManager.sendMessage(msg.senderId, sender->port, response);
    }
}

void SystemCenter::handleEventReport(const Message& msg) {
    // Обробка події від станції
    Event event;
    event.eventId = msg.senderId + "_" + std::to_string(std::chrono::system_clock::to_time_t(msg.timestamp));
    event.description = msg.payload;
    event.sourceIp = msg.senderId;
    event.timestamp = msg.timestamp;
    event.severity = 5; // За замовчуванням

    reportEvent(event);
}

bool SystemCenter::executeFunction(SystemFunction func) {
    // Тут імітація виконання функцій
    std::cout << "Executing function: " << getFunctionName(func) << std::endl;

    // Реальна логіка залежить від функції
    std::this_thread::sleep_for(std::chrono::milliseconds(100));
```

```
    return true; // Успішне виконання
}

std::string SystemCenter::getFunctionName(SystemFunction func) {
    switch (func) {
        case SystemFunction::SCAN_NETWORK: return "Scan Network";
        case SystemFunction::LOG_ACTIVITY: return "Log Activity";
        case SystemFunction::CAPTURE_TRAFFIC: return "Capture Traffic";
        case SystemFunction::ANALYZE_PATTERNS: return "Analyze Patterns";
        case SystemFunction::GENERATE_DECOY: return "Generate Decoy";
        case SystemFunction::UPDATE_RULES: return "Update Rules";
        case SystemFunction::SEND_ALERT: return "Send Alert";
        case SystemFunction::ISOLATE_SEGMENT: return "Isolate Segment";
        case SystemFunction::MIRROR_TRAFFIC: return "Mirror Traffic";
        case SystemFunction::ROTATE_CREDENTIALS: return "Rotate Credentials";
        case SystemFunction::BACKUP_STATE: return "Backup State";
        default: return "Unknown Function";
    }
}

            Файл: AdminGUI.cpp
            #define _CRT_SECURE_NO_WARNINGS
#define NOMINMAX
#include "DeceptionSystem.h"
#include <windows.h>
#include <commctrl.h>
#include <string>
#include <vector>
#include <thread>
#include <atomic>
#include <random>
#include <queue>
#include <mutex>

#pragma comment(lib, "comctl32.lib")

// Прототипи з SystemCenter.cpp
class SystemCenter;

// ID елементів GUI
#define IDC_LISTVIEW_STATIONS 1001
#define IDC_BUTTON_ADD 1002
#define IDC_BUTTON_REMOVE 1003
#define IDC_BUTTON_REFRESH 1004
#define IDC_BUTTON_GENERATE_REPORT 1005
#define IDC_BUTTON_TRANSFER_CENTER 1006
#define IDC_BUTTON_SEND_COMMAND 1007
#define IDC_EDIT_IP 1008
#define IDC_EDIT_PORT 1009
#define IDC_STATIC_STATUS 1010
#define IDC_LISTVIEW_EVENTS 1011
#define IDC_BUTTON_START_SYSTEM 1012
#define IDC_BUTTON_STOP_SYSTEM 1013

// Forward declaration для SystemCenter
class SystemCenter {
private:
    std::string myIpAddress;
    StationDatabase database;
    NetworkManager networkManager;
    std::atomic<bool> isRunning;
    std::atomic<bool> isCenter;
```

```
    std::thread integrityThread;
    std::thread commandThread;
    std::thread listenerThread;

    std::queue<Event> eventQueue;
    std::mutex eventMutex;

    std::random_device rd;
    std::mt19937 gen;

    bool executeFunction(SystemFunction func);
    std::string getFunctionName(SystemFunction func);
    void integrityCheckLoop();
    void commandExecutionLoop();
    void messageListenerLoop();
    void handleMessage(const Message& msg, SOCKET clientSocket);
    void handleIntegrityCheck(const Message& msg);
    void handleCommandExecute(const Message& msg);
    void handleEventReport(const Message& msg);

public:
    SystemCenter(const std::string& myIp, int port = DEFAULT_PORT);
    ~SystemCenter();

