МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ДЕРЖАВНИЙ УНІВЕРСИТЕТ ТЕЛЕКОМУНІКАЦІЙ

Кваліфікаційна наукова
праця на правах рукопису

МАРЧЕНКО ВІТАЛІЙ ВІКТОРОВИЧ

УДК 004.056

ДИСЕРТАЦІЯ

МЕТОД ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОЦЕСІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ ПІДПРИЄМСТВА НА ОСНОВІ ІДЕНТИФІКАЦІЇ ТА ДІАГНОСТУВАННЯ СТАНІВ ЛОГІЧНИХ ОБ'ЄКТІВ

Спеціальність 125 «Кібербезпека»

Подається на здобуття наукового ступеня доктора філософії

Дисертація містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.
_____ В.В. Марченко

Науковий керівник:
ГАЙДУР Галина Іванівна
доктор технічних наук,
професор

КИЇВ — 2023

АНОТАЦІЯ

Марченко В.В. Метод виявлення шкідливих процесів інформаційної системи підприємства на основі ідентифікації та діагностування станів логічних об’єктів. — Кваліфікаційна наукова праця на правах рукопису.

Дисертація на здобуття наукового ступеня доктора філософії за спеціальністю 125 «Кібербезпека». — Державний університет телекомунікацій, МОН України, Київ, 2023.

Дисертаційна робота присвячена вирішенню актуального наукового завдання, сутність якого полягає в розробці методу виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностуванні станів логічних об’єктів.

Сучасні інформаційні технології є важливим елементом функціонування підприємств, але одночасно вони також відкривають шлях до можливих шкідливих впливів на інформаційну систему підприємства. Шкідливі процеси можуть призвести до порушення конфіденційності, цілісності та доступності даних. Для захисту від таких процесів необхідно розробити ефективні методи виявлення та діагностики станів логічних об'єктів, що забезпечують роботу інформаційної системи підприємства.

Однак, вирішення питання виявлення шкідливих процесів в інформаційній системі підприємства, що дозволяє зменшити недоліки існуючих методів, зокрема, обробка великих даних в режимі реального часу та швидкість реагування на відхилення функціонування логічних об’єктів від нормального функціонування без помилок, залишається не до кінця вивченим. Тобто, існує протиріччя між необхідністю оперативного ідентифікування (в режимі реального часу) шкідливого процесу, якісного виявлення та зменшення показника хибно-позитивних спрацювань.

У зв’язку з цим, існує необхідність вирішення актуального наукового завдання, сутність якого полягає в розробці методу виявлення шкідливих процесів на основі ідентифікації та діагностування станів логічних об’єктів.

Для досягнення мети підвищення ефективності роботи інформаційної системи підприємства за рахунок зменшення часу на пошук та вирішення проблем, що виникають у процесі її роботи шляхом вчасного виявлення можливих проблем та шкідливих процесів, що можуть впливати на її функціонування було отримано наукові результати:

1) вперше розроблено модель ідентифікації та діагностування станів логічних об’єктів, наукова новизна якого полягає в тому, що вона ґрунтується на теорії скінченних автоматів на основі протоколу TCP, що дозволяє на стадії створення з’єднання ідентифікувати порушення пакету зі встановленими параметрами та діагностувати роботу транспортного протоколу;

2) удосконалено метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню, щодо виявлення шкідливих процесів, в якому на відміну від існуючих, на основі методу основних компонентів обґрунтовано вибір параметрів вхідних даних логічного об’єкту моделі розширеного скінченного автомату TCP протоколу. Такий підхід дозволяє прискорити обробку даних при виявленні шкідливих процесів в інформаційній системі організації в реальному часі;

3) удосконалено метод виявлення шкідливих процесів в інформаційній системі організації, який на відміну від існуючих, базується на алгоритмах машинного навчання та враховує обрані параметри які підлягають ідентифікації та діагностуванню. Такий метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі.

У вступі обґрунтовується важливість й актуальність теми дисертаційного дослідження, сформульовано мету та задачі роботи, визначено основні положення, наукову та практичну цінність отриманих результатів дослідження та введено особистий внесок автора.

У першому розділі здійснено аналіз поточного стану та перспектив застосування методу виявлення шкідливих процесів в інформаційній системі підприємства, зокрема проблеми визначення в реальному часі та необхідних ресурсів на опрацювання великих даних.

У другому розділі розроблено розширену модель функціонування протоколу TCP на основі скінченних автоматів, що дозволяє наочно переглянути стани логічних об’єктів за допомогою матриць переходів станів, як в стані спокою так і допустимих переходів в розширеній матриці переходів станів TCP протоколу.

У третьому розділі удосконалено вибір критеріїв параметрів логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів та вибору оптимальних параметрів для машинного навчання, який базується на методі головних компонент на основі розширеного скінченного автомату TCP протоколу. Удосконалено метод машинного навчання на базі методу опорних векторів, з метою покращення ефективності виявлення шкідливих процесів в інформаційній системі. Досліджено ефективність методу виявлення шкідливих процесів в інформаційній системі підприємства. В результаті розроблені рекомендації щодо впровадження методу виявлення шкідливих процесів на основі ідентифікації та діагностуванні станів логічних об’єктів.

Узагальнюючим результатом проведених досліджень є метод виявлення шкідливих процесів на основі ідентифікації та діагностування станів логічних об’єктів. Даний метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі на 65-99% від існуючих сучасних методів та зменшення кількості хибно-позитивних спрацювань на 13-14%.

Дисертація виконувалась в Державному університеті телекомунікацій.

Результати наукових досліджень були впроваджені в навчальний процес на кафедрі інформаційної та кібернетичної безпеки Навчально-наукового інституту захисту інформації в дисципліни: «Захист від шкідливого

програмного забезпечення», «Технології виявлення уразливостей та протидії злоякісному програмному забезпеченню», «Цифрова криміналістика», «Технології виявлення уразливостей мережевих ресурсів», «Організація проведення наукових досліджень» та на кафедрі штучного інтелекту Навчально-наукового інституту інформаційних технологій в дисципліну «Штучний інтелект», при підготовці здобувачів освіти за спеціальністю 125 Кібербезпека.

Результати наукових досліджень були використані на кафедрі інформаційної та кібернетичної безпеки Навчально-наукового інституту захисту інформації під час виконання науково-дослідної роботи на тему «Розробка методів та засобів підвищення живучості інформаційно-комунікаційних систем в умовах кібернетичних атак» (№ 0114U000391 м. Київ) та «Методологія виявлення шкідливих процесів в інформаційних системах (№ 0121U113613, м. Київ), ДУТ.

Також результати наукових досліджень прийняті до впровадження в діяльність ТОВ «Смартс» та в ТОВ «ЄВРОТЕЛЕКОМ».

Ключові слова: атака, загрози безпеці, метод**,** логічний об’єкт, інформаційна система, ідентифікація, діагностування, шкідливий процес, скінченний автомат, машинне навчання, кореляція, метод опорних векторів, вибір головних компонент.

ANOTATION

Marchenko V.V. The method of detecting harmful processes of the information system of the enterprise based on the identification and diagnosis of the states of logical objects. — Qualifying scientific work on manuscript rights.

Dissertation for obtaining the scientific degree of Doctor of Philosophy in specialty 125 "Cybersecurity". — State University of Telecommunications, MES of Ukraine, Kyiv, 2023.

The dissertation work is devoted to the solution of an actual scientific task, the essence of which is the development of a method for detecting harmful processes in the information system of the enterprise based on the identification and diagnosis of the states of logical objects.

Modern information technologies are an important element of the functioning of enterprises, but at the same time they also open the way to possible harmful effects on the information system of the enterprise. Malicious processes can compromise the confidentiality, integrity, and availability of data. To protect against such processes, it is necessary to develop effective methods for detecting and diagnosing the states of logical objects that ensure the operation of the enterprise's information system.

However, the solution to the issue of detecting malicious processes in the information system of the enterprise, which allows to reduce the shortcomings of existing methods, in particular, the processing of large data in real time and the speed of response to the deviation of the functioning of logical objects from the normal functioning without errors, remains not fully studied. That is, there is a contradiction between the need for prompt identification (in real time) of a malicious process, qualitative detection and reduction of the rate of false positives.

In this regard, there is a need to solve an actual scientific task, the essence of which is to develop a method for detecting harmful processes based on the identification and diagnosis of the states of logical objects.

In order to achieve the goal of increasing the efficiency of the company's information system by reducing the time it takes to search for and solve problems that arise during its operation by timely identifying possible problems and harmful processes that can affect its functioning, the following scientific results were obtained:

1) a model for identifying and diagnosing the states of logical objects was developed for the first time, the scientific novelty of which is that it is based on the theory of finite automata based on the TCP protocol, which allows at the stage of establishing a connection to identify violations of a packet with set parameters and diagnose the operation transport protocol;

2) the method of selecting the criteria of logical objects to be identified and diagnosed for the detection of malicious processes has been improved, in which, unlike the existing methods, the choice of input data parameters of the logical object of the model of the extended finite state machine of the TCP protocol is substantiated based on the method of principal components. This approach allows you to speed up data processing when malicious processes are detected in the organization's information system in real time;

3) the method of detecting malicious processes in the organization's information system has been improved, which, unlike the existing ones, is based on machine learning algorithms and takes into account selected parameters that are subject to identification and diagnosis. This method makes it possible to increase the efficiency of identification and diagnosis of the states of logical objects in the organization's information system in real time.

The introduction substantiates the importance and relevance of the topic of the dissertation research, formulates the purpose and tasks of the work, defines the main provisions, scientific and practical value of the obtained research results, and introduces the author's personal contribution.

In the first section, an analysis of the current state and prospects for the application of the method of detecting malicious processes in the information system of the enterprise is carried out, in particular, the problems of real-time determination and the necessary resources for processing big data.

In the second section, an extended model of the functioning of the TCP protocol based on finite automata is developed, which allows you to visually review the states of logical objects using state transition matrices, both in the state of rest and admissible transitions in the extended state transition matrix of the TCP protocol.

In the third section, the selection of criteria for the parameters of logical objects, which are subject to identification and diagnosis for the detection of malicious processes and the selection of optimal parameters for machine learning, which is based on the method of principal components based on the extended finite state machine of the TCP protocol, is improved. The method of machine learning based on the method of support vectors has been improved in order to improve the effectiveness of detecting malicious processes in the information system. The effectiveness of the method of detecting harmful processes in the information system of the enterprise was investigated. As a result, recommendations were developed for the implementation of the method of detecting malicious processes based on the identification and diagnosis of the states of logical objects.

The general result of the conducted research is a method of detecting malicious processes based on the identification and diagnosis of states of logical objects. This method makes it possible to increase the efficiency of identification and diagnosis of the states of logical objects in the organization's information system in real time by 65-99% compared to existing modern methods and to reduce the number of false positives by 13-14%.

The dissertation was performed at the State University of Telecommunications.

The results of scientific research were implemented in the educational process at the department of information and cyber security of the Educational and Scientific

Institute of Information Protection in the following disciplines: "Protection against malicious software", "Technologies for detecting vulnerabilities and countering malicious software", "Digital forensics", "Detection technologies vulnerabilities of network resources", "Organization of conducting scientific research" and at the department of artificial intelligence of the Educational and Scientific Institute of Information Technologies in the discipline "Artificial Intelligence", in the preparation of students in the specialty 125 Cybersecurity.

The results of scientific research were used at the department of information and cyber security of the Educational and Scientific Institute of Information Protection during the implementation of research work on the topic "Development of methods and means of increasing the survivability of information and communication systems in conditions of cyber attacks" (No. 0114U000391, Kyiv) and "Methodology of detection of harmful processes in information systems (No. 0121U113613, Kyiv), SUT.

Also, the results of scientific research were accepted for implementation in the activities of Smarts LLC and EUROTELECOM LLC.

Keywords: attack, security threats, method, logical object, information system, identification, diagnosis, harmful process, finite automaton, machine learning, correlation, support vector method, selection of principal components.

СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Savchenko, V., Haidur, H., Dzyuba, T., Marchenko, V., Matsko, O., & Havryliuk, I. (2021). Providing of data center information security on the basis of Performance Balance. *2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT)*. https://doi.org/10.1109/atit54053.2021.9678679.

2. Гайдур, Г. І., Гахов, С. О., & Марченко, В. В. (2022). Метод побудови динамічної моделі логічного об’єкта інформаційної системи та визначення закону його функціонування. *RADIOELECTRONIC AND COMPUTER SYSTEMS*, (1), 129–140. https://doi.org/10.32620/reks.2022.1.10.

3. Лаптєв, С., Лаптєва, Т., Гребенніков, А., Кітура, О., Марченко, В., & Луценко, М. (2021). Удосконалення моделі протидії витоку інсайдерської інформації з урахуванням нестабільності функціонування окремих елементів системи захисту інформації. *Journal of Science. Lyon*, (20/2021). ISSN 3475-3281.

4. Dorokhin, O. O., Marchenko, V. V., & Semenova, I. D. (2021). Threat Intelligence Technology and methods of its use to protect the company from Cyber Threats. *Modern Information Security*, *3*(47). https://doi.org/10.31673/2409-7292.2021.031216.

5. Киричок, Р., Зінченко, О., Срібна, І., Марченко, В., & Кітура, О. (2021). Удосконалений метод автоматичного активного аналізу захищеності корпоративної мережі. *Захист Інформації*, *2*(2021). https://doi.org/doi.org/10.18372/2410-7840.23.15725.

6. Веселков, Н., & Марченко, В. (2021). Можливості та перспективи використання Open-Source технологій виявлення вторгнень в сферах малого та середнього бізнесу України. *Сучасний Захист Інформації*, *45*(1). https://doi.org/10.31673/2409-7292.2021.011925.

7. Гапоненко, О., Марченко, В., & Гайдур, Г. (2020). Переваги та недоліки HoneyPot — приманки для хакерів. *Сучасний Захист Інформації*, *1*(42). https://doi.org/10.31673/2409-7292.2020.025968.

8. Марченко, В., & Крючкова, Л. (2019). Радіомоніторинг частотних каналів мережі Wi-Fi. *Сучасний Захист Інформації*, *2*(38). https://doi.org/10.31673/2409-7292.2019.023236.

9. Савченко, В., Мацько, О., Легомінова, С., Полторак, І., & Марченко, В. (2019). Моделювання кібератак засобами теорії графів. *Сучасний Захист Інформації*, *4*(40). https://doi.org/10.31673/2409-7292.2019.040611.

10. Марченко, В. (2019). *What is security intelligence and why does it matter today?* (thesis). IX Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Київ.

11. Марченко, В., & Шевчук, Я. (2017). *Захист інформації на підприємстві* (thesis). V Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Киїів.

12. Юрченко, В., & Марченко, В. (2018). *Методика організації мережевої розвідки на основі Sql-ін'єкції* (thesis). V міжнародна науково-практична конференція «актуальні проблеми забезпечення Інформаційної та кібернетичної безпеки», Київ.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 14
ВСТУП 15
РОЗДІЛ 1 АНАЛІЗ СТАНУ ТА ПОСТАНОВКА ЗАВДАННЯ ЩОДО РОЗРОБКИ МЕТОДУ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОЦЕСІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ ПІДПРИЄМСТВА НА ОСНОВІ ІДЕНТИФІКАЦІЇ ТА ДІАГНОСТУВАННЯ СТАНІВ ЛОГІЧНИХ ОБ’ЄКТІВ 21
1.1. Аналіз механізмів захисту інформаційної системи підприємства 21
1.2. Підходи машинного навчання щодо виявлення шкідливих процесів в інформаційній системі організації 25
1.3. Дослідження науково-методичних підходів виявлення аномалії 37
1.4. Аналіз протиріч та постановка наукового завдання 39
Висновки до розділу 1 41
РОЗДІЛ 2 РОЗРОБКА РОЗШИРЕНОЇ МОДЕЛІ ІДЕНТИФІКАЦІЇ ТА ДІАГНОСТУВАННЯ СТАНІВ ЛОГІЧНОГО ОБ’ЄКТУ НА БАЗІ РОБОТИ TCP ПРОТОКОЛУ 43
2.1. Виявлення шкідливих процесів інформаційної системи на основі скінченних автоматів TCP протоколу 43
2.2. Процес розробки специфікації роботи протоколів щодо ідентифікації та діагностування станів логічних об’єктів 52
2.3. Розробка моделі розширеного автомата функціонування транспортного протоколу TCP для ідентифікації та діагностування станів логічного об’єкта інформаційної системи 58
2.4. Розробка моделі процесу функціонування ідентифікації та діагностування стану логічних об’єктів 91
Висновки до розділу 2 107

РОЗДІЛ 3 РОЗРОБКА МЕТОДУ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОЦЕСІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ ОРГАНІЗАЦІЇ ТА РЕКОМЕНДАЦІЙ ЩОДО ЙОГО ЗАСТОСУВАННЯ 109

3.1. Метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі алгоритму аналізу основних компонентів 109

3.2. Метод виявлення шкідливих процесів в інформаційній системі організації на основі алгоритмів машинного навчання 116

3.3. Дослідження ефективності та достовірності застосування методу виявлення шкідливих процесів в інформаційній системі підприємства 127

3.4. Розробка рекомендацій щодо впровадження методу виявлення шкідливих процесів в інформаційній системі підприємства 131

Висновки до розділу 3 136

ВИСНОВКИ 138

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 141

ДОДАТОК А 155

ДОДАТОК Б 159

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ШП — шкідливий процес

ІС — інформаційна система

СА — скінченний автомат

PCA — метод головних компонент

SVM — метод опорних векторів

TCP — транспортний протокол

ІСП — інформаційна система підприємства

ЛО — логічний об’єкт

DDoS — відмова в обслуговуванні

МН — машинне навчання

МВШП — метод виявлення шкідливого програмного забезпечення

МІД — модель ідентифікації та діагностування

ПЛО — пасивний логічний об’єкт

АЛО — активний логічний об’єкт

ВСТУП

Актуальність теми

Сучасні інформаційні технології є важливим елементом функціонування підприємств, але одночасно вони також відкривають шлях до можливих шкідливих впливів на інформаційну систему підприємства. Шкідливі процеси можуть призвести до порушення конфіденційності, цілісності та доступності даних. Для захисту від таких процесів необхідно розробити ефективні методи виявлення та діагностики станів логічних об'єктів, що забезпечують роботу інформаційної системи підприємства.