    bool start();
    void stop();
    bool registerStation(const std::string& ip, int port);
    bool unregisterStation(const std::string& ip);
    std::vector<Station> getStations();
    bool checkStationIntegrity(const std::string& ip);
    void checkAllStationsIntegrity();
    bool sendRandomCommand();
    bool sendCommandToStation(const std::string& ip, SystemFunction func);
    bool broadcastCommand(SystemFunction func);
    bool transferCenterTo(const std::string& newCenterIp);
    void reportEvent(const Event& event);
    std::vector<Event> getRecentEvents(int count = 10);
    bool generateReport(const std::string& outputPath);
    bool isCenterNode() const { return isCenter.load(); }
    std::string getMyIp() const { return myIpAddress; }
};

// Клас GUI адміністратора
class AdminGUI {
private:
    HWND hwndMain;
    HWND hwndListViewStations;
    HWND hwndListViewEvents;
    HWND hwndEditIp;
    HWND hwndEditPort;
    HWND hwndStatus;

    SystemCenter* systemCenter;
    std::string myIpAddress;

    static AdminGUI* instance;

    static LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

    void OnButtonAddStation();
    void OnButtonRemoveStation();
    void OnButtonRefresh();
    void OnButtonGenerateReport();
```

```
    void OnButtonTransferCenter();
    void OnButtonSendCommand();
    void OnButtonStartSystem();
    void OnButtonStopSystem();

    void UpdateStationsList();
    void UpdateEventsList();
    void UpdateStatus(const std::string& status);

    void InitializeControls(HWND hwnd);
    void InitializeListView(HWND hwndLV, bool isStations);

public:
    AdminGUI(const std::string& myIp);
    ~AdminGUI();

    bool Create(HINSTANCE hInstance);
    void Run();
};

AdminGUI* AdminGUI::instance = nullptr;

AdminGUI::AdminGUI(const std::string& myIp)
    : hwndMain(nullptr),
      hwndListViewStations(nullptr),
      hwndListViewEvents(nullptr),
      hwndEditIp(nullptr),
      hwndEditPort(nullptr),
      hwndStatus(nullptr),
      myIpAddress(myIp),
      systemCenter(nullptr) {
    instance = this;
}

AdminGUI::~AdminGUI() {
    if (systemCenter) {
        systemCenter->stop();
        delete systemCenter;
    }
}

bool AdminGUI::Create(HINSTANCE hInstance) {
    WNDCLASSEX wc = {0};
    wc.cbSize = sizeof(WNDCLASSEX);
    wc.style = CS_HREDRAW | CS_VREDRAW;
    wc.lpfnWndProc = WindowProc;
    wc.hInstance = hInstance;
    wc.hCursor = LoadCursor(NULL, IDC_ARROW);
    wc.hbrBackground = (HBRUSH)(COLOR_WINDOW + 1);
    wc.lpszClassName = L"DeceptionSystemAdmin";

    if (!RegisterClassEx(&wc)) {
        return false;
    }

    hwndMain = CreateWindowEx(
        0,
        L"DeceptionSystemAdmin",
        L"Адміністрування обманної системи",
        WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, CW_USEDEFAULT,
        1000, 700,
        NULL, NULL, hInstance, NULL
```

```
    );

    if (!hwndMain) {
        return false;
    }

    InitializeControls(hwndMain);

    ShowWindow(hwndMain, SW_SHOW);
    UpdateWindow(hwndMain);

    return true;
}

void AdminGUI::InitializeControls(HWND hwnd) {
    HINSTANCE hInstance = GetModuleHandle(NULL);

    CreateWindow(L"STATIC", L"Станції системи:",
        WS_VISIBLE | WS_CHILD,
        10, 10, 200, 20,
        hwnd, NULL, hInstance, NULL);

    hwndListViewStations = CreateWindow(WC_LISTVIEW, L"",
        WS_VISIBLE | WS_CHILD | WS_BORDER | LVS_REPORT | LVS_SINGLESEL,
        10, 35, 750, 250,
        hwnd, (HMENU)IDC_LISTVIEW_STATIONS, hInstance, NULL);