Одним із методів забезпечення безпеки інформаційної системи підприємства є виявлення шкідливих процесів в системі та діагностування стану логічних об'єктів. Такий метод дозволяє вчасно виявляти зміни в стані системи та забезпечувати реакцію на них. Ідентифікація та діагностування станів логічних об'єктів є важливим етапом при виявленні шкідливих процесів в системі. Він дозволяє перевіряти стан системи на відповідність з заданими параметрами та специфікаціями.

Існують різні методи виявлення шкідливих процесів в інформаційній системі підприємства, проте не всі з них є ефективними. Застосування недостатньо ефективних методів може призвести до недостатнього рівня захисту інформаційної системи та виникнення проблем в роботі системи. Тому, розробка нових методів виявлення шкідливих процесів є важливою задачею для забезпечення безпеки інформаційної системи.

Тема «метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів» є актуальною в сучасних умовах, оскільки інформаційні технології стали невід'ємною частиною бізнесу та повсякденного життя. Зараз підприємства активно використовують інформаційні системи для забезпечення своєї діяльності, тому надійність та безпека цих систем є дуже важливою. Проте інформаційні системи підлягають різним шкідливим

процесам, таким як хакерські атаки, віруси, збої у роботі обладнання та програмного забезпечення тощо. Ці шкідливі процеси можуть призвести до втрати даних, порушення роботи інформаційної системи та значного збитку для підприємства.

Отже, розробка та впровадження методу виявлення шкідливих процесів інформаційної системи підприємства на основі ідентифікації та діагностування станів логічних об'єктів може стати ефективним рішенням для підвищення безпеки інформаційної системи підприємства. Даний метод дозволяє виявляти можливі загрози безпеці та реагувати на них у найкоротший термін.

Застосування такого методу може бути особливо ефективним в разі виявлення внутрішніх загроз безпеці, які можуть виникати в результаті дій недоброзичливих працівників, які мають доступ до системи. Діагностування станів логічних об'єктів дозволяє виявити незвичайні зміни в поведінці системи, що можуть свідчити про зловживання доступом до інформації.

Дослідженням даного питання займалися як українські, так і закордонні вчені, серед яких: Людмила Кіріченко, Петро Зінченко, Тамара Радівілова, Максим Тавалбех, Галина Гайдур, Сергій Гахов, Світлана Казмірчук, Junjie Zhang, Jihong Liu та Yi Wang, Ivan Homoliak, Petr Hanacek, Harsha A K, Thayagaraja Murthy.

Недоліки методів, що використовуються цими науковцями включають:

— потреба у великій кількості даних для навчання моделей;

— помилкові спрацювання внаслідок недостатньої кількості даних або обмеженої точності моделей;

— нестабільність моделей машинного навчання при роботі з великими об'ємами даних;

— високу кількість помилок при виявленні аномалій;

— невміння розпізнати нові типи загроз, які не були включені до тренувальних даних;

— складність налаштування моделей машинного навчання та потреба в великій кількості обчислювальних ресурсів для їх тренування та експлуатації.

Таким чином, з приведеного аналізу публікацій можна зробити висновок, що існує протиріччя між необхідністю оперативного ідентифікування (в режимі реального часу) шкідливих процесів, якісного виявлення та зменшення показника хибно-позитивних спрацювань.

У зв'язку з цим, існує необхідність вирішення *актуального наукового завдання*, сутність якого полягає в розробці методу виявлення шкідливих процесів в реальному часі на основі ідентифікації та діагностування станів логічних об'єктів інформаційної системи підприємства.

Зв'язок роботи з науковими програмами, планами, темами.

Напрям дисертаційного дослідження безпосередньо пов'язаний з реалізацією Закону України «Про основні засади забезпечення кібербезпеки України», доктрини інформаційної безпеки України, наказу Адміністрації Держспецзв'язку «Про затвердження Методичних рекомендацій щодо підвищення рівня кіберзахисту критичної інформаційної інфраструктури», Стратегії інформаційної безпеки та Стратегії кібербезпеки України. Дисертаційна робота виконана відповідно до планів наукової і науково-технічної діяльності Державного університету телекомунікацій в рамках науково-дослідних робот «Розробка методів та засобів підвищення живучості інформаційно-комунікаційних систем в умовах кібернетичних атак» (№ 0114U000391, ДУТ, м. Київ) та «Методологія виявлення шкідливих процесів в інформаційних системах (№ 0121U113613, м. Київ).

Мета і завдання дослідження.

Підвищення ефективності роботи інформаційної системи підприємства за рахунок зменшення часу на пошук та вирішення проблем, що виникають у процесі її роботи шляхом вчасного виявлення можливих проблем та шкідливих процесів, що можуть впливати на її функціонування на основі ідентифікації та діагностування станів логічних об'єктів.

У відповідності до поставленої мети, для вирішення наукового завдання, в роботі сформульовані такі часткові завдання дослідження:

1) розробити модель розширеного скінченного автомату функціонування транспортного протоколу TCP;

2) розробити метод вибір критерії параметрів логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів;

3) розробити метод виявлення шкідливих процесів на основі машинного навчання з метою покращення ефективності в інформаційній системі підприємства в реальному часі;

4) дослідити ефективність методу виявлення шкідливих процесів в інформаційній системі підприємства та розробити рекомендації щодо його застосування та провадження.

Об’єкт дослідження — процес виявлення шкідливих процесів в інформаційній системі підприємства.

Предметом дослідження є метод виявлення шкідливих процесів в інформаційній системі підприємства.

Методи дослідження. Для проведення досліджень в дисертаційній роботі використовувалися методи теорії скінчених автоматів, методи кореляційної теорії, методи мета-аналізу, системного аналізу, методів машинного навчання.

Наукова новизна одержаних результатів

1) вперше розроблено модель ідентифікації та діагностування станів логічних об’єктів, наукова новизна якого полягає в тому, що вона ґрунтується на теорії скінченних автоматів на основі протоколу TCP, що дозволяє на стадії створення з’єднання ідентифікувати порушення пакету зі встановленими параметрами та діагностувати роботу транспортного протоколу;

2) удосконалено метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню, щодо виявлення шкідливих процесів, в якому на відміну від існуючих, на основі методу основних

компонентів обґрунтовано вибір параметрів вхідних даних логічного об’єкту моделі розширеного скінченного автомату TCP протоколу. Такий підхід дозволяє прискорити обробку даних при виявленні шкідливих процесів в інформаційній системі організації в реальному часі;

3) удосконалено метод виявлення шкідливих процесів в інформаційній системі організації, який на відміну від існуючих, базується на алгоритмах машинного навчання та враховує обрані параметри які підлягають ідентифікації та діагностуванню. Такий метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі.

Практичне значення одержаних результатів полягає в тому, що в дослідженні розроблено метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об’єктів. Даний метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі на 65-99% (в залежності від використаного ядра та гіперпараметру *C*) від існуючих сучасних методів та зменшення кількості хибно-позитивних спрацювань на 13-14%.

Результати досліджень прийняті до впровадження в діяльність ТОВ «Смартс» та в ТОВ «ЄВРОТЕЛЕКОМ».

Особистий внесок здобувача.

Основні наукові та прикладні результати дисертаційної роботи, що виносяться до захисту, отримані автором особисто. У наукових роботах, що опубліковані у співавторстві, автору належать: в [26] ефективне розвідування загроз у підприємствах, що потребують побудови захищеної IT-інфраструктури. В [98] аналіз методу виявлення вторгнень. В [99] дослідження логічних об’єктів в інформаційних системах. Проаналізовано переваги та недоліки використанні приманок для хакерів в [100]. В роботі [102] проведено дослідження ефективності удосконаленого методу автоматичного активного аналізу захищеності корпоративної мережі. В роботі [108] було досліджено

вразливості функціонування бездротових мереж. В [110] виявлення залежності уразливостей окремих компонентів системи та мережевий статус компонентів.

Апробація результатів дисертації.

Основні теоретичні та практичні результати були представлені та обговорені в ході ряду наукових конференцій:

— IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT) (15-17 Dec 2021);

— IX Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій» (20 жовтня 2019р., м. Київ);

— V міжнародна науково-практична конференція «актуальні проблеми забезпечення Інформаційної та кібернетичної безпеки» (17-18 грудня 2018р., м. Київ);

— V Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій» (07 грудня 2017р., м. Київ).

Публікації. За результатами дисертаційних досліджень опубліковано 12 наукових праць [1-12], серед яких [2, 5, 7-9] опубліковані у спеціалізованих фахових виданнях, затверджених наказом МОН України, [1-3] опубліковано у закордонному науковому виданні в тому числі Scopus [1-2]. Матеріали виступів на науково-технічних конференціях опубліковано у 4-х збірниках тез доповідей [1, 10-12].

Структура та обсяг дисертаційної роботи.

Дисертація складається зі вступу, трьох розділів, висновків, списку використаних джерел із 111 найменувань на 14 сторінках. Загальний обсяг роботи становить 160 сторінки серед яких 126 сторінок основного тексту, 22 рисунок, 15 таблиць.

РОЗДІЛ 1 АНАЛІЗ СТАНУ ТА ПОСТАНОВКА ЗАВДАННЯ ЩОДО РОЗРОБКИ МЕТОДУ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОЦЕСІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ ПІДПРИЄМСТВА НА ОСНОВІ ІДЕНТИФІКАЦІЇ ТА ДІАГНОСТУВАННЯ СТАНІВ ЛОГІЧНИХ ОБ'ЄКТІВ

1.1. Аналіз механізмів захисту інформаційної системи підприємства

Сучасний етап економічного та технологічного розвитку більшості країн світу супроводжується стрімким розвитком інформаційно-комунікаційних технологій. Зокрема, сьогодні однією з найважливіших частин інфраструктури більшості організацій у світі є комп'ютерна мережа, яку часто називають корпоративною мережею [100-111].

Корпоративна мережа (КМ) – це інтегроване середовище, яке забезпечує широкий спектр телекомунікаційних послуг від традиційної передачі даних, електронної пошти, телефонії, роботи в режимі online з базами даних до охорони та відеоспостереження, а основним призначенням – є забезпечення максимально ефективної, ергономічної та захищеної роботи конкретного підприємства або організації [79, 97, 100].

Основні механізми захисту інформаційної системи підприємства (ІСП) відіграють важливу роль в забезпеченні конфіденційності, цілісності та доступності даних. Вони забезпечують захист від зловмисних атак, вірусів, черв'яків, троянських програм і інших загроз, що можуть вплинути на роботу системи та безпеку даних [80, 101].

Основні механізми захисту ІСП включають:

Аутентифікацію - механізм, що забезпечує перевірку ідентифікаційних даних користувача для доступу до системи.

Авторизацію - механізм, що контролює доступ користувачів до різних ресурсів системи відповідно до їхніх прав.

Шифрування - механізм, що застосовується для захисту конфіденційності даних шляхом перетворення їх у криптографічно надійний формат.

Резервне копіювання - механізм, що забезпечує збереження даних в разі їх втрати або пошкодження.

Моніторинг - механізм, що дозволяє виявляти інциденти безпеки в режимі реального часу та надавати інформацію про них адміністратору системи.

Аудит - механізм, що забезпечує збір та аналіз журналів подій для виявлення порушень безпеки та відновлення роботи системи.

Безпеку мережі - механізм, що забезпечує захист мережі від зловмисних атак, вірусів та інших загроз [80, 97, 101].

Слід відзначити, що в корпоративних мережах можуть бути присутні як внутрішні загрози з боку недосвідчених користувачів, так і зовнішні з можливістю проведення кібератак, при чому, в залежності від своїх цілей, зловмисники можуть реалізовувати цілі стратегії нападу, які складаються з багатокрокових ланцюгів атакуючих дій. В якості визначення поняття атаки на корпоративні мережі будемо використовувати наступне: атака – це будь-яка несанкціонована дія зловмисника скерована на пошук, виявлення та використання слабких місць, вразливостей компонентів, як самої КМ, так і компонентів системи її захисту [7].

Шкідливий процес інформаційної системи підприємства (ШП) - це будь-яка небезпечна діяльність або подія, яка може завдати шкоди або порушити нормальну роботу інформаційної системи підприємства. ШП може бути спричинений зловмисною діяльністю з боку злочинців або звичайних користувачів системи, а також може бути наслідком помилкової дії адміністратора системи або природної катастрофи. ШП може мати різноманітні форми прояву, такі як віруси, черви, троянські програми,

фішингові атаки, DoS/DDoS атаки тощо [79, 88, 100, 106]. В цілому, ШП є небезпечним для інформаційної безпеки підприємства, оскільки може призвести до втрати конфіденційної інформації, порушення нормального функціонування інформаційних систем, а також призвести до фінансових втрат підприємства.

Таким чином, сьогодні, питання надійного забезпечення інформаційної безпеки корпоративних мереж є особливо актуальним, що також підтверджується даними з останніх опублікованих звітів Рис. 1.1.

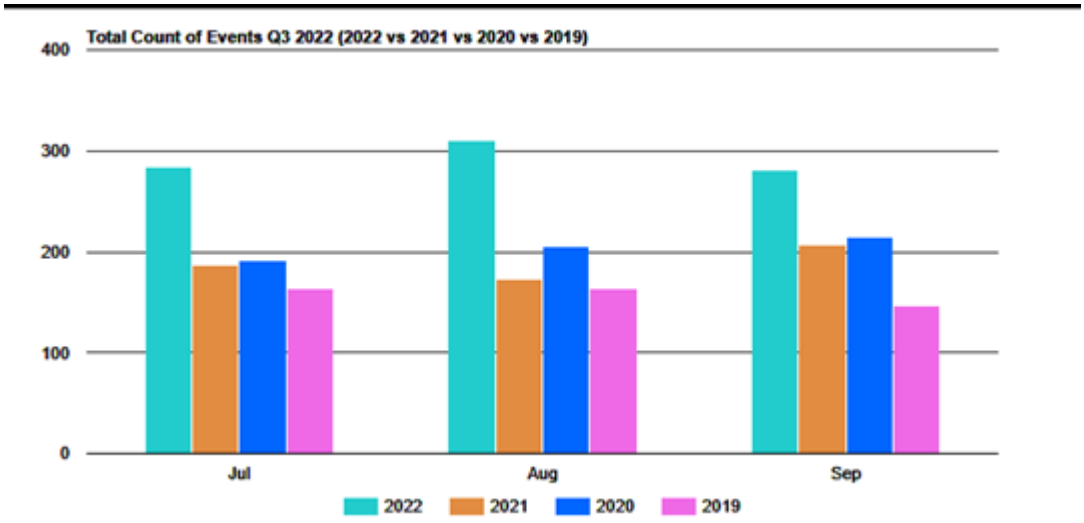

Рис.1.1. Втручання зловмисників в інформаційну систему підприємства за 2019-2022 роки [64]

Щодо методів, які використовуються для реалізації шкідливих процесів, тут необхідно відзначити, що велику частину займають шкідливе програмне забезпечення, викрадення акаунтів, таргетовані атаки, вразливості системи та DDoS, згідно з рис. 1.2 [64].

Одним із актуальних напрямів забезпечення кібербезпеки корпоративних мереж вважається впровадження механізмів, до яких належать в тому числі методи виявлення шкідливих процесів. Але вони мають свої слабкі сторони. Тому ідея методу виявлення ШП полягає у детектуванні станів

логічних об’єктів протоколу TCP для завчасного реагування на порушення інформаційної системи підприємства [104].

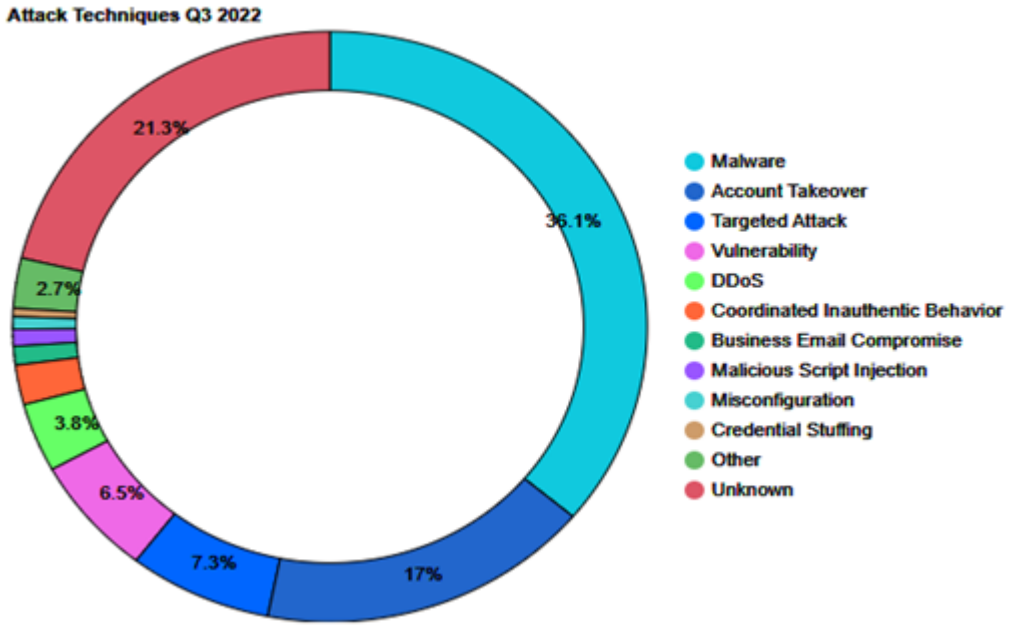

Рис. 1.2. Статистика найчастіших шкідливих процесів за 2022 рік [64]

Виявлення ШП в інформаційних системах підприємства є важливою складовою інформаційної безпеки. Вона вимагає ідентифікації та діагностики стану логічних об'єктів у системі.