    InitializeListView(hwndListViewStations, true);

    CreateWindow(L"STATIC", L"IP-адреса:",
        WS_VISIBLE | WS_CHILD,
        10, 300, 80, 20,
        hwnd, NULL, hInstance, NULL);

    hwndEditIp = CreateWindow(L"EDIT", L"",
        WS_VISIBLE | WS_CHILD | WS_BORDER | ES_AUTOHSCROLL,
        100, 297, 150, 25,
        hwnd, (HMENU)IDC_EDIT_IP, hInstance, NULL);

    CreateWindow(L"STATIC", L"Порт:",
        WS_VISIBLE | WS_CHILD,
        260, 300, 50, 20,
        hwnd, NULL, hInstance, NULL);

    hwndEditPort = CreateWindow(L"EDIT", L"8888",
        WS_VISIBLE | WS_CHILD | WS_BORDER | ES_AUTOHSCROLL | ES_NUMBER,
        310, 297, 80, 25,
        hwnd, (HMENU)IDC_EDIT_PORT, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Додати станцію",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        400, 295, 120, 30,
        hwnd, (HMENU)IDC_BUTTON_ADD, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Видалити",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        530, 295, 100, 30,
        hwnd, (HMENU)IDC_BUTTON_REMOVE, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Оновити",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        640, 295, 100, 30,
        hwnd, (HMENU)IDC_BUTTON_REFRESH, hInstance, NULL);
```

```
    CreateWindow(L"BUTTON", L"Запустити систему",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        770, 35, 200, 35,
        hwnd, (HMENU)IDC_BUTTON_START_SYSTEM, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Зупинити систему",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        770, 80, 200, 35,
        hwnd, (HMENU)IDC_BUTTON_STOP_SYSTEM, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Згенерувати звіт",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        770, 130, 200, 35,
        hwnd, (HMENU)IDC_BUTTON_GENERATE_REPORT, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Передати центр",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        770, 175, 200, 35,
        hwnd, (HMENU)IDC_BUTTON_TRANSFER_CENTER, hInstance, NULL);

    CreateWindow(L"BUTTON", L"Вибір команди згідно популяційного алгоритму",
        WS_VISIBLE | WS_CHILD | BS_PUSHBUTTON,
        770, 220, 200, 35,
        hwnd, (HMENU)IDC_BUTTON_SEND_COMMAND, hInstance, NULL);

    CreateWindow(L"STATIC", L"Події системи:",
        WS_VISIBLE | WS_CHILD,
        10, 340, 200, 20,
        hwnd, NULL, hInstance, NULL);

    hwndListViewEvents = CreateWindow(WC_LISTVIEW, L"",
        WS_VISIBLE | WS_CHILD | WS_BORDER | LVS_REPORT,
        10, 365, 960, 200,
        hwnd, (HMENU)IDC_LISTVIEW_EVENTS, hInstance, NULL);

    InitializeListView(hwndListViewEvents, false);

    hwndStatus = CreateWindow(L"STATIC", L"Статус: Готово до роботи",
        WS_VISIBLE | WS_CHILD | SS_LEFT,
        10, 575, 960, 20,
        hwnd, (HMENU)IDC_STATIC_STATUS, hInstance, NULL);
}

void AdminGUI::InitializeListView(HWND hwndLV, bool isStations) {
    ListView_SetExtendedListViewStyle(hwndLV, LVS_EX_FULLROWSELECT | LVS_EX_GRIDLINES);

    LVCOLUMN lvc = {0};
    lvc.mask = LVCF_TEXT | LVCF_WIDTH | LVCF_SUBITEM;

    if (isStations) {
        lvc.pszText = (LPWSTR)L"IP-адреса";
        lvc.cx = 150;
        ListView_InsertColumn(hwndLV, 0, &lvc);

        lvc.pszText = (LPWSTR)L"Порт";
        lvc.cx = 80;
        ListView_InsertColumn(hwndLV, 1, &lvc);

        lvc.pszText = (LPWSTR)L"Статус";
        lvc.cx = 100;
        ListView_InsertColumn(hwndLV, 2, &lvc);
```