Ідентифікація. Першим кроком у виявленні шкідливих процесів є визначення їх наявності. Це можна зробити різними способами, такими як моніторинг мережевого трафіку, аналіз системних журналів і використання систем виявлення вторгнень. Моніторинг мережевого трафіку може виявити шкідливі програми та дії, а аналіз журналу може виявити підозрілу поведінку [80, 101]. Системи виявлення вторгнень виявляють відомі загрози, а також можуть сповіщати адміністраторів про підозрілу активність.

Діагностика. Після того, як виявлено наявність шкідливого процесу, необхідно провести діагностику стану системи [80, 101]. Це можна зробити різними способами, наприклад аналізом системних конфігурацій, перевіркою системних файлів і перевіркою на наявність аномалій. Аналіз конфігурації системи може виявити неправильні конфігурації, які можуть вказувати на

наявність шкідливих процесів. Перевірка системних файлів може виявити наявність підозрілих програм або файлів, які були встановлені без відома адміністратора. Перевірка на наявність аномалій також може виявити підозрілу поведінку, наприклад несподівані підключення або доступ до системи.

Виходячи з цього, під захищеністю ІС будемо розуміти певний стан логічного об’єкта.

1.2. Підходи машинного навчання щодо виявлення шкідливих процесів в інформаційній системі організації

Замість того, щоб мати справу з необробленим шкідливим програмним забезпеченням, процес підготовки даних традиційних підходів до машинного навчання передбачає попередню обробку виконуваного файлу для вилучення набору функцій, які забезпечують абстрактне уявлення про програмне забезпечення [5]. Після цього функції використовуються для навчання моделі вирішувати поставлене завдання. Через різноманітність функціональних можливостей зловмисного програмного забезпечення важливо не лише виявляти зловмисне програмне забезпечення, а й розрізняти різні типи зловмисного програмного забезпечення, щоб краще зрозуміти їхні можливості. У машинному навчанні робочий процес — це ітеративний процес, який включає збір доступних даних, очищення та підготовку даних, створення моделей, перевірку та розгортання у виробництві [30-35]. Рис. 1.3.

Основна відмінність між рішеннями машинного навчання для виявлення або класифікації зловмисного програмного забезпечення полягає в результатах, які повертає реалізована система.

З одного боку, система виявлення зловмисного програмного забезпечення видає одне значення $y = f(x)$, в діапазоні від 0 до 1, що вказує на шкідливість виконуваного файлу. З іншого боку, система класифікації видає ймовірність належності даного виконуваного файлу до кожного класу або сімейства виводу, $y \in R^N$, де N вказує на кількість різних родин [34].

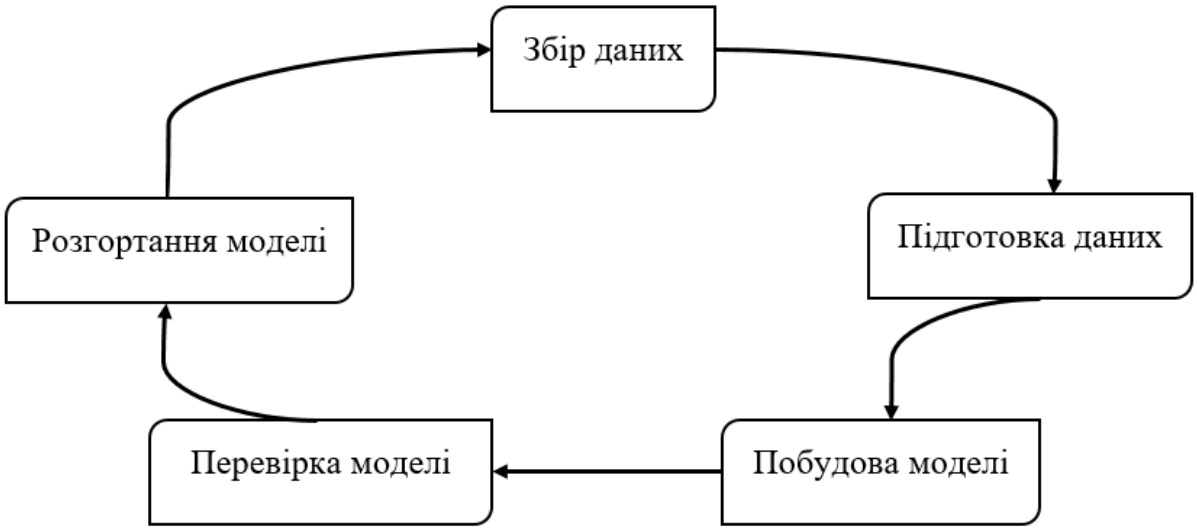

Рис. 1.3. Робочий процес машинного навчання

Класифікація функцій наведена на рис. 1.4. Відповідно, типи функцій можна розділити на дві групи, як і підходи до аналізу зловмисного програмного забезпечення: (1) статичні функції та (2) динамічні функції. Кожен тип функції окремо нижче [30-35].

Static Features (статичні функції) — статичні функції витягуються з частини програми без участі її виконання. У портативних виконуваних файлах Windows статичні функції в основному отримують із двох джерел інформації: двійкового вмісту виконуваного файлу або вихідного файлу мови асемблера, отриманого після декомпіляції та розбирання двійкового виконуваного файлу.

Strings (аналіз рядків) — аналіз рядків означає виділення кожного рядка, який можна роздрукувати, у виконуваному файлі чи програмі. Рядок відноситься до послідовності символів. Пошук рядків є найпростішим

способом отримати підказки про функціональність програми. Інформацією, яку можна знайти в цих рядках, можуть бути, наприклад, URL-адреси, до яких підключається програма, розташування файлів або шляхи до файлів, до яких програма звертається/змінює їх, назви меню програми тощо [30-35]. Утиліта під назвою «Рядки» можна використовувати для пошуку у виконуваному файлі рядків ASCII і Unicode, ігноруючи контекст і форматування.

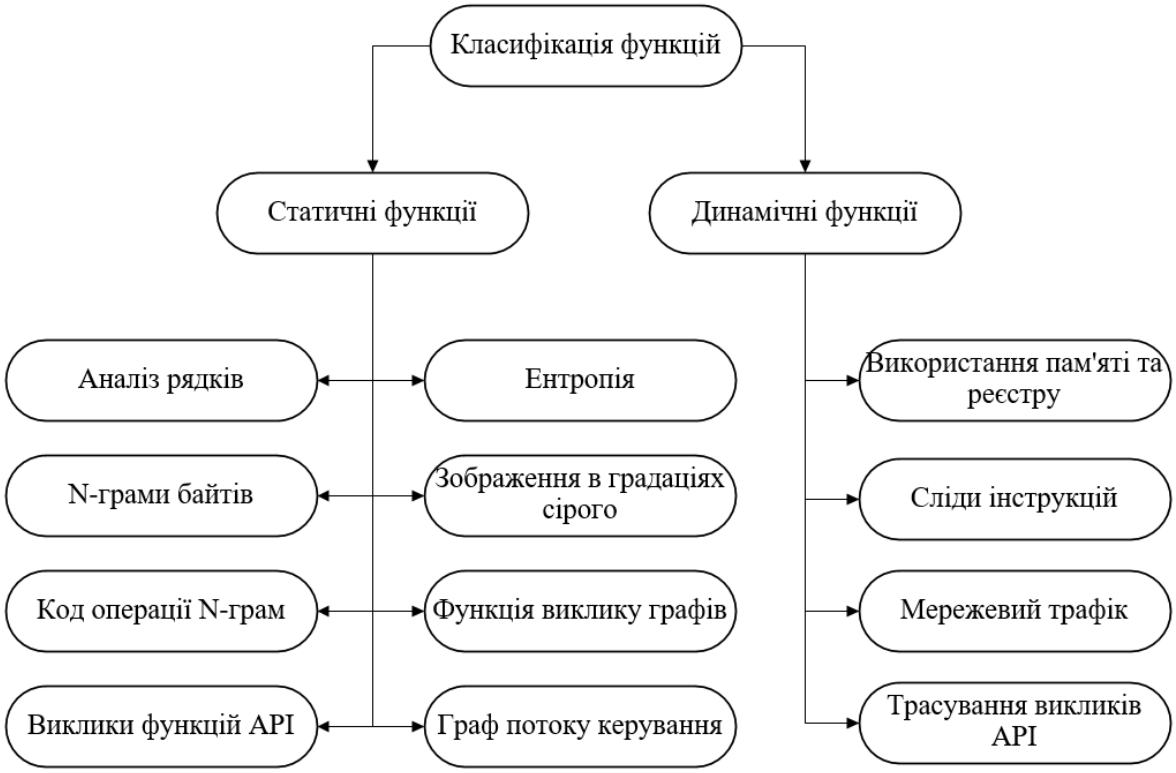

Рис. 1.4. Таксономія функцій, які використовуються традиційними підходами машинного навчання [34]

Bytes N-gram, Opcode N-grams (N-грами байтів і коду операції) — найпоширенішим типом функцій для виявлення та класифікації шкідливих програм є n-грами. N-грама — це безперервна послідовність із n елементів із заданої послідовності тексту. N-грами можна отримати з послідовностей байтів, що представляють бінарний вміст шкідливого програмного забезпечення, і з вихідного коду мови асемблера [21, 34]. Розглядаючи файл як послідовність байтів, n-грами байтів витягуються шляхом розгляду

унікальної комбінації кожних n послідовних байтів як окремої функції. З іншого боку, послідовність інструкцій мови асемблератакож можна отримати з вихідного коду мови асемблера. У цьому випадку зберігається лише мнемоніка інструкції, тобто «ADD», «MUL», «PUSH» тощо. Таким чином, код операції або мнемонічні n-грами стосуються унікальної комбінації кожних n послідовних кодів операції як окремої функції [34,77].

Незважаючи на успіх у виявленні шкідливих програм, підходи n-gram мають деякі проблеми, про які варто згадати. По-перше, вичерпно перераховувати всі n-грами непрактично та обчислювально важко. Оцінка параметрів моделі, коли кількість ознак перевищує кількість зразків, може призвести до прокляття розмірності. Як наслідок, необхідно використовувати методи вибору та зменшення ознак. По-друге, дослідники [8] дійшли висновку, що байтові n-грами, здається, вивчають переважно вміст рядків у виконуваному файлі, зокрема елементи з заголовка PE. Оскільки існують мільйони потенційних n-грамів (для більшого n), методи вибору ознак мають тенденцію вибирати як ознаки ті, які трапляються досить часто. Це заохочує вибір функцій з низькою ентропією, які складаються переважно з рядків і заповнення. По-третє, незалежно від того, які n-грами вивчені, ми повинні отримати точний збіг під час класифікації нового зразка. Отже, будь-яка незначна зміна призведе до того, що ця функція не з’явиться і, таким чином, не вплине на нашу модель. Таким чином, цей брак узагальнення є потенційним джерелом надмірного підгонки.

API Function Calls (виклики функцій API) — інтерфейси прикладного програмування (API) та їх виклики функцій вважаються дуже дискримінаційними функціями. Література показала, що виклик функцій API можна використовувати для моделювання поведінки програми. По суті, функції API та системні виклики пов’язані зі службами, що надаються операційними системами, такими як мережі, безпека, керування файлами тощо [34,77]. Оскільки для програмного забезпечення немає іншого способу доступу до системних ресурсів без використання функцій API, виклик певних

функцій API надає ключову інформацію для представлення поведінки зловмисного програмного забезпечення.

Entropy-based (ентропія) — автори зловмисного програмного забезпечення часто використовують різноманітні методи обфускації, щоб приховати зловмисне призначення виконуваних файлів. Два найбільш часто використовуваних — це стиснення та шифрування, які використовуються для приховування шкідливих сегментів від статичного аналізу [34,82]. Отже, для індустрії інформаційної безпеки представляє великий інтерес можливість виявити наявність зашифрованих або стиснутих сегментів коду у виконуваних файлах. Для цього використовувався ентропійний аналіз, оскільки файли з сегментами коду, які були стиснуті або зашифровані, як правило, мають вищу ентропію, ніж рідний код [90]. У контексті теорії інформації ентропія послідовності байтів відображає її статистичну варіацію. Зокрема, нульова ентропія означатиме, що той самий символ повторювався в аналізованому сегменті. Таку поведінку можна спостерігати у «доповненому» фрагменті коду. Навпаки, високе значення ентропії вказуватиме на те, що фрагмент повністю складається з різних значень. Наприклад, [58] проаналізував групу файлів, що складається з простих текстових файлів, власних, стиснених і зашифрованих виконуваних файлів, і помітив, що середня ентропія виконуваних файлів становила 5,09; 6,80 і 7,17 відповідно.

У результаті попередні дослідження використовували високу середню ентропію для визначення наявності шифрування та стиснення. Однак, коли шкідливий код приховано складним чином, його може бути важко виявити за допомогою такої простої статистики ентропії. Поширеним підходом до зменшення ентропії файлу є додавання інструкцій «nop». Тим не менш, файли із зашифрованими, стисненими, рідними або доповненими сегментами, як правило, мають різні та унікальні рівні ентропії. Таким чином, дослідники [82] почали аналізувати так звану структурну ентропію файлу, представлення послідовності байтів шкідливого програмного забезпечення у вигляді потоку значень ентропії, де кожне значення вказує на кількість ентропії в невеликій

частині файлу. код у певному місці. Зокрема, [82] порівняли подібність між структурною ентропією невідомого файлу та навчального набору даних для виявлення шкідливих програм.

IMG-based (представлення зловмисного програмного забезпечення як зображення в градаціях сірого) — цікавий підхід для візуалізації зловмисного програмного забезпечення вперше представив.

В роботі [61], візуалізували двійковий вміст шкідливого програмного забезпечення як зображення в градаціях сірого. Це досягається шляхом інтерпретації кожного байта як одного пікселя в зображенні, де значення коливаються від 0 до 255 (0:чорний, 255:білий). Після цього отриманий масив реорганізується як 2-D масив.

На рис. 1.5 представлені зразки з двох сімейств шкідливих програм [34], представлені у вигляді зображень у градаціях сірого. Помітно, що представлення зображень зразків певної родини досить схоже, але відрізняється від того, що належить іншій родині. Ця візуальна подібність є результатом повторного використання коду для створення нових двійкових файлів. Таким чином, якщо старі зразки повторно використовувати для впровадження нових двійкових файлів, отримані зразки будуть подібними. У більшості випадків, представляючи виконуваний файл як зображення в градаціях сірого, можна було б виявити невеликі варіації між зразками, що належать до одного сімейства.

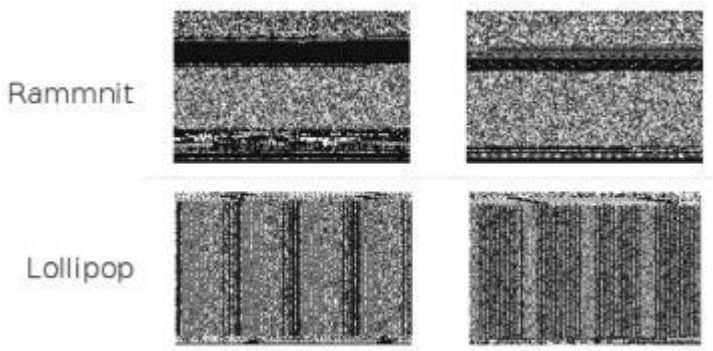

Рис. 1.5. Відображення у градаціях сірого двійкового вмісту зразків зловмисного програмного забезпечення сімейств Ramnit і Lollipop [34]

Цю візуальну подібність використовували різні автори для виявлення та класифікації зловмисного програмного забезпечення. Зокрема, [61] витягли ознаки GIST із представлення двійкового вмісту зловмисного програмного забезпечення в градаціях сірого. Нарешті, новий виконуваний файл класифікується в тому чи іншому сімействі за допомогою алгоритму K-Nearest Neighbor (K-NN) з евклідовою відстанню як метрикою. [4] витягнув функції Haralick і Local Binary Pattern для класифікації зловмисного програмного забезпечення за допомогою класифікаторів розширеного дерева.

Автор [43] виділив три набори ознак: (1) на основі інтенсивності, (2) на основі вейвлетів і (3) на основі Габора. Зокрема, вони витягли середню інтенсивність, дисперсію, моду, асиметрію, ексцес і кількість пікселів із значенням інтенсивності 0 і 255. Щодо функцій на основі вейвлетів, вони застосували вейвлет-декомпозицію рівня 3 за допомогою Добешівейвлет, також відомий як db4, і вони отримали один набір приблизних коефіцієнтів і три набори детальних коефіцієнтів. Характеристиками, витягнутими з кожного з цих коефіцієнтів, були середнє значення, дисперсія, максимальне та мінімальне значення. Нарешті, щоб витягти функції на основі Габора [43], вони застосували до зображення фільтр Габора (згортка вхідних даних із функцією Габора). Ефективність опорних векторних машин як алгоритму навчання оцінювалася на наборі даних із 15 000 шкідливих і 12 000 безпечних зразків, де 70 % використовувалися для навчання, а 30 % — для тестування.

Представлення зображень у градаціях сірого програмного забезпечення має деякі недоліки, безпосередньо пов’язані з тим, як генеруються зображення. По-перше, двійкові файли не є двовимірними зображеннями, і, перетворюючи їх як такі, ви вводите непотрібні попередні. По-перше, щоб створити зображення, вам потрібно вибрати ширину зображення, яка додає новий гіперпараметр для налаштування [34, 43]. Зауважте, що вибір ширини відповідно визначає висоту зображення залежно від розміру двійкового файла. По-друге, він накладає неіснуючі просторові кореляції між пікселями в різних рядках, що може бути неправдою.

Крім того, як і більшість статичних функцій, він страждає від методів обфускації коду. Зокрема, такі методи, як шифрування та стиснення, можуть повністю змінити структуру байтів двійкової програми, і, отже, методи, засновані на такому представленні, не зможуть правильно класифікувати її клас. Це можна спостерігати в сірому представленні зразків, що належать до автозапуску. Сімейства з набору даних MalImg [61], які майже однакові через те, що обидва були стиснуті за допомогою пакувальника UPX.

Function Call Graph (функція виклику графів) — це спрямований граф, вершини якого представляють функції, з яких складається програма, а ребра символізують виклики функцій. Вершина представлена одним із двох типів функцій:

1. Локальні функції, реалізовані програмістом для виконання конкретних завдань.

2. Зовнішні функції: надаються ОС або системою та зовнішніми бібліотеками.

Особливістю графа є те, що лише локальні функції можуть викликати зовнішні функції, а не навпаки [13]. Графіки викликів функцій створюються на основі статичного аналізу файлу розбирання. Щоб видобути FCG виконуваних файлів Windows PE, можна використовувати IDA Pro або Radare2.

Control Flow Graph (граф потоку керування) — це орієнтований граф, у якому вузли представляють основні блоки, а ребра — шляхи потоку керування. Базовий блок — це лінійна послідовність програмних інструкцій, яка має точку входу (перша виконана інструкція) і точку виходу (остання виконана інструкція) [34]. CFG — це представлення всіх шляхів, які можна пройти під час виконання програми.

Dynamic Features (динамічні функції) — це функції, отримані під час виконання зловмисного програмного забезпечення під час виконання. Динамічний аналіз передбачає моніторинг зловмисного програмного забезпечення (і спостереження за реальною послідовністю виконаних

інструкцій або послідовності запущених функцій API) під час його роботи або перевірку системи після виконання зловмисного програмного забезпечення [34]. Він розкриває процес створення, маніпулювання файлами та реєстром, а також модифікацію значень пам’яті, регістрів і змінних.

Memory and Register’s Usage (використання пам'яті та реєстру) — поведінка комп'ютерної програми може бути представлена значеннями вмісту пам'яті під час виконання [34]. Іншими словами, значення, що зберігаються в різних реєстрах під час роботи комп’ютерної програми, можуть відрізнити доброякісні програми від шкідливих.

Instructions Traces (сліди інструкцій) — динамічна траса інструкцій — це послідовність інструкцій процесора, що викликається під час виконання програми. На відміну від трасування статичних інструкцій, динамічні трасування впорядковуються в міру їх виконання, тоді як статичні трасування впорядковуються так, як вони з’являються у двійковому файлі [34]. Динамічні траси є більш надійним показником поведінки програми, оскільки пакувальники коду та шифрувальники можуть заплутувати та перешкоджати статичному аналізу інструкцій коду.

Network Traffic (мережевий трафік) — виявлення зловмисного трафіку в мережі може надати унікальне розуміння поведінки шкідливих програм. Щойно зловмисне програмне забезпечення заражає хост-машину, воно може встановити зв’язок із зовнішнім сервером, щоб отримати команди для виконання на жертві або завантажувати оновлення, інше зловмисне програмне забезпечення або витік конфіденційної та конфіденційної інформації користувача/пристрою [34, 80]. Як наслідок, моніторинг мережевого трафіку, що входить і виходить з мережі, трафіку всередині мережі та активності хоста, надає корисну інформацію для виявлення зловмисної поведінки. Підходи в літературі виділяють події на кількох рівнях абстракції, від необроблених пакетів до мережевих потоків, детального декодування протоколу, наприклад запитів HTTP та DNS, до подій на основі хоста та метаданих, таких як IP-адреси, порти та кількість пакетів.

API Call Traces (трасування викликів API) — програмісти використовують Windows API для доступу до основних ресурсів, доступних системі Windows, включаючи, але не обмежуючись цим, файлові системи, пристрої, процеси, потоки та обробку помилок, а також для доступу до функцій поза межами ядра, таких як реєстр Windows, запуск/зупинити/створити службу Windows, керувати обліковими записами користувачів тощо [34]. Отже, трасування викликів Windows API використовувалися в літературі для фіксації поведінки шкідливих програм.

Підходи до глибокого навчання. Наведені вище традиційні підходи до машинного навчання покладаються головним чином на функції, створені вручну, на основі експертних знань про предметну область [8, 95]. Ці рішення забезпечують абстрактне уявлення про зловмисне програмне забезпечення, яке використовує класифікатор машинного навчання, наприклад, нейронна мережа, дерево рішень, опорна векторна машина тощо, щоб прийняти рішення. Розробка функцій і вилучення функцій є ключовими, трудомісткими процесами робочого процесу машинного навчання. Слідкуючи за останніми тенденціями в області комп’ютерного зору та обробки природної мови. Розробка рішень ML для виявлення зловмисного програмного забезпечення почала прямувати до архітектур глибокого навчання [33, 34, 94]. Ці рішення замінили вищезгаданий процес розробки функцій робочого процесу ML на систему, яка повністю піддається навчанню, починаючи від необробленого введення до остаточного виведення розпізнаних об’єктів.

Підходи глибокого навчання для вирішення проблеми виявлення та класифікації зловмисного програмного забезпечення можна класифікувати на різні групи залежно від того, як вхідні дані попередньо обробляються перед подачею в алгоритм навчання.

Мультимодальні підходи. Було представилено підходи, які здебільшого покладаються на один тип функції або модальності даних для виявлення та класифікації зловмисного програмного забезпечення. Однак виявлення зловмисного програмного забезпечення є дослідницькою

проблемою, яка характеризується як мультимодальна, оскільки включає в себе кілька модальностей даних [97]. Мультимодальне навчання – це сфера, яка вивчає, як інтерпретувати такі мультимодальні сигнали разом. Хоча поєднання різних модальностей або типів інформації для покращення продуктивності здається інтуїтивно привабливим завданням, дуже складно поєднати різні рівні шуму та конфлікт між модальностями. Мультимодальні підходи можна класифікувати на три групи з огляду на те, як поєднуються численні модальності [34].

Вхідний рівень або раннє злиття. Ранні методи синтезу створюють спільне представлення унімодальних особливостей, виділених окремо з кількох модальностей. Найпростіший спосіб об'єднати ці унімодальні вектори ознак – це об'єднати їх для отримання злитого представлення. Далі одна модель навчається для вивчення кореляції та взаємодії між особливостями кожної модальності. Кінцевий результат моделі можна записати як

$$p = h([v_1, v_2, \dots, v_m]) \tag{1.1}$$

де h позначає одну модель $[v_1, v_2, \dots, v_m]$, представляє конкатенацію векторів ознак, а m є кількістю різних унімодальних векторів ознак.

Рівень прийняття рішення або пізній синтез. На відміну від раннього злиття, методи пізнього злиття навчають одну модель на модальність і об'єднують отримані значення рішення з механізмом злиття, таким як усереднення, голосування, навчена модель тощо. Основна перевага пізнього злиття полягає в тому, що він дозволяє використовувати різні моделі на різних модальностях, таким чином, будучи більш гнучким. Крім того, оскільки прогнози для кожної модальності робляться окремо, легше обробляти відсутні модальності. Припускаючи, що модель h i є значенням рішення щодо модальності i, остаточний прогноз є

$$p = F(h_1(v_1), h_2(v_2), \dots, h_m(v_m)) \quad (1.2)$$

де F позначає тип стратегії синтезу.

Проміжне злиття. Методи проміжного злиття створюють спільне представлення шляхом об’єднання проміжних функцій, отриманих за допомогою окремих моделей машинного навчання [30-34]. Згодом ці проміжні функції об’єднуються, а потім навчається модель машинного навчання, щоб фіксувати взаємодію між модальностями.

Крім того, функції, витягнуті з обох типів аналізу, статичного та динамічного, можна комбінувати для створення надійніших класифікаторів. Підходи, які поєднують статичний аналіз і динамічний аналіз, відомі як гібридні підходи.

З одного боку, статичний аналіз спрямований на пошук шкідливих характеристик виконуваного файлу, додатка чи програми без їх фактичного запуску. Статичний аналіз швидший, але страждає від обфускації коду. Тобто шкідливі характеристики можна приховати за допомогою різних методів обфускації [94] або поліморфних і метаморфічних шкідливих програм [58]. З іншого боку, ця техніка обфускації не дає змоги динамічному аналізу, оскільки вона відстежує та аналізує поведінку програми під час її виконання в контрольованому середовищі. Але є деякі обмеження для динамічного аналізу. Процес моніторингу займає багато часу, і середовище, де запускається програма, має бути захищеним, щоб не заразити платформу. Крім того, контрольоване середовище може відрізнятися від реального середовища виконання і зловмисне програмне забезпечення може поводитися по-різному, спричиняючи неточну поведінку журналу [34]. Крім того, деякі дії програми запускаються лише за умови виконання певних умов і можуть не бути виявлені/активовані в контрольованому середовищі. Враховуючи переваги та недоліки статичного та динамічного виявлення зловмисного програмного забезпечення, природним удосконаленням і напрямом дослідження є зосередження на гібридних схемах, які поєднують елементи обох. Зауважте,

що гібридні підходи також включають різні модальності даних і можуть бути включені до однієї категорії. Основна відмінність між гібридним і мультимодальним підходами полягає в тому [48], що гібридні підходи поєднують характеристики як статичного, так і динамічного аналізу, тоді як мультимодальні підходи не повинні цього робити.

Отримання хороших навчальних даних є одним із найскладніших аспектів будь-якої проблеми машинного навчання. Класифікатори машинного навчання ефективні настільки, наскільки якісні дані, які використовуються для їх навчання, а надійні позначені дані особливо важливі для завдання виявлення зловмисного програмного забезпечення, коли процес позначення файлу може бути дуже трудомістким процесом.

1.3. Дослідження науково-методичних підходів виявлення аномалії

Джерела інформації, такі як мережеві пакети, створюють серйозні проблеми для методів виявлення аномалій з двох причин. По-перше, обсяг даних і простір можливих статистичних властивостей, що цікавлять, дуже великі. По-друге, необроблені мережеві пакетні дані часто є неструктурованими, що ускладнює відокремлення значущої інформації від «фонового шуму». Щоб вирішити цю проблему, необроблені пакетні дані часто обробляються для вилучення важливих «функцій», які вважаються цікавими. Цей процес значно зменшує обсяг даних, які система виявлення аномалій повинна обробити [13, 25, 57, 80]. Крім того, він визначає важливу інформацію з потоку пакетів, відкидаючи менш корисну інформацію.

Більшість дослідників, які виявляють аномалії, визнають важливість правильного вибору ознак. Наразі вибір функцій залежить від знань і суджень експертів щодо того, що є «корисною інформацією» для виявлення атаки. Незважаючи на те, що експерти-люди часто можуть ідентифікувати деякі

корисні функції, далеко не зрозуміло, що вони можуть виконувати комплексну роботу.

На їх сприйняття корисних функцій часто впливає знання відомих атак. Тому вони не обов’язково вибирають функції, корисні для виявлення невідомих атак.

У роботі [93] продемонстрований підхід до глибокого навчання який може автоматично виділяти функції високого рівня з низькорівневих і отримати потужне представлення та висновки. Розроблено рекурентну глибоку нейронну мережу, щоб вивчати шаблони з послідовностей мережевого трафіку та відстежувати дії мережевих атак.

В роботі [37] надані два набори даних, що містять кілька методів протилежної обфускації, які були застосовані як до зловмисних, так і до законних зразків трафіку під час «виконання» певних мережевих з’єднань TCP. Для ухилення від класифікаторів виявлення мережевих вторгнень на основі машинного навчання використовувалися методи обфускації. Зокрема, використана обфускація тунелювання в протоколі HTTP(S) і обфускації [62, 93], не засновані на корисному навантаженні, змінюючи різні властивості мережевого трафіку, наприклад, сегментацію TCP, повторну передачу, пошкодження та зміну порядку пакетів тощо. Надані набори даних дозволяють тестувати стійкість до ухилення довільного класифікатора, який використовує функції ASNM.

У роботі [42] запропоновано ефективний підхід до виявлення зловмисного програмного забезпечення з використанням алгоритмів класифікації в машинному навчанні, таких як метод опорних векторів, випадковий ліс і екстремальне підвищення градієнта. Використано широкий процес вибору функцій, щоб зменшити розмірність вибраного набору даних. Алгоритми машинного навчання навчаються за допомогою навчального набору. Алгоритми підсилення випадкового лісу та екстремального градієнта показали виняткові результати в експериментах статті. Робота демонструє, що трафік шкідливих програм можна ефективно класифікувати за допомогою

звичайних алгоритмів машинного навчання, а також показує важливість зменшення розмірності в таких проблемах класифікації.

Недоліки методів, що використовуються цими науковцями включають:

потреба у великій кількості даних для навчання моделей;

помилкові спрацювання внаслідок недостатньої кількості даних або обмеженої точності моделей;

нестабільність моделей ML при роботі з великими об'ємами даних;

високу кількість помилок при виявленні аномалій;

невміння розпізнати нові типи загроз, які не були включені до тренувальних даних;

складність налаштування моделей ML та потреба в великій кількості обчислювальних ресурсів для їх тренування та експлуатації [4, 16, 19, 42, 77].

В результаті проведеного дослідження науково-методичних підходів виявлення аномалій в інформаційних системах було встановлено, що використання машинного навчання та методів статистичного аналізу є ефективними способами для виявлення аномалій. Але не існує єдиного універсального підхоту, оскільки з кожним днем з’являються нові загрози інформаційним системам. Таким чином, можна зробити висновок, що виявлення аномалій є важливим етапом в забезпеченні безпеки інформаційних систем підприємства, і використання машинного навчання та методів статистичного аналізу є ефективними інструментами для вирішення схожих питань.

1.4. Аналіз протиріч та постановка наукового завдання

На сьогоднішній день, при динамічному зростанні вразливостей та шкідливих процесів, збільшується роль методів ідентифікації та діагностування станів логічних об’єктів ІС підприємства.

Одним з недоліків таких методів є недосконалість виявлення шкідливих процесів, які є специфічними інформаційним системам конкретного підприємства.

З іншого боку сутність, методів, які базуються на розпізнанні певних моделей атак; методів, які базуються на аналізі журналів подій; методів, які використовують машинне навчання полягає в балансуванні між точністю виявлення ШП та кількістю хибно-позитивних спрацювань в режимі реального часу.

Оскільки для точного виявлення ШП в інформаційній системі потрібно багато часу та ресурсів на обрахування. Подальший розвиток методів ідентифікації шкідливих процесів, може змінити кількість хибно-позитивних спрацювань в режимі реального часу в системах виявлення та протидії вторгнень.

Таким чином, можна зробити висновок, що існує протиріччя між необхідністю оперативного ідентифікування (в режимі реального часу) ШП, якісного виявлення та зменшення показника хибно-позитивних спрацювань рис. 1.6.

Рис. 1.6. Визначення наукового протиріччя та постановка наукового завдання

У зв’язку з цим, існує необхідність вирішення актуального наукового завдання, сутність якого полягає в розробці методу виявлення шкідливих процесів в реальному часі на основі ідентифікації та діагностування станів логічних об’єктів інформаційної системи підприємства.

Метою дисертаційного дослідження є підвищення ефективності роботи інформаційної системи підприємства за рахунок зменшення часу на пошук та вирішення проблем, що виникають у процесі її роботи шляхом вчасного виявлення можливих проблем та шкідливих процесів, що можуть впливати на її функціонування на основі ідентифікації та діагностування станів логічних об’єктів.

У відповідності до поставленої мети, для вирішення наукового завдання, в роботі сформульовані такі часткові завдання дослідження:

1) розробити модель розширеного скінченного автомату функціонування транспортного протоколу TCP;

2) розробити метод вибір критерії параметрів логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів;

3) розробити метод виявлення шкідливих процесів на основі машинного навчання з метою покращення ефективності в інформаційній системі підприємства в реальному часі;

4) дослідити ефективність методу виявлення шкідливих процесів в інформаційній системі підприємства та розробити рекомендації щодо його застосування та провадження.

Висновки до розділу 1

1. В результаті аналізу встановлено, що сучасний етап економічного та технологічного розвитку більшості країн світу супроводжується стрімким

розвитком інформаційно-комунікаційних технологій, на які націлені основні атаки та шкідливі процеси для виведення її з ладу, спотворення інформації та несанкціонованого доступу то даних. Будь-який шкідливий процес може бути направлений на інформаційну систему підприємства. Тому існує необхідність в створенні надійних систем захисту, а також методів виявлення та реагування на шкідливі процеси в реальному часі.