```
        lvc.pszText = (LPWSTR)L"Центр";
        lvc.cx = 80;
        ListView_InsertColumn(hwndLV, 3, &lvc);

        lvc.pszText = (LPWSTR)L"Сегмент";
        lvc.cx = 80;
        ListView_InsertColumn(hwndLV, 4, &lvc);

        lvc.pszText = (LPWSTR)L"Радіус";
        lvc.cx = 80;
        ListView_InsertColumn(hwndLV, 5, &lvc);

        lvc.pszText = (LPWSTR)L"Кут";
        lvc.cx = 80;
        ListView_InsertColumn(hwndLV, 6, &lvc);
    } else {
        lvc.pszText = (LPWSTR)L"Час";
        lvc.cx = 150;
        ListView_InsertColumn(hwndLV, 0, &lvc);

        lvc.pszText = (LPWSTR)L"Джерело";
        lvc.cx = 130;
        ListView_InsertColumn(hwndLV, 1, &lvc);

        lvc.pszText = (LPWSTR)L"Опис події";
        lvc.cx = 500;
        ListView_InsertColumn(hwndLV, 2, &lvc);

        lvc.pszText = (LPWSTR)L"Серйозність";
        lvc.cx = 100;
        ListView_InsertColumn(hwndLV, 3, &lvc);
    }
}

void AdminGUI::UpdateStationsList() {
    if (!systemCenter) return;

    ListView_DeleteAllItems(hwndListViewStations);

    std::vector<Station> stations = systemCenter->getStations();

    LVITEM lvi = {0};
    lvi.mask = LVIF_TEXT;

    for (size_t i = 0; i < stations.size(); i++) {
        const Station& station = stations[i];

        lvi.iItem = static_cast<int>(i);
        lvi.iSubItem = 0;
        std::wstring wip(station.ipAddress.begin(), station.ipAddress.end());
        lvi.pszText = (LPWSTR)wip.c_str();
        ListView_InsertItem(hwndListViewStations, &lvi);

        std::wstring wport = std::to_wstring(station.port);
        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 1, (LPWSTR)wport.c_str());

        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 2,
            station.isActive ? (LPWSTR)L"Активна" : (LPWSTR)L"Неактивна");

        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 3,
            station.isCenter ? (LPWSTR)L"Так" : (LPWSTR)L"Ні");

        std::wstring wseg = std::to_wstring(station.segmentId);
```

```
        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 4, (LPWSTR)wseg.c_str());

        std::wstring wrad = std::to_wstring(static_cast<int>(station.radius));
        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 5, (LPWSTR)wrad.c_str());

        std::wstring wang = std::to_wstring(static_cast<int>(station.angle));
        ListView_SetItemText(hwndListViewStations, static_cast<int>(i), 6, (LPWSTR)wang.c_str());
    }
}

void AdminGUI::UpdateEventsList() {
    if (!systemCenter) return;

    ListView_DeleteAllItems(hwndListViewEvents);

    std::vector<Event> events = systemCenter->getRecentEvents(50);

    LVITEM lvi = {0};
    lvi.mask = LVIF_TEXT;

    for (size_t i = 0; i < events.size(); i++) {
        const Event& event = events[i];

        lvi.iItem = static_cast<int>(i);
        lvi.iSubItem = 0;

        time_t tt = std::chrono::system_clock::to_time_t(event.timestamp);
        char timeStr[100];
        struct tm timeinfo;
        localtime_s(&timeinfo, &tt);
        strftime(timeStr, sizeof(timeStr), "%Y-%m-%d %H:%M:%S", &timeinfo);
        std::wstring wtime(timeStr, timeStr + strlen(timeStr));
        lvi.pszText = (LPWSTR)wtime.c_str();
        ListView_InsertItem(hwndListViewEvents, &lvi);