2. Опрацьований зміст сучасної літератури, де наведено широкий спектр методів виявлення аномалій в інформаційних системах підприємства показав, що динамічне зростання кількості вразливостей приводить до зростанню необхідності застосування превентивних методів забезпечення безпеки, до яких відноситься зокрема виявлення шкідливих процесів в інформаційній системі. Дані методи дозволяють виявляти шкідливі процеси та попереджувати можливості їх застосування з боку зловмисників шляхом реагування.

3. В результаті аналізу існуючих методів виявлення шкідливих процесів, було визначено, що існує необхідність в розробці методів виявлення шкідливих процесів в реальному часі.

4. Основне протиріччя, яке лежить в основі наукового дослідження полягає, з одного боку, що існує необхідність оперативного ідентифікування в режимі реального часу шкідливих процесів, а з іншого боку, якісне виявлення для зменшення показника хибно-позитивних спрацювань.

5. Таким чином вирішенню підлягає актуальне наукове завдання, щодо розробки методу виявлення шкідливих процесів в реальному часі на основі ідентифікації та діагностування станів логічних об’єктів інформаційної системи підприємства.

РОЗДІЛ 2 РОЗРОБКА РОЗШИРЕНОЇ МОДЕЛІ ІДЕНТИФІКАЦІЇ ТА ДІАГНОСТУВАННЯ СТАНІВ ЛОГІЧНОГО ОБ'ЄКТУ НА БАЗІ РОБОТИ TCP ПРОТОКОЛУ

2.1. Виявлення шкідливих процесів інформаційної системи на основі скінченних автоматів TCP протоколу

Технології виявлення вторгнень можна розділити на виявлення неправильного використання, виявлення аномалій і виявлення на основі специфікацій. Виявлення шкідливих процесів базується на збігу сигнатур атаки. Цей підхід може точно виявляти відомі атаки, але неефективний проти раніше невідомих атак, оскільки для таких атак немає сигнатур.

Виявлення аномалій долає обмеження виявлення неправильного використання, зосереджуючись на нормальній поведінці системи, а не на поведінці атак. На відміну від виявлення вторгнень на основі сигнатур, виявлення аномалій має перевагу у виявленні раніше невідомих атак, але ціною відносно високого рівня помилкових тривог.

У роботі [73] представили третю категорію виявлення вторгнень на основі специфікацій.

Підхід, заснований на специфікаціях, передбачає ручну розробку специфікації, яка фіксує законну поведінку системи та виявляє будь-які відхилення від неї.

Цей підхід може виявити невідомі атаки з низьким рівнем помилкових тривог.

У роботі, використаємо виявлення вторгнень на основі специфікацій у реалізації розширеного скінченного автомата TCP для системи виявлення вторгнень на основі хоста. Специфікації мережевих протоколів отримують із стандартних документів, таких як Інтернет-RFC. У реалізації абстрагуємося

від специфікацій, щоб охопити лише важливі деталі протоколів, які моделюємо. Починаючи з 1990-х років, зловмисники почали використовувати більше ніж один протокол для своїх атак. Такі атаки, як перехоплення TCP [74, 83, 84] і Targa3 [52], є хорошими прикладами таких крос-протокольних атак.

У системах передачі голосу через Інтернет-протокол (VoIP) міжпротокольні атаки зростають відповідно через залучення низки протоколів для надання послуг IP-телефонії.

Системи виявлення вторгнень, які базуються на моніторингу поведінки протоколу окремо від інших протоколів, залучених до сеансу, вже недостатньо для боротьби зі складною природою атак. Даний підхід є інноваційним з точки зору створення основи для протоколів для обміну корисною інформацією на етапі виявлення. Стани та змінні стану доступні між розширеними скінченними автоматами різних протоколів, щоб допомогти системі точно виявляти атаки. Система виявлення також є основою для всіх додатків, які базуються на TCP як протоколі транспортного рівня.

Скінченні автомати (FSM) зазвичай використовуються для опису протоколів. Критерії проектування можна легко виразити в термінах бажаних або небажаних станів протоколу та переходів між станами. Поведінку скінченних автоматів легше зрозуміти, якщо їх графічно представити у вигляді діаграми переходів станів [1, 9, 58], як показано на рисунку 2.1. Графічне представлення автомата є наочною допомогою для розробника протоколу. Це представлення допомагає у тестуванні та перевірці правильності оформлення. Контрольні стани представлені колами, а правила переходу вказані як направлені лінії.

Мережеві протоколи також моделюються за допомогою розширених скінченних автоматів (EFSM).

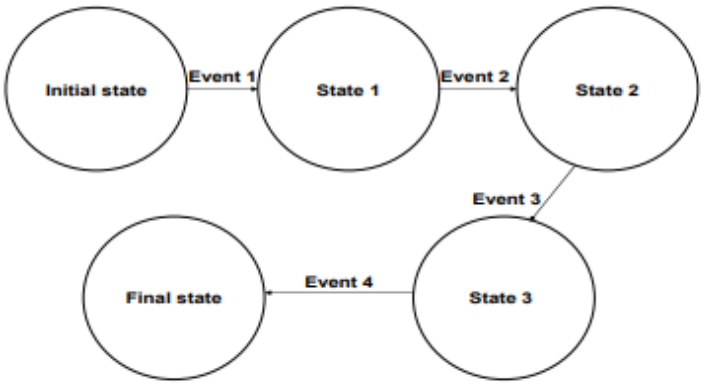

Рис. 2.1. Типова схема скінченних автоматів [15]

EFSM подібний до скінченного автомата, але з деякими відмінностями:

(a) EFSM здійснює переходи за подіями, які можуть мати аргументи, і (b) він може використовувати скінченний набір змінних стану, у яких можуть зберігатися значення.

Формально EFSM записується $(\Sigma, Q, s, f, V, D, \delta)$, де:

- Σ — алфавіт EFSM. Це алфавіт подій, тобто елементи Σ характеризуються назвою події, а також аргументами подій.
- Q — скінченний набір станів (контрольні стани) EFSM.
- $s \in Q$ — початковий стан EFSM.
- $f \in Q$ — скінченний стан.
- V — скінченний кортеж $(v_1, \ldots, v_n)$ змінних стану.
- D — скінченний кортеж $(D_1, \ldots, D_n)$, де D_i позначає область значень для змінної v_i
- $\delta : Q \times D \times \Sigma \rightarrow (Q,D)$ є співвідношенням переходу.

Мережеві атаки на транспортному рівні. Існує декілька типів DDoS атак, кожна з них може бути використана одним зловмисником або використовуючи Botnet [49, 77] (Табл. 2.1).

Протокол TCP призначений для організації надійного обміну даних між процесами в різнорідних мережевих середовищах. Не дивлячись на популярність протоколу транспортного рівня існують безліч атак для порушення безпеки. Головним недоліком протоколу TCP/IP є відсутність

інтегрованих функцій захисту, криптографічного захисту пакетів повідомлення, контролю достовірності та автентифікації учасників обміну. Тому пакети повідомлень доступні для аналізу та модифікації.

Таблиця 2.1.

Типи DDoS атак на мережевому, транспортному рівні та рівні додатків

Атака	**Опис**
SYN-флуд	Заповнює кілька TCP-портів цільової системи повідомленнями, щоб ініціювати з'єднання між системою-джерелом і цільовою системою. Цільова система відповідає повідомленням SYN-ACK на кожне отримане SYN-повідомлення. Джерело атаки ніколи не надсилає фінальних повідомлень ACK, тому з'єднання ніколи не завершується, а цільова система перевантажена незавершеними з'єднаннями.
UDP-флуд	Зловмисник надсилає UDP-пакети на кожен з 65 535 UDP-портів цільової системи. Цільова система перевантажується, обробляючи UDP-пакети та намагаючись надіслати відповідні повідомлення до системи-джерела.
UDP-фрагментація	Різновид UDP-флуду. Зловмисник використовує великі, фрагментовані підроблені пакети, щоб споживати більше пропускної здатності з меншою кількістю пакетів. Жертва витрачає ресурси процесора на "перезбірку" непотрібних пакетів.
ICMP - флуд	Пакети ICMP законно використовуються для усунення несправностей у мережі, але при використанні для DDoS-атаки ці невеликі пакети можуть перевантажити цільову систему, позбавляючи її можливості своєчасно обслуговувати дійсні мережеві запити.
Атаки Ping	Адаптація ICMP-флуду для конкретного додатка. Під час Ping-флуду сервер-жертва отримує підроблені ping-запити (echo-запити ICMP) з дуже високою швидкістю передачі пакетів і з великого діапазону IP-адрес. Сервер-жертва перевантажений великою кількістю вхідних Ping-пакетів.

Продовження таблиці 2.1.

Типи DDoS атак на мережевому, транспортному рівні та рівні додатків

Атака	**Опис**
Smurf	Просунута техніка ICMP-флуду, в якій зловмисник надсилає ICMP-запити з підробленою IP-адресою на інші системи, щоб це виглядало так, ніби цільова система надсилає повідомлення із запитом. Всі системи, які отримують ICMP Echo Request повідомлення, відповідають на підроблену IP-адресу ICMP Echo Reply повідомленнями, перевантажуючи цільову систему.
Атаки на відбиття	Зловмисники підміняють свої вихідні IP-адреси, щоб здавалося, що вони походять з мережі жертви, і починають великий SYN-флуд проти жертви з третьої сторони. Сервери або проксі-сервери жертви починають відповідати SYN-ACK, але вони спрямовуються назад на мережу жертви, перевантажуючи мережевий брандмауер.
DNS-атаки	Велика кількість підроблених пакетів DNS-запитів перевантажує цільовий сервер. Ця атака споживає мережеві ресурси та доступну пропускну здатність.
ACK і Push-флуд	Жертва отримує підроблені ACK-пакети з високою швидкістю передачі, які не належать жодному сеансу в списку підключень сервера. ACK-флуд виснажує сервер жертви, виснажуючи його системні ресурси.
Фрагментований ACK	Використовує пакети розміром 1500 байт для споживання смуги пропускання. Оскільки маршрутизатори не збирають фрагментовані пакети на рівні IP, ці пакети проходять через маршрутизатори та засоби захисту периметра мережі, споживаючи пропускну здатність.
HTTP-флуд	Зловмисник надсилає велику кількість легітимних запитів до програми. Наприклад, HTTP-флуд-атака може спричинити сотні тисяч запитів сторінок до веб-сервера, що може вичерпати всі можливості сервера з обробки даних. Під час HTTP-атаки переповнення зловмисник надсилає SYN-пакет, а цільова система відповідає SYN ACK. Зловмисник завершує тристороннє рукостискання пакетом ACK, а потім надсилає HTTP-запит GET на загальну сторінку в цільовій системі.

Продовження таблиці 2.1.

Типи DDoS атак на мережевому, транспортному рівні та рівні додатків

Атака	**Опис**
SMTP-флуд	Спамери надсилають потік трафіку, який перевантажує поштовий сервер
Атака уповільненого читання	Повільно читає HTTP-відповідь, встановлюючи розмір вікна прийому, менший за буфер відправки цільового сервера. Оскільки TCP підтримує відкриті з'єднання, навіть якщо дані не передаються, зловмисник може змусити сервер тримати велику кількість з'єднань відкритими, що в кінцевому підсумку призведе до відмови в обслуговуванні.
Sockstress	Використовує недоліки дизайну в протоколі TCP. Успішна Sockstress-атака може завдати шкоди від відмови в TCP-з'єднанні з ціллю до вичерпання пам'яті ядра.
Повільний HTTP POST	Надсилає заголовки, які сигналізують про те, скільки даних потрібно відправити, але відправляє дані дуже повільно, використовуючи тисячі HTTP POST-з'єднань для DDoS-атак на веб-сервер.
Slowloris	Надсилає часткові запити на цільовий сервер, відкриваючи з'єднання, потім надсилаючи HTTP-заголовки, доповнюючи, але ніколи не завершуючи запит.

Для того щоб реалізувати атаки, які будуть описані в роботі зловмиснику необхідно мати доступ над однією із систем, що підключені до Інтернету.

Атаки на TCP/IP можна розділити на два типи:

- пасивні;

- активні.

До пасивних відносять атаку підслуховування (англ. sniffing), яка реалізовується переведенням інтерфейсу персонального комп’ютера в режим прослуховування, при якому можливий відбір трафіку за встановленими критеріями. Сутність атаки полягає в перехопленні мережевого потоку та його аналізі. Оскільки TCP/IP трафік, як правило, не шифрується, зловмисник використовує відповідний інструментарій та перехоплює TCP/IP-пакети (наприклад telnet-сесії) та отримує з них імена користувачів та їх паролі.

Оскільки мережевий потік не змінюється, тому цей тип атаки неможливо відслідкувати не маючи доступу до системи зловмисника.

До активних відносять атаки, що модифікують та/або фільтрують вміст пакетів повідомлень для омани отримувача або порушення працездатності вузла, що приймає дані.

- IP spoofing — підміна IP. В цьому випадку зловмисник видає себе за іншу систему, якій довіряє система жертви.

Зловмиснику потрібно знаходитися в одному сегменті мережі між двома вузлами або на шляху між ними. Кожен з вузлів при встановленні зв’язку знають номери дейтаграм, які може надіслати співрозмовник, а також розмір вікна. В такому випадку зловмисник виступає в якості посередника і може не тільки прослуховувати трафік, але і активно підміняти його.

- IP Hijacking — цей метод має комбінацію атак «прослуховування» та «IP spoofing» [77]. На відмінну від попередньої атаки, в цьому методі зловмисник перехоплює увесь мережевий потік, модифікує його та фільтрує довільним чином.

При передачі даних постійно використовуються sequence number та acknowledge number. Існує можливість в «десинхронізації з’єднання», коли надіслані sequence number та acknowledge number не співпадають з очікуваними значеннями клієнта та навпаки. В такому випадку зловмисник «прослуховує» лінію, може взяти на себе функції посередника, що генерує коректні пакети для клієнта і сервера та перехоплює їх відповіді.

Метод дозволяє повністю минути такі системи захисту, як наприклад, одноразові паролі, оскільки зловмисник починає роботу вже після того, як відбудеться авторизація користувача.

- Сканування мережі — виявлення підключених до мережі комп’ютерів і визначення на них працюючих мережевих сервісів.

Сканування підключених до мережі комп'ютерів здійснюється за допомогою програми ping надсиланням ICMP-повідомлень Echo по широкомовній адресі, на яку надішлють відповідь усі комп'ютери, що підтримують обробку даних повідомлень.

Зворотне сканування реалізується надсиланням RST-сегментів, відповіді на неіснуючі запити DNS і т.д. Якщо повідомлення у відповідь не отримано, то вузол існує, підключений до мережі і працює.

Програми сканери дозволяють виявити підключені комп'ютери та відкриті порти, для реалізації атак на рівні додатків. Наприклад, за допомогою стандартної програми telnet можлива спроба встановлення з'єднання TCP за допомогою функції connect, що дозволяє вказати довільний номер порту для встановлення з'єднання [60, 72].

Сканування у режимі половинного відкриття.

Для цього на сканований порт надсилається SYN-сегмент, отримання сегмента у відповідь з бітами SYN і ACK означає, що порт відкритий, сегмент з бітом RST -порт закритий. Якщо спроба відкриття порту здійснилася, то зловмисник негайно ліквідує спробу з'єднання відправляючи сегмент з бітом RST.

Сканування за допомогою FIN сегментів. На сканований порт надсилається пакет із встановленим бітом FIN (PSH, URG), отримання сегмента у відповідь з бітами RST означає, що порт закритий, в іншому випадку порт відкритий.

- Ping flood — «затоплення» ICMP-пакетами. Реалізовується за допомогою команди ping, що призначена для оцінки якості лінії зв’язку, має параметр для «агресивного» тестування [60, 72].

Ping надсилає ICMP-пакет типу ECHO REQUEST, вказуючи у ньому час та його ідентифікатор. Отримувач відповідає на такий запит пакетом ICMP ECHO REPLY. Отримавши його ping видає швидкість проходження пакета.

При стандартному режимі роботи пакети надсилаються через деякі проміжки часу, практично не навантажуючи мережу. Але в «агресивному» режимі потік ICMP echo request/reply-пакетів може викликати навантаження невеликої лінії, позбавивши її здатності передавати корисну інформацію.

Зловмисник може підробляти зворотну адресу подібних пакетів, ускладнюючи його виявлення.

- SYN flood — «затоплення» SYN-пакетами.

Сутність полягає в надсиланні зловмисником SYN-запитів на вузол, що атакується, у кількості більшій, ніж той може обробити. Для цього зловмисник від імені неіснуючого відправника надсилає SYN-запити вузлу, що атакується, з певною частотою [60, 72]. При отриманні запиту SYN модуль TCP створює блок обслуговування і відправляє підтвердження у вигляді SYN, на яке він ніколи не отримає відповідь. Зрештою, ресурси вузла виснажуються і відбувається відмова.

Виявлення DDoS-атак на рівні протоколу TCP це складне завдання, оскільки DDoS-атаки часто стосуються великої кількості скомпрометованих пристроїв, які генерують великий обсяг трафіку. Проте перевірка прапорів TCP може надати деякі підказки щодо того, чи відбувається атака.

Одним із поширених методів виявлення DDoS є пошук ненормальних шаблонів пакетів TCP SYN. У типовому TCP-з’єднанні тристороннє рукостискання включає пакет SYN, надісланий клієнтом, пакет SYN-ACK, надісланий сервером, і пакет ACK, надісланий клієнтом для завершення рукостискання. Однак під час атаки SYN flood зловмисник надсилає велику кількість SYN-пакетів, не завершуючи рукостискання, що може призвести до перевантаження сервера [60, 72].