        std::wstring wsource(event.sourceIp.begin(), event.sourceIp.end());
        ListView_SetItemText(hwndListViewEvents, static_cast<int>(i), 1, (LPWSTR)wsource.c_str());

        std::wstring wdesc(event.description.begin(), event.description.end());
        ListView_SetItemText(hwndListViewEvents, static_cast<int>(i), 2, (LPWSTR)wdesc.c_str());

        std::wstring wsev = std::to_wstring(event.severity);
        ListView_SetItemText(hwndListViewEvents, static_cast<int>(i), 3, (LPWSTR)wsev.c_str());
    }
}

void AdminGUI::UpdateStatus(const std::string& status) {
    std::wstring wstatus = std::wstring(status.begin(), status.end());
    SetWindowText(hwndStatus, wstatus.c_str());
}

void AdminGUI::OnButtonAddStation() {
    wchar_t ip[256];
    wchar_t port[16];

    GetWindowText(hwndEditIp, ip, 256);
    GetWindowText(hwndEditPort, port, 16);

    std::wstring wip(ip);
    std::string sip(wip.begin(), wip.end());

    int nPort = _wtoi(port);
```

```
    if (sip.empty() || nPort == 0) {
        MessageBox(hwndMain, L"Введіть коректну IP-адресу та порт", L"Помилка", MB_OK | MB_ICONERROR);
        return;
    }

    if (systemCenter && systemCenter->registerStation(sip, nPort)) {
        UpdateStationsList();
        UpdateStatus("Станцію додано: " + sip);
        SetWindowText(hwndEditIp, L"");
    } else {
        MessageBox(hwndMain, L"Не вдалося додати станцію", L"Помилка", MB_OK | MB_ICONERROR);
    }
}

void AdminGUI::OnButtonRemoveStation() {
    int selectedIndex = ListView_GetNextItem(hwndListViewStations, -1, LVNI_SELECTED);

    if (selectedIndex == -1) {
        MessageBox(hwndMain, L"Виберіть станцію для видалення", L"Помилка", MB_OK | MB_ICONWARNING);
        return;
    }

    wchar_t ip[256];
    ListView_GetItemText(hwndListViewStations, selectedIndex, 0, ip, 256);

    std::wstring wip(ip);
    std::string sip(wip.begin(), wip.end());

    if (systemCenter && systemCenter->unregisterStation(sip)) {
        UpdateStationsList();
        UpdateStatus("Станцію видалено: " + sip);
    }
}

void AdminGUI::OnButtonRefresh() {
    UpdateStationsList();
    UpdateEventsList();
    UpdateStatus("Дані оновлено");
}

void AdminGUI::OnButtonGenerateReport() {
    if (!systemCenter) return;

    wchar_t filePath[MAX_PATH];
    wcscpy_s(filePath, L"DeceptionSystem_AppendixB.txt");

    OPENFILENAME ofn = {0};
    ofn.lStructSize = sizeof(ofn);
    ofn.hwndOwner = hwndMain;
    ofn.lpstrFile = filePath;
    ofn.nMaxFile = MAX_PATH;
    ofn.lpstrFilter = L"Text Files (*.txt)\0*.txt\0All Files (*.*)\0*.*\0";
    ofn.nFilterIndex = 1;
    ofn.Flags = OFN_PATHMUSTEXIST | OFN_OVERWRITEPROMPT;

    if (GetSaveFileName(&ofn)) {
        std::wstring wpath(filePath);
        std::string spath(wpath.begin(), wpath.end());

        if (systemCenter->generateReport(spath)) {
            MessageBox(hwndMain, L"Звіт успішно згенеровано", L"Успіх", MB_OK | MB_ICONINFORMATION);
            UpdateStatus("Звіт згенеровано: " + spath);
        } else {
```