Щоб виявити атаку SYN flood, потрібно контролювати прапори TCP у вхідних пакетах. У звичайному TCP-з'єднанні прапор SYN встановлюється в першому пакеті рукостискання, прапор SYN-ACK встановлюється в другому пакеті, а прапор ACK встановлюється в третьому пакеті. Під час атаки SYN flood вхідні пакети матимуть встановлений прапор SYN, але не матимуть прапора ACK, що вказує на те, що рукостискання не було завершено.

Інший спосіб виявлення DDoS-атак полягає в пошуку ненормальних шаблонів пакетів TCP RST [60, 72]. У типовому TCP-з'єднанні пакет RST надсилається, щоб вказати, що з'єднання припиняється. Однак під час DDoS-атаки зловмисник може надіслати велику кількість RST-пакетів, щоб

порушити з’єднання. Відстежуючи флаги TCP у вхідних пакетах, потрібно шукати шаблони пакетів RST, які вказують на те, що відбувається атака.

Загалом виявлення DDoS-атак на рівні протоколу TCP вимагає ретельного моніторингу вхідного трафіку та глибокого розуміння нормальних шаблонів трафіку TCP.

2.2. Процес розробки специфікації роботи протоколів щодо ідентифікації та діагностування станів логічних об’єктів

На відміну від програмного забезпечення загалом, мережеві протоколи розроблені через ретельний і продуманий процес. Конструкція точно фіксується в стандартних документах. Такі документи є очевидною відправною точкою для наших специфікацій скінченного автомату.

У роботі абстрагувалися від цієї специфікації, щоб охопити лише основні деталі більшості протоколів. Таку інформацію можна легко отримати зі стандартних текстів про мережеві протоколи, а не (набагато довших) Інтернет-запитів RFC [10, 41, 64-66, 79-82]. Хоча суворе дотримання документів стандартів протоколу можливе, це може бути небажаним з двох причин. По-перше, розробка точних специфікацій вимагатиме більше зусиль, ніж те, що вимагається для більш абстрактних специфікацій. По-друге, із суворими специфікаціями завжди існує ймовірність того, що через незначні відмінності в інтерпретації деякий трафік може бути класифікований скінченним автоматом як недійсний і, отже, не оброблений належним чином. Крім того, така неправильна обробка може статися з деякими реалізаціями TCP, а не з іншими. Використання більш абстрактної специфікації, де скінченні автомати приймають надмножину того, що дозволено стандартами, забезпечує задовільний розв’язок цієї проблеми.

Специфікація наочно зображена на рис.2.1. Новий сеанс починається в стані LISTEN. Передача даних відбувається в стані з'єднання ESTABLISHED. Якщо з'єднання TCP ініціюється із зовнішнього сайту, скінченний автомат проходить стани SYN RECD і ACK WAIT, щоб досягти стану ESTABLISHED. Якщо підключення ініціюється з внутрішньої машини, стан ESTABLISHED досягається через стан SYN SENT.

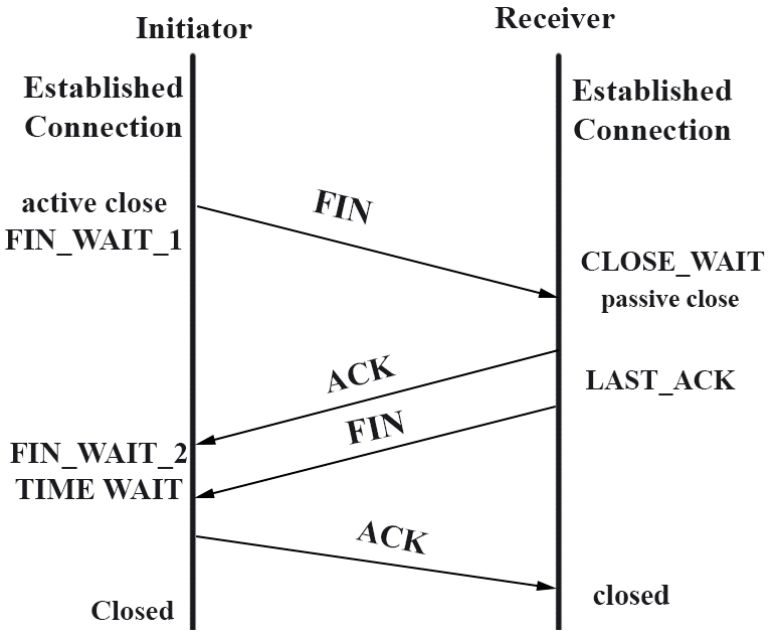

Рис. 2.1. Наочна специфікація TCP протоколу [31]

Щоб розірвати з’єднання, будь-яка сторона може надіслати TCP сегмент із встановленим бітом FIN. Якщо пакет FIN надсилається внутрішнім хостом, скінченний автомат чекає на надходження ACK FIN ззовні. Дані можуть продовжуватися, доки не буде отримано цей ACK для FIN. Також можливо, що зовнішній сайт може ініціювати закриття TCP-з’єднання. У цьому випадку ми можемо отримати FIN або FIN + ACK від зовнішнього сайту [1, 9, 58]. Цей сценарій представлено станами FIN WAIT 1, FIN WAIT 2, CLOSING, CLOSING 1 і CLOSING 2. Наш скінченний автомат характеризує події прийому та передачі окремо, і це вимагає додаткових проміжних станів, які не визначені в TCP RFC.

Якщо припинення з’єднання ініційовано зовнішнім хостом, зауважемо, що TCP RFC не мають станів CLOSE WAIT 1, CLOSE WAIT 2, LAST ACK 1 і LAST ACK 2, оскільки вони мають справу з пакетами, які спостерігаються на одному з кінців з’єднання [10, 41, 64-66, 80-83]. У цьому випадку розумно припустити, що реалізація стеку TCP не надсилатиме жодних пакетів після того, як вона отримає FIN з іншої сторони. Спостерігаємо за трафіком на проміжному вузлі (шлюзі), тому процес закриття схожий незалежно від того, який кінець ініціював закриття.

Щоб зменшити перешкоди, не показано такі класи ненормальних переходів: (a) ненормальні умови, за яких TCP-з’єднання може бути розірвано, включно з тим, коли пакет RST (з правильним порядковим номером) надсилається будь-яким боком з’єднання, а також як тайм-аути, (b) умови, коли ненормальний пакет відкидається без зміни стану, наприклад, пакети, отримані без правильних порядкових номерів (після встановлення з’єднання), і пакети з неправильними налаштуваннями прапора [1, 9, 58].

Відображення властивостей послідовності пакетів на властивості переходів скінченного автомату. Специфікації поділяють послідовності пакетів на траси, де кожна траса відповідає шляху в скінченному автоматі. Наприклад, скінченний IP-автомат, розбиває послідовність пакетів, отриманих на зовнішньому інтерфейсі шлюзу або переданих на внутрішньому інтерфейсі, на один із таких типів трас:

rx(ext, pkt), де pkt призначений для gateway

rx(ext, pkt1) tx(int, pkt2) де pkt2 — це пакет, який ретранслюється шлюзом у відповідь на отримання pkt1.

rx(ext, pkt1) тайм-аут, коли шлюз отримує пакет з адресою призначення, відмінною від адреси шлюзу, але пакет не ретранслюється шлюзом (зазвичай через помилку пакета, наприклад недійсну IP-адресу або помилку/збій шлюзу).

Це розділення вносить багато структури в те, що інакше було б довгою, неструктурованою послідовністю пакетів. Це також зменшує простір

можливих цікавих властивостей, оскільки траса, будучи досить короткою, має набагато менше властивостей, ніж надзвичайно довгі послідовності пакетів. На додаток до зменшення простору можливих властивостей, переходи в специфікаціях скінченного автомата дають конкретні підказки про те, які властивості можуть бути цікавими. Наприклад, деякі переходи представляють несподівані пакети, які зазвичай виникають через збої мережі або атаку. Подібним чином відсутність очікуваних пакетів і наступний перехід через подію тайм-ауту свідчить про збій або атаку [10, 41, 64-66, 80-83]. З цієї причини наш підхід зосереджений на властивостях, пов'язаних з окремими переходами. Ми виділяємо дві категорії властивостей:

Тип 1: чи береться певний перехід на скінченному автоматі трасуванням. (Приклад: чи перехід часу очікування береться трасуванням?)

Тип 2: значення певної змінної стану або поля пакету, коли перехід проходить трасування. (Приклад: який розмір IP-пакета під час переходу від стану INIT до PKT RCVD?)

Більш складні властивості, які включають численні переходи, наприклад, чи проходить траса певну комбінацію переходів, також можна зафіксувати в підході. Це було б досягнуто шляхом доповнення вихідної специфікації допоміжною змінною стану, яка встановлювалася б, якщо траса проходила потрібну комбінацію переходів. Тепер початкова властивість зводиться до простої властивості переходу стану – зокрема, до останнього з переходів за умови, що допоміжна змінна стану встановлена.

Вивчення статистичних властивостей. Виявлення аномалій стосується виявлення «незвичайної поведінки». З моделями скінченного автомата, зрештою, відображаємо поведінку на переходах скінченного автомата. Таким чином, незвичайна поведінка може бути виявлена, якщо підхід дізнається, як часто виконується перехід (для властивостей типу 1) [17, 20, 74, 92], або типові значення змінних стану під час переходу (для властивостей типу 2). Одним із очевидних способів представлення цієї інформації є середнє значення, наприклад, середня частота, з якою береться

перехід. Однак добре відомо, що мережеві явища мають тенденцію до високої швидкості, і, отже, середні значення не забезпечують адекватний спосіб охарактеризувати такі явища. Тому в підході зосереджуємося на охопленні розподілів, а не середніх значень. Для властивостей типу 1 зберігаємо частотні розподіли, тоді як для властивості типу 2 підтримуємо розподіл значень для змінної стану, що цікавить.

Представлення розподілів відрізняється залежно від характеру значень у розподілі. Якщо значення категоричні (наприклад, IP-адреса), розподіл просто підраховує кількість разів, коли кожне окреме значення трапляється в розподілі. Часто кількість можливих категорій може бути занадто великою, тому розподіл може представляти лише ті категорії, які зустрічаються найчастіше [17, 20, 74, 92]. Якщо значення представляють скалярну величину, таку як розмір пакета, то розподіл можна представити компактно за допомогою гістограми. Оскільки частоти являють собою скалярну величину, розподіли частот також можна представити за допомогою гістограм.

Часто цікавлять властивості, які зберігаються в підмножині «сліди». Один із способів вибору слідів, що цікавлять, заснований на нещодавності, наприклад, сліди, які спостерігалися протягом останніх t секунд. Це дозволить зосередитись на недавній поведінці, на відміну від поведінки, яка спостерігалася давно в минулому. Другий спосіб вибору трас базується на значеннях змінних стану або полів пакета.

Наприклад:

сліди, що відповідають фрагментованим пакетам,

сліди, що включають пакети від певного хоста та/або до конкретного хоста.

Статистичні властивості, які потрібно вивчити, можна зручно вказати мовою специфікацій наступним чином. Щоб проілюструвати такі специфікації, розглянемо твердження:

на всій шкалі часу частоти

(0,001, 0,02, 0,5, 10, 100, 1000)

Це твердження вказує на те, що інформація про розподіл частоти повинна бути отримана для всіх перетворень, і необхідно підтримувати шість різних розподілів, що відповідають шести різним шкалам часу [17, 20, 74, 92]. Шкала часу визначає період часу, протягом якого ми підраховуємо кількість переходів. Атаки можна виявити швидше за коротший час. Однак, оскільки мережеві явища мають тенденцію бути більш нестабільними за коротших часових масштабів, повільніші атаки, як правило, пропускаються за коротших часових масштабів [17, 20, 74, 92]. Їх можна виявити, переглянувши статистичні дані за довший часовий масштаб, але ці часові масштаби передбачають більші затримки до виявлення атаки. Використовуючи шість шкал часу в діапазоні від мілісекунд до тисяч секунд, ми поєднуємо перевагу швидкого виявлення швидкоплинних атак із перевагою відстроченого (але більш певного) виявлення повільніших атак.

Як другий приклад розглянемо таке твердження:

на всій частоті відносно (src) розмір 100 часовий масштаб (0,001, 0,02, 0,5, 10, 100, 1000)

Це твердження вказує на те, що хочемо підтримувати розподіл частот на основі кожного джерела та хосту. Оскільки кількість можливих вихідних хостів можуть бути великими, мова дозволяє використовувати оголошення розміру для обмеження вимог до зберігання.

Наприклад, у наведеному вище операторі оголошено межу 100. Якщо в будь-який час більше, ніж це багато вихідних хостів, то лише 100 найбільш активних із цих хостів будуть збережені в таблиці, а інші будуть очищеними. Поняття [17, 20, 74, 92] «найактивніший» включає старіння, тому хости, які були активними в минулому, але стали неактивними протягом тривалого періоду часу, будуть відкинуті з таблиці. (*src* — це ім’я змінної стану в специфікації скінченного автомата.)

Ключове слово значення використовується замість частоти для позначення розподілу значень, на відміну від розподілу частоти. Це дозволяє контролювати конкретні діапазони значень, які може приймати змінна стану.

Подібним чином можна обмежити моніторинг певною підмножиною переходів, перерахувавши їхні мітки замість ключового слова *all*.

Під час фази виявлення статистика вказана для навчання обчислюються знову та порівнюються зі значеннями, отриманими під час фази навчання. Якщо статистичні дані суттєво відрізняються від того, що було вивчено, виникає аномалія.

2.3. Розробка моделі розширеного автомата функціонування транспортного протоколу TCP для ідентифікації та діагностування станів логічного об’єкта інформаційної системи

Функціональні складові моделі ідентифікації та діагностування станів логічного об’єкта інформаційної системи. Відповідно до [8, 18, 51], під протоколом розуміється набір правил і форматів (семантичних і синтаксичних), що визначають процедури зв’язку логічних об’єктів при виконанні ними функцій. Саме протоколи містять вимоги, які будуть виступати вихідними даними для побудови динамічних моделей функціонування відповідних логічних об’єктів інформаційних систем. Під логічним об’єктом розуміється активний елемент рівня взаємозв'язку відкритих систем, який виконує певну підмножину його функцій. притаманних для даного рівня. Коли розглядається взаємодія функціональних компонентів певного рівня Еталонної моделі взаємодії відкритих систем (OSI), то їх називають логічними об’єктами.

Для того, щоб ефективно протистояти кібернетичним впливам на процеси функціонування інформаційної системи необхідно щоб моніторинг, ідентифікація та діагностування стану безпеки обраного логічного об’єкта інформаційної системи відбувалися в режимі реального часу та, відповідно, мають бути автоматичними [9].

Проектування моніторингу стану логічних об’єктів потребує розробки теорії даного питання та відповідної сукупності методів забезпечення її реалізації. Для вирішення проблеми виявлення шкідливих процесів на основі аналізу станів логічного об’єкта інформаційної системи необхідно розробити метод побудови динамічної моделі функціонування логічного об’єкта за відповідним протоколом, який забезпечить точність моделі функціонування обраного логічного об’єкта.

Відповідно до RFC 4949 під вторгненням (англ. Intrusion) розуміється «подія безпеки або комбінація декількох подій безпеки, що є складовою частиною інциденту безпеки, при якому зловмисник отримує або намагається отримати несанкціонований доступ до системи або системного ресурсу». Вторгнення в інформаційну систему буде мати відповідне відображення у вигляді процесу в системі, який виникає при цьому [31, 39, 53, 96]. Оскільки це не буде цільовим процесом, тому будемо називати такий процес шкідливим.

Етапи функціонування роботи TCP протоколу

Розглянемо основні етапи функціонування роботи TCP протоколу.

Роботу функціонування TCP протоколу будемо розглядати з використанням скінченних автоматів (СА) зі сторони сервера та клієнта рис. 2.2.

Нормальне функціонування протоколу відбувається в три етапи:

1. Встановлення з’єднання «Трьохстороннє рукостискання».
2. Передача даних.
3. Завершення з’єднання «Чотирьохстороннє рукостискання».

4. Основною відмінністю TCP від UDP є те, що на протокол TCP покладено додаткове завдання – забезпечити надійну доставку повідомлень, використовуючи як основу ненадійний дейтаграмний протокол IP. Встановлені в скінченних вузлах протокольні модулі TCP вирішують завдання забезпечення надійного обміну даними шляхом встановлення між собою логічного обміну. Завдяки логічному з'єднанню TCP стежить, щоб сегменти,

що передаються, не були втрачені, не були продубльовані і прийшли до одержувача в тому порядку, в якому були відправлені.

Етап 1. «Трьохстороннє рукостискання»

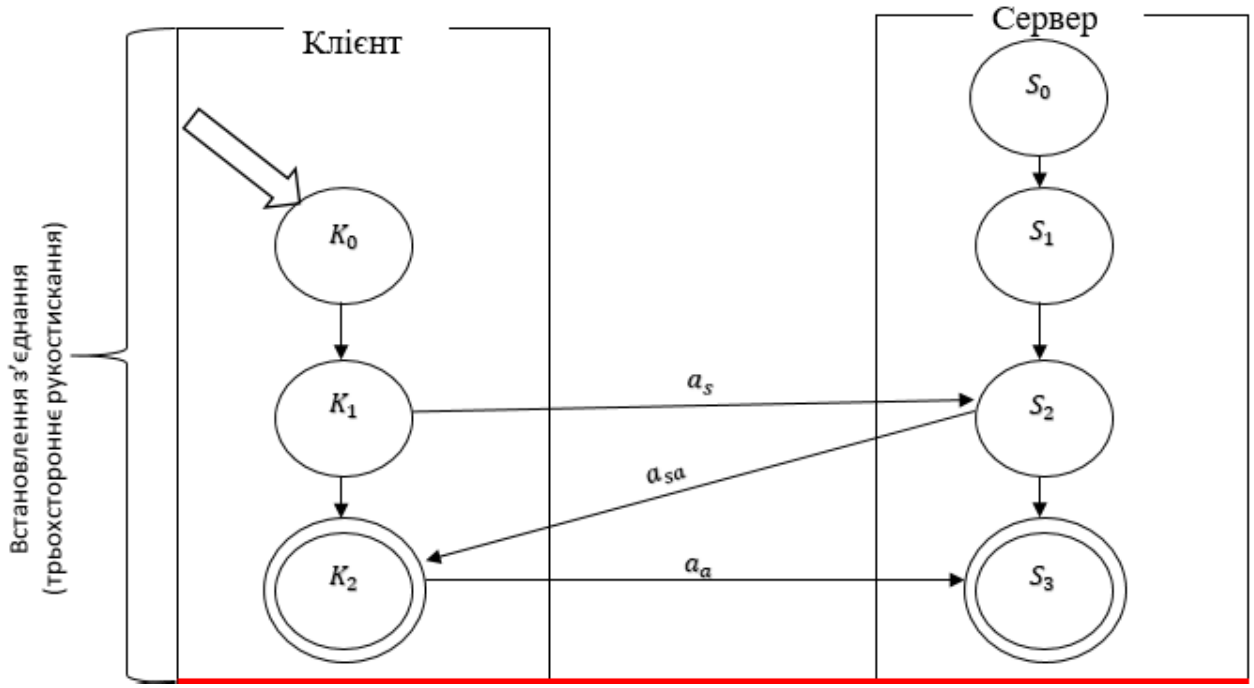

Рис. 2.2. Встановлення з'єднання «Трьохстороннє рукостискання»

Стани системи представлені в табл. 2.2.

Таблиця 2.2.

Стани системи TCP з'єднання

Стан	**Позначення**	**Опис**
CLOSED	K_0, S_0	Закрито. З'єднання не активне та не знаходиться в процесі встановлення з'єднання
LISTEN	S_1	Очікування. Сервер очікує вхідний запит.
SYN SENT	K_1	Відправлений запит на з'єднання. Початок встановлення з'єднання.
SYN RCVD	S_2	Отримання запиту з'єднання. Очікування підтвердження.
ESTABLISHED	K_2, S_3	Встановлене з'єднання. Нормальний стан передачі даних.

Щоб встановити з'єднання, одна сторона (наприклад, сервер) пасивно очікує вхідного з'єднання, виконуючи примітиви LISTEN (оголошення про бажання прийняти з'єднання) і ACCEPT (блокування дзвінка до отримання спроби з'єднання), або вказуючи конкретне джерело, або не вказуючи його. Інша сторона (наприклад, клієнт) виконує примітив CONNECT (активно намагатися встановити з'єднання), вказуючи IP-адресу та порт, з яким він хоче встановити з'єднання, максимальний розмір TCP-сегменту і, за бажанням, деякі дані користувача (наприклад, пароль). Примітив CONNECT посилає TCP-сегмент із встановленим бітом SYN і скинутим бітом ACK і чекає на відповідь.

Коли цей сегмент [1-10] прибуває в пункт призначення, TCP-сутність перевіряє, чи виконав якийсь процес примітив LISTEN, вказавши як параметр той самий порт, який міститься в полі порт одержувача. Якщо такого процесу немає, вона відповідає відправленням сегмента із встановленим бітом RST для відмови від з'єднання.

Якщо будь-який процес прослуховує якийсь порт, то вхідний TCP-сегмент передається цьому процесу. Останній може прийняти з'єднання або відмовитись від нього. Якщо процес приймає з'єднання, він посилає у відповідь підтвердження [31-58]. Послідовність TCP-сегментів, що посилаються у нормальному випадку, показана на рис. 2.3. Необхідно звернути увагу, що сегмент із встановленим бітом SYN займає 1 байт простору порядкових номерів, що дозволяє уникнути неоднозначності в їх підтвердженнях.

Якщо два хоста одночасно спробують встановити з'єднання один з одним, то послідовність подій, що відбуваються при цьому, буде відповідати зіткненню викликів. В результаті буде встановлено лише одне з'єднання, а не два, оскільки пара кінцевих точок однозначно визначає з'єднання. Тобто якщо обидві з'єднання намагаються ідентифікувати себе за допомогою пари (x, y) робиться лише один табличний запис для (x, y).

В результаті переговорного процесу модулів TCP із двох сторін з'єднання визначаються параметри з'єднання. Одні з них залишаються постійними протягом усього сеансу зв'язку, інші адаптивно змінюються. Зокрема, залежно від завантаження буфера сторони [11-30], що приймає, а також надійності роботи мережі динамічно змінюється розмір вікна відправника. Створення з'єднання означає також виділення операційною системою на кожній стороні з'єднання певних системних ресурсів: для організації буферів, таймерів, лічильників. Ці ресурси будуть закріплені за з'єднанням з моменту створення до моменту розриву.

Логічне TCP-з'єднання однозначно ідентифікується парою сокетів.

Кожен сокет одночасно може брати участь у кількох з’єднаннях. Так, якщо (IP1, n1), (IP2, n2), (IP3, n3) – сокети трьох різних додатків, де IP1, IP2, IP3 – їх IP-адреси, а n1, n2, n3 – номери їх TCP-портів, то можливе утворення таких з’єднань:

з'єднання 1 - {(IP2, n2), (IP1, n1)};

з'єднання 2 - {(IP1, n1), (IP3, n3)};

з'єднання 3 - {(IP2, n2), (IP3, n3)}.

Етап 2. Передача даних рис. 2.3.

При встановленні логічного з'єднання модулі TCP домовляються між собою про параметри процедури обміну даними. У протоколі TCP кожна сторона з'єднання посилає протилежній стороні такі параметри :

максимальний розмір сегмента, що вона готова прийняти;

максимальний обсяг даних (можливо кілька сегментів), які вона дозволяє іншій стороні передавати у свій бік. Навіть якщо та ще не отримала квитанцію на попередню кількість даних (розмір вікна);

початковий порядковий номер байта, з якого починає відлік потоку даних у межах даного з'єднання.

Рис. 2.3. Другий етап функціонування протоколу TCP

Після ініціалізації з'єднання корисне навантаження буде переміщуватися в обох напрямках TCP-з'єднання. Усі пакети обов'язково міститимуть встановлений прапор ACK. Інші прапори, такі як PSH або URG, можуть бути, а можуть і не бути встановленими [10, 41, 64-66, 80-83].

Етап 3. «Чотирьохстороннє рукостискання»

Рис. 2.4 показує, як TCP-з'єднання закривається (також називається очищенням або припиненням). Будь-яка сторона може ініціювати операцію закриття, і одночасне закриття також підтримується, але рідко. Традиційно найчастіше клієнт ініціював закриття. Однак інші сервери (наприклад, веб-сервери) ініціюють закриття після завершення запиту. Зазвичай операція закриття починається з того, що програма вказує на бажання припинити з’єднання (наприклад, за допомогою системного виклику close()). TCP, що закриває, ініціює операцію закриття, надсилаючи сегмент FIN (тобто сегмент TCP із набором бітового поля FIN). Операція повного закриття відбувається після того, як обидві сторони завершили закриття:

1. Активний завершувач (The active closer) надсилає сегмент FIN із зазначенням поточного порядкового номера, який одержувач очікує побачити (K). FIN також містить ACK для останніх даних, надісланих в іншому напрямку (позначених L).

2. Пасивний завершувач (The passive closer) відповідає значенням ACKing K + 1, щоб вказати успішне отримання FIN активного завершувача. У цей момент програма отримує сповіщення про те, що інший кінець її з’єднання

завершив закриття. Зазвичай це призводить до того, що програма починає власну операцію закриття. Тоді пасивний завершувач фактично стає іншим активним завершувачем і надсилає власний FIN. Порядковий номер дорівнює L.

3. Щоб завершити закриття, останній сегмент містить ACK для останнього FIN. Зауважимо, що якщо FIN втрачено, він повторно передається, доки не буде отримано ACK для нього.

У той час як для встановлення з’єднання потрібні три сегменти, для завершення одного – чотири [10, 41, 64-66, 80-83]. Також можливо, що з’єднання перебуває в напіврозкритому стані, хоча це не часто. Ця причина полягає в тому, що модель передачі даних TCP є двонаправленою, тобто можлива робота лише в одному з двох напрямків. Операція напівзакриття в TCP закриває лише один напрямок потоку даних. Дві операції напівзакриття разом закривають усе з’єднання.

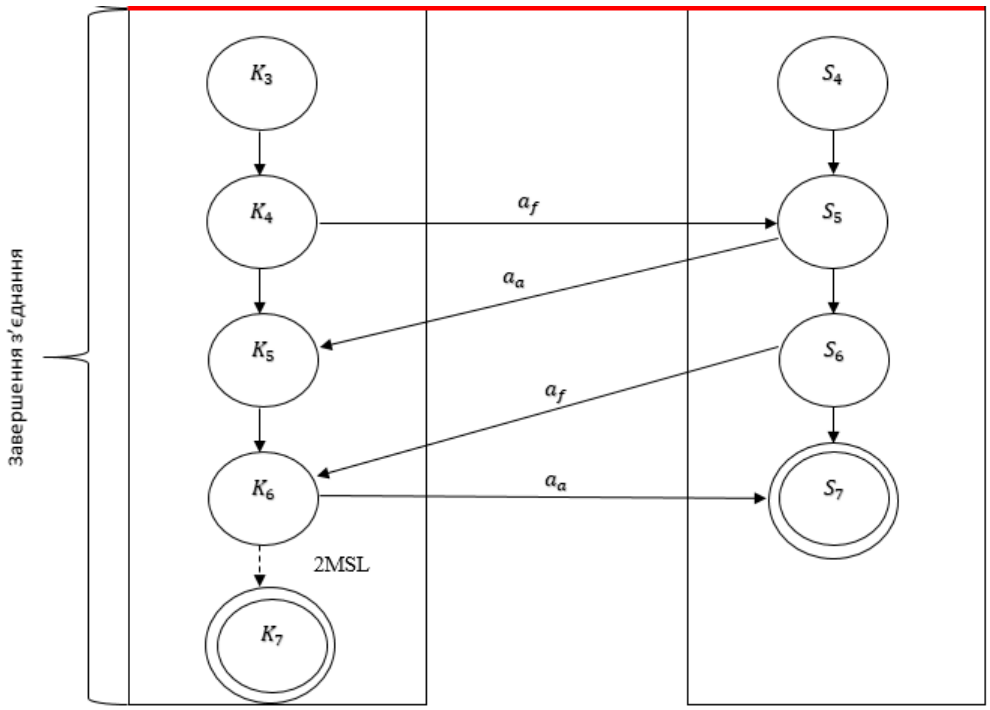

Рис. 2.4. Завершення з’єднання TCP

Правило полягає в тому, що будь-яка сторона може надіслати FIN після завершення надсилання даних. Коли TCP отримує FIN, він повинен сповістити програму, що інша сторона припинила цей напрямок потоку даних. Надсилання FIN зазвичай є результатом операції закриття програми, яка зазвичай спричиняє закриття обох напрямків.

Сім сегментів, які розглянуто [1-10, 22-30, 53-58], є базовими накладними витратами для будь-якого TCP-з'єднання, яке встановлено та очищено. (Існують більш різкі способи розірвати з'єднання TCP за допомогою спеціальних сегментів скидання). Стани системи завершення з'єднання представлені в табл. 2.3

Таблиця 2.3

Стани системи TCP при завершенні з'єднання

Стан	Позначення	Опис
ESTABLISHED	K_3, S_4	Встановлене з'єднання. Початок завершення з'єднання після передачі даних.
FIN WAIT 1	K_4	Клієнт надсилає повідомлення, що більше нічого передавати.
FIN WAIT 2	K_5	Очікування погодження завершення з'єднання від іншої сторони.
TIME WAIT	K_6	Очікування зникнення усіх пакетів.
CLOSE WAIT	S_5	Сторона ініціалізувала завершення з'єднання.
LAST ACK	S_6	Очікування зникнення усіх пакетів.
CLOSED	K_7, S_7	Закрито. З'єднання завершено.

Розглянувши основні етапи встановлення з'єднання, передачі даних та завершення з'єднання, побудуємо матрицю переходів станів СА протоколу TCP в стані спокою табл. 2.4.

Таблиця 2.4.

Матриця переходів станів скінченного автомата TCP протоколу (нормальне функціонування)

	SYN_SENT	ESTABLISHED	FIN_WAIT_1	CLOSED
SYN	SYN_SENT			
SYN-ACK		ESTABLISHED		
ACK		ESTABLISHED	FIN_WAIT_1	
FIN-ACK		ESTABLISHED	FIN_WAIT_2	CLOSED
FIN			FIN_WAIT_1	CLOSED
RST	CLOSED	CLOSED	CLOSED	CLOSED

Формалізація компонентів моделі логічного об'єкта. Формалізація компонентів моделі ідентифікації та діагностики логічного об'єкта (ЛО), побудовано в рамках деякої формальної інформаційної системи. Як математичний апарат такої теорії (числення об'єктів) доцільно використати логіку першого порядку.

Нехай α – множина функціональних, предикативних і константних символів. Кожному, функціональному символу $f \in \alpha$ можна поставити у відповідність ціле позитивне число *F(f)* таке, що якщо $n = F(f)$, то *f* називатимемо n-арним функціональним символом. Кожен предикативний символ $R \in \alpha$ можна пов'язати з позитивним цілим числом *F(R)*; якщо $n = F(R)$, то *R* називатимемо n-арним предикативним символом. Предикативні символи за своєю суттю є символами відносин. Відповідно до [18,51], під алгебраїчною системою Ф розумітимемо непорожню сукупність М об'єктів, яка є областю дії кванторів, разом з інтерпретацією основних предикативних, функціональних і константних символів з α.

Алгебраїчна система для мови α є пара Ф = <M,F>, де М непорожня множина, F – відображення з областю визначення α таке, що:

1) якщо $R \in \alpha$ – n-арний предикативний символ, то $F(R) \subseteq M^n$;

2) якщо $f \in \alpha$ – n-арний функціональний символ, то F(f): M^n —> M ;

3) якщо $c \in \alpha$ – константний символ, то F(c) $\in$ M.

Основними синтаксичними поняттями логіки 1-го порядку є: логічні зв'язки $\&, \vee, \neg, \rightarrow, =$; квантори спільності і існування $\forall$ і $\exists$, символи логічних змінних x, y, z... Всяку скінчену послідовність, елементами якої є основні символи або елементи, будемо назвати виразом.

Терми мови α утворюють найменшу множину виразів, що містять x, y, z, всі константні символи α (якщо такі є) і замкнуту відносно правила утворення: якщо $t_1,\ldots,t_n$ – терми α і якщо $f \in \alpha$ – n-місний функціональний символ, то вираз $f(t_1,\ldots,t_n)$ є термом мови α. Терм [14,22,27], що не містить змінних, називатимемо замкнутим.

Атомарна формула мови α – це вираз одного з наступних видів:

$$(t_1 = t_2),\ R(t_1,\ldots,t_n),$$

де t_1, t_2 – терми мови, а $R \in \alpha$ — довільний n-місний предикативний символ.

Формули першого порядку мови α утворюють найменшу множину виразів, що містить атомарні формули і замкнуту щодо наступного правила утворення:

1) якщо ϕ і ψ – формули, то вирази $\neg\phi$, $(\phi \& \psi)$, $(\phi \vee \psi)$, $(\phi \rightarrow \psi)$ також є формулами.

2) якщо ϕ – формула і v – змінна, то $(\exists v\phi)$ і $(\forall v\phi)$ також є формулами.

Множина $Fv(\phi)$ вільних змінних формули ϕ визначається таким чином:

1) якщо ϕ – атомарна формула, то $Fv(\phi)$ в точності множина змінних, що зустрічаються в ϕ;

2) $Fv(\neg\phi) = Fv(\phi)$;

3) $Fv(\phi \& \psi) = Fv(\phi v) = Fv(\phi) \vee Fv(\psi)$;

4) $Fv(\exists v\phi) = Fv(\forall v\phi) = Fv(\phi) - \{v\}$.

Реченням (першого порядку) мови α називатимемо формулу, що не містить вільних змінних.

Будуємо систему алгебри для мови логіки 1-го порядку α над множиною М-логічних об'єктів. Змінними позначаються ЛО [18, 51], далі вводяться предикативні і функціональні символи над множиною М.

Логічний об'єкт пропонується визначити наступним чином:

$$O: = <N_O, \{A\}, \{O\}, \{F\}>$$

де N_O – ім'я об'єкта; {A} – множина атрибутів об'єкта ($A_0,\ldots,A_n$), де A_i – і-й атрибут ЛО; {O} – множина об'єктів, які структурно входять до даного об'єкта, ($O_{NO_1}, O_{NO_2}, \ldots, O_{NO_m}$), де O_{NO_i} і-й підпорядкований об'єкт, об'єкту з ім'ям N_O; {F} – множина функцій, які виконує даний ЛО.

Атрибут ЛО визначимо як:

$$A = <N_A, S_A, V_A>$$

де N_A – ім'я атрибута, S_A – множина, на якій визначається значення атрибута, V_A – значення атрибута, тобто $\alpha \in S_A$ в даний момент часу t.

Примітивним ЛО будемо називати такий ЛО, що $O = <N_O, \{A\}, \varnothing, \{F\}>$, тобто що має порожню множину вкладених ЛО. Можна ввести структурний ЛО (або пасивний) [14, 22, 27], який визначається як $O = <N_O, \{A\}, \varnothing, \varnothing>$, тобто який має порожню множину вкладених ЛО і порожню множину функцій, а також ввести однопараметричний ЛО: $O = <N_O, A, \varnothing, \varnothing>$ для скорочення запису що позначається як $<N_O, A>$, що має тільки один атрибут. Таке визначення ЛО дозволяє розглядати множину М алгебраїчної системи як актуальну і залишатися в рамках логіки 1-го порядку при розгляді відносин між ЛО.

Почнемо введення відносин в дану модель Ф з відношення належності об'єктів (інакше кажучи, відношення структурної підпорядкованості) Rs.

Аксіома R.1: Відношення Rs – антирефлексивне: $\forall x(\neg Rs(x,x))$. Це витікає із змістовного поняття структурної вкладеності (сам предмет не містить самого себе у якості структурного елемента).

Аксіома R.2: Відношення Rs – транзитивне:

$\forall x \forall y \forall z(Rs(x,y) \& Rs(y,z) \to Rs(x,z))$.

Аксіома R.3: Відношення Rs – несиметричне:

$\forall x \forall y(Rs(x,y) \to \neg Rs(y,x))$.

Розглядаючи ці властивості як аксіоми в численні предикатів, покажемо, що вони є загальнозначущими формулами в мові α (або тавтологіями).

Для аксіоми R.1: $\forall x(\neg Rs(x,x))$. Скористаємося геометричною інтерпретацією.

Для самого себе об'єкт x не забезпечує істинності Rs. Тому для всіх x аксіома Rs(1) є тавтологією.

Для аксіоми R.2 розглянемо наступні варіанти структурних відносин:

Варіант 1. Об'єкти x, y і z незалежні один від одного. Формулу $(Rs(x,y) \& Rs(y,z) \to Rs(x,z))$ розкриваємо як диз'юнкцію відповідно до правил числення предикатів, тоді отримуємо: $(\neg(Rs(x,y) \& Rs(y,z)) \vee Rs(x,z))$. При даних відносинах об'єктів маємо істинносні оцінки $Rs(x,y)=f$, $Rs(y,z)=f$, $Rs(x,z)=f$ отже $\neg(f \& f) \vee f = \neg(f) \vee f = t \vee f = t$.

Варіант 2. Об'єкт y включає x, а z – незалежний. Тоді $Rs(x,y)=t$, $Rs(y,z)=f$, $Rs(x,z)=f$, і отже $\neg(t \& f) \vee f = \neg(f) \vee f = t \vee f = t$. З точністю до перейменування змінних, аналогічні результати виходять для випадків включення одного об'єкта в іншій і незалежності третього.

Варіант 3. Два об'єкти x і y входять в об'єкт z. В цьому випадку $Rs(x,y)=f$ $Rs(y,z)=t$ $Rs(x,z)=t$ $\neg(f\,\&\,t)\vee t=\neg(f)\vee t=t\vee t=t$. З точністю до перейменування змінних, отримаємо докази істинності формул, аналогічно варіанту 2.

Варіант 4. Об'єкт x входить в об'єкт *y*, а *y* входить в *z*. Маємо $Rs(x,y)=t, Rs(y,z)=t, Rs(x,z)=t$, тоді $\neg(t\,\&\,t)\vee t=\neg(t)\vee t=f\vee t=t$. Таким чином, при всіх інтерпретаціях *x, y, z* дана пропозиція є тавтологією, що і потрібно було довести.

Для аксіоми R.3: $\forall x\forall y(Rs(x,y))\rightarrow\neg Rs(x,y)$.

Розглядаємо варіанти структурних відносин двох об'єктів x і y.

Варіант 1. Об'єкти x і y структурно незалежні. Тоді $Rs(x,y)=f$, $Rs(y,x)=f$, і $\neg Rs(x,y)\vee Rs(y,x)=\neg f\vee\neg f=t\vee t=t$

Варіант 2. Один об'єкт включає інший, наприклад x включає y, тоді $Rs(x,y)=f$, $Rs(y,x)=t$ і $\neg Rs(x,y)\vee Rs(y,x)=\neg f\vee\neg t=t\vee f=t$.

Варіант, коли y включає x, розглядається аналогічно, шляхом перейменування об'єктів.

В результаті доведено значимість аксіом R.1 – R.3 в мові α.

Як засіб отримання кількісної оцінки структурних відносин об'єктів можна ввести функцію структурної потужності об'єкта, позначивши її функціональним символом Pw. Для цієї функції пропонується наступна система аксіом:

Аксіома Pw.1: Pw(x) = 1, якщо x – примітивний ЛО;

Аксіома Pw.2: $Pw(x) = Pw(x_1)+ Pw(x_2)$, якщо $x = <N_O,\{A\},(x_1, x_2),\{F\}>$ або, в загальному випадку, $x \in \{<N_O> \{A\}, (x_1, x_2), \{F\}>\}$, де {A} і {F} можуть бути і порожніми множинами. Знак + розуміється тут як символ додавання натуральних чисел.

На підставі аксіом Pw.1 і Pw.2 покажемо, що функція Pw є адитивною, тобто справедлива **теорема 2.1**:

$$Pw(x)=\sum_{i=1}^{n} Pw(x_i)$$

де $x \in \{<N_O>, \{A\}, (x_1, x_{2,\ldots,} x_n), \{F\}>\}$.

Для доказу скористаємося методом індукції.

Для i = 1 маємо: $Pw(x) = Pw(x_1)$, у випадку, якщо x_1 примітивний ЛО, то $Pw(x_1) = 1$, і отже, $Pw(x) = 1$. Якщо ж x_1 не примітивний ЛО, то повертаємося до початкової посилки.

Для i = 2 теорема 2.1 справедлива внаслідок аксіоми Pw(2). Покажемо, що якщо теорема вірна при i = n, то вона вірна і при i = n + 1. В цьому випадку $Pw(x) = Pw(<N_O>, \{A\}, (x_1, x_2, \ldots, x_n, x_{n+1}), \{F\}>)$.

Оскільки дія функції Pw не зачіпає такі складові об'єкта, як N_O, $\{A\}$, $\{F\}$, то можемо ввести новий об'єкт x^*_n, що включає x_n і x_{n+1}:

$$x^*_n = <N_O, \{A\}, (x_n, x_{n+1}), \{F\}>.$$

Тоді

$$Pw(x) = Pw(<N_O, \{A\}, (x_1, x_2, \ldots, x^*_n), \{F\}>) =$$

$$=\sum_{i=1}^{n} Pw(x_i) = Pw(x_1) + Pw(x_2) + \ldots + Pw(x^*_n) =$$

$$= Pw(x_1) + Pw(x_2) + \ldots + Pw(<N_O, \{A\}, (x_n, x_{n+1}), \{F\}>) =$$

$$= Pw(x_1) + Pw(x_2) + \ldots + Pw(x_n) + Pw(x_{n+1})$$

по аксіомі Pw.2. Отже, теорема 2.1 вірна і для довільного натурального числа n.

Далі покажемо, що справедлива **теорема 2.2**:

Функція Pw(x) повертає кількість примітивних об'єктів, що містяться в логічному об'єкті X.

Скористаємося індукцією по глибині індекса вкладення об'єктів l. Нехай l = 1, тоді $\sum_{i=1}^{n} Pw(x_i) = n$, якщо всі x_i що входять в X є примітивними ЛО.

Покажемо, що якщо теорема справедлива при глибині індекса вкладення k, то вона справедлива і при k+1. При глибині вкладення, яка дорівнює k маємо:

$$\sum_{i=1}^{n} Pw_k(x) = \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} Pw\left(xl_1l_2\dots l_k\right) = \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} l_k\,,$$

коли всі $xl_1l_2\dots l_k$ є примітивними ЛО і $l_k = Pw\left(xl_1l_2\dots l_k\right)$. При глибині вкладення k+1 маємо:

$$Pw_{k+1}(x) = \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k}\sum_{l_{k+1}=1}^{nl_{k+1}} Pw\left(xl_1l_2\dots l_kl_{k+1}\right).$$

Відповідно до теореми 2.1 можемо цю суму розкрити як

$$Pw_{k+1}(x) = \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k}\begin{pmatrix} Pw\left(xl_1l_2\dots l_{k-1}l_{k1}\right) + Pw\left(xl_1l_2\dots l_{k-1}l_{k2}\right) + \dots + \\ +Pw\left(xl_1l_2\dots l_{k-1}l_kl_{k+1}\right) \end{pmatrix}$$

Розкриваючи дужки, отримаємо:

$$\begin{aligned} Pw_{k+1}(x) = &\sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} Pw\left(xl_1l_2\dots l_{k-1}l_{k1}\right) + \\ &+\sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} Pw\left(xl_1l_2\dots l_{k-1}l_{k2}\right) + \dots + \\ &+\sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} Pw\left(xl_1l_2\dots l_{k-1}l_kl_{k+1}\right) \end{aligned}$$

Для кожного з цих доданків, виходячи із заданої передумови теореми, отримуємо:

$$Pw_{k+1}(x) = \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} l_{k1} + \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} l_{k2} + \sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k} l_{kl_{k+1}} =$$
$$\sum_{l_1=1}^{nl_1}\sum_{l_2=1}^{nl_2}\dots\sum_{l_k=1}^{nl_k}\sum_{l_{k+1}=1}^{nl_{k+1}} l_{k+1}$$

що і потрібно було довести.

Для забезпечення замкнутості числення об’єктів на множині M необхідно поставити умову, щоб результат дії функціонального символу Pw також був об’єктом. Тому замість безпосереднього використання натурального числа вважатимемо, що $Pw(x) = <N_O, A>$ –однопараметричний об’єкт, де

$$A = < N_A, \{N\}, a>,$$

де {N} – множина натуральних чисел; $a \in \{N\}$. У свою чергу, оскільки мова α допускає побудову рекурсивних термів, формально припустимим є терм Pw(Pw(x)). Тому $Pw(Pw(x))= Pw(< N_O, A >)$ покладемо рівним x_0, де $x_0 = <\varnothing, \varnothing,\varnothing,\varnothing>$, тобто порожній об’єкт [14, 22, 27]. Порожній об’єкт x_0 необхідно замкнути сам на себе і вважати, що $Pw(x_0) = x_0$, тоді рекурсія виду Pw(Pw... Pw(x)...) завжди дасть x_0. З міркувань суворості доведеться доповнити систему аксіом для відношення Rs:

Аксіома R.1.1: $\forall x\left(\neg Rs\left(x_0, x\right)\right) \forall x\left(\neg Rs\left(x_0, x\right)\right)$;

Аксіома R.1.2: $\forall x\left(\neg Rs\left(x, x_0\right)\right) \forall x\left(\neg Rs\left(x, x_0\right)\right)$.

Обґрунтування того, що аксіоми R.1.1 і R.1.2 є тавтологіями, абсолютно аналогічно обґрунтуванню аксіоми R.1.

Використання числення предикатів з рівністю примушує нас ввести поняття рівності об’єктів. Насправді, трактувати його необхідно як

еквівалентність об’єктів. Вважатимемо, що два об’єкти рівні (еквівалентні один одному), якщо

$$x = <NA_x, \{A\}_x, \{O\}_x, \{F\}_x>;$$

$$y = <NA_y, \{A\}_y, \{O\}_y, \{F\}_y>;$$

$$\{A\}_x = \{A\}_y, \{O\}_x = \{O\}_y, \{F\}_x = \{F\}_y.$$

Імена об’єктів при цьому можуть бути різними.

Очевидно, що далі потрібно визначити відношення рівності для множин атрибутів, вкладених об’єктів і моделей функціонування. Дві множини атрибутів $\{A\}_x$ і $\{A\}_y$ вважатимемо рівними (еквівалентними), якщо для всіх $A_i g \in g\{A\}_x$ і $A_j g \in g\{A\}_y$ спостерігається попарна рівність при однаковому значенні індекса i, тобто $A_{ix} = A_{iy}$, де $i = 1, 2 \ldots n$, при цьому n – потужність множин $\{A\}_x$ і $\{A\}_y$.

Два атрибути A_φ і A_ψ назвемо рівними (еквівалентними), якщо $NA_\varphi = NA_\psi$, $SA_\varphi = SA_\psi$, $VA_\varphi = VA_\psi$.

Рівність списків вкладених об’єктів $\{O\}_x$ і $\{O\}_y$ визначається аналогічно рівності множин атрибутів, тобто $\{O\}_x = \{O\}_y$, якщо потужності множин $\{O\}_x$ і $\{O\}_y$ рівні і кожен $O_{ix} = O_{iy}$. Очевидно, що це визначення рекурсивне, оскільки об’єкти O_{ix} і O_{iy} можуть мати свої вкладені об’єкти.

Необхідно також ввести поняття потужності об’єкта Pp, тобто функції, що показує кількість вкладених об’єктів.

Введемо наступні аксіоми:

Аксіома Pp.1: $Pp(x)=\varnothing$, якщо x – примітивний ЛО;

Аксіома Pp.2: $Pp(x)= k$, якщо $x = <NO, \{A\}, (x_1, x_2,..., x_k), \{F\}>$, де $\{A\}$ і $\{F\}$ можуть бути і порожніми множинами.

Можна ввести Pp(x) і конструктивно, тобто використовувати продукційні правила для обчислення k, оскільки це число в явному вигляді може бути і невідоме. За аналогією з функцією Pw вважатимемо, що $Pp(x) = < N_O, A >$ – однопараметричний об’єкт, де $A – < N_A, \{N\}, a >$, де $a \in \{N\}$. $Pp(x_0)$ покладемо рівним x_0.

Розширення алгебраїчної системи Ф можна здійснити шляхом введення нових відношень над ЛО (відповідно нових предикативних символів в мову α), що дозволяє заздалегідь не обмежувати спільність системи Ф.

Оскільки найбільш поширені відношення мають типові властивості, то можна запропонувати сукупність аксіом, визначених апріорно і відомих формальній системі, щоб згодом задавати їх просто посиланнями. Запишемо їх як

Аксіома А.1: $\forall x \forall y (xRy \rightarrow yRx)$;

Аксіома А.2: $\forall x \forall y (xRy \rightarrow \neg (yRx))$;

Аксіома А.3: $\forall x (xRx)$;

Аксіома А.4: $\forall x \neg (xRx)$;

Аксіома А.5: $\forall x \forall y \forall z ((xRy) \& (yRz) \rightarrow xRz)$.

Тут аксіома А.1 виражає властивість симетричності, аксіома А.2 – асиметричності, аксіома А.3 – рефлексивності, аксіома А.4 – антирефлексивності і аксіома А.5 – транзитивності.

Задаючи таким чином нові відношення (предикативні символи), ми можемо будувати різні, довільні до деякої міри аксіоматизовані системи Ф в мові числення предикатів.

Відношення рівності (еквівалентності) об’єктів, очевидно, вимагає детальнішого розгляду внаслідок складної структури ЛО. Введемо наступні визначення. Два логічні об’єкти *x* і *y* назвемо одновидовими, якщо $N_{Ox} \neq N_{Oy}$, $\{A\}_x = \{A\}_y$, $\{O\}_x = \{O\}_y$, $\{F\}_x = \{F\}_y$, тобто вони еквівалентні з точністю до власних імен об’єктів. Відношення одновидової еквівалентності є симетричним, антирефлексивним і транзитивним (задовольняє аксіомам А.1, А.4, А.5).

Введемо еквівалентність з точністю до імен атрибутів.

Два ЛО x і y називаються об’єктами одного класу, якщо $N_{Ox} \neq N_{Oy}$, $\{O\}_x = \{O\}_y$, $\{F\}_x = \{F\}_y$ і для всіх i $NA_{xi} = NA_{yi}$, $SA_{xi} = SA_{yi}$, де i=1,2...,k, де k –

число атрибутів в множині $\{A\}_x$ і $\{A\}_y$ (потужність $\{A\}_x = \{A\}_y$. Відношення класової еквівалентності також є симетричним, антирефлексивним і транзитивним.

Якщо два об'єкти знаходяться у відношенні одновидової еквівалентності, то вони обов'язково знаходяться і у відношенні класової еквівалентності, тобто справедлива наступна теорема.

Теорема 2.3. $\forall x \forall y \forall z \left(\left(xR_0 y \right) \rightarrow \left(xR_k y \right) \right)$.

Дана теорема безпосередньо виходить з визначень одновидової та класової еквівалентності, оскільки одновидова еквівалентність є окремим випадком класової еквівалентності при $VA_{xi} = VA_{yi}$.

Зворотнє твердження, природньо, є невірним з тих же причин. Тут через R_0 позначаємо одновидову еквівалентність, а через R_k – класову еквівалентність.

Два ЛО x і y називаються структурно еквівалентними, якщо $N_{Ox} \neq N_{Oy}$, $\{O\}_x = \{O\}_y$, $\{F\}_x = \{F\}_y$, $\{A\}_x \cong \{A\}_y$, де знак $\cong$ позначає структурну еквівалентність. СЕ для $\{A\}$ означає, що потужності множин $\{A\}_x$ і $\{A\}_y$ однакові і $S_{Axi} = S_{Ayi}$ при i = 1,2,...,k, де