```
            MessageBox(hwndMain, L"Не вдалося згенерувати звіт", L"Помилка", MB_OK | MB_ICONERROR);
        }
    }
}

void AdminGUI::OnButtonTransferCenter() {
    int selectedIndex = ListView_GetNextItem(hwndListViewStations, -1, LVNI_SELECTED);

    if (selectedIndex == -1) {
        MessageBox(hwndMain, L"Виберіть станцію для передачі центру", L"Помилка", MB_OK | MB_ICONWARNING);
        return;
    }

    wchar_t ip[256];
    ListView_GetItemText(hwndListViewStations, selectedIndex, 0, ip, 256);

    std::wstring wip(ip);
    std::string sip(wip.begin(), wip.end());

    if (systemCenter && systemCenter->transferCenterTo(sip)) {
        UpdateStationsList();
        UpdateStatus("Центр передано на станцію: " + sip);
        MessageBox(hwndMain, L"Центр системи успішно передано", L"Успіх", MB_OK | MB_ICONINFORMATION);
    } else {
        MessageBox(hwndMain, L"Не вдалося передати центр", L"Помилка", MB_OK | MB_ICONERROR);
    }
}

void AdminGUI::OnButtonSendCommand() {
    if (!systemCenter) return;

    if (systemCenter->sendRandomCommand()) {
        UpdateStatus("Випадкову команду надіслано всім станціям");
    } else {
        MessageBox(hwndMain, L"Не вдалося надіслати команду", L"Помилка", MB_OK | MB_ICONERROR);
    }
}

void AdminGUI::OnButtonStartSystem() {
    if (systemCenter) {
        MessageBox(hwndMain, L"Система вже запущена", L"Інформація", MB_OK | MB_ICONINFORMATION);
        return;
    }

    systemCenter = new SystemCenter(myIpAddress, DEFAULT_PORT);

    if (systemCenter->start()) {
        UpdateStatus("Систему запущено");
        UpdateStationsList();
        MessageBox(hwndMain, L"Систему успішно запущено", L"Успіх", MB_OK | MB_ICONINFORMATION);
    } else {
        delete systemCenter;
        systemCenter = nullptr;
        MessageBox(hwndMain, L"Не вдалося запустити систему", L"Помилка", MB_OK | MB_ICONERROR);
    }
}

void AdminGUI::OnButtonStopSystem() {
    if (!systemCenter) {
        MessageBox(hwndMain, L"Система не запущена", L"Інформація", MB_OK | MB_ICONINFORMATION);
        return;
    }
```

```
    systemCenter->stop();
    delete systemCenter;
    systemCenter = nullptr;

    UpdateStatus("Систему зупинено");
    MessageBox(hwndMain, L"Систему зупинено", L"Інформація", MB_OK | MB_ICONINFORMATION);
}

LRESULT CALLBACK AdminGUI::WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {
    switch (uMsg) {
        case WM_COMMAND:
            switch (LOWORD(wParam)) {
                case IDC_BUTTON_ADD:
                    instance->OnButtonAddStation();
                    break;
                case IDC_BUTTON_REMOVE:
                    instance->OnButtonRemoveStation();
                    break;
                case IDC_BUTTON_REFRESH:
                    instance->OnButtonRefresh();
                    break;
                case IDC_BUTTON_GENERATE_REPORT:
                    instance->OnButtonGenerateReport();
                    break;
                case IDC_BUTTON_TRANSFER_CENTER:
                    instance->OnButtonTransferCenter();
                    break;
                case IDC_BUTTON_SEND_COMMAND:
                    instance->OnButtonSendCommand();
                    break;
                case IDC_BUTTON_START_SYSTEM:
                    instance->OnButtonStartSystem();
                    break;
                case IDC_BUTTON_STOP_SYSTEM:
                    instance->OnButtonStopSystem();
                    break;
            }
            break;

        case WM_DESTROY:
            PostQuitMessage(0);
            return 0;
    }
        return DefWindowProc(hwnd, uMsg, wParam, lParam);
}

void AdminGUI::Run() {
    MSG msg;
    while (GetMessage(&msg, NULL, 0, 0)) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
}
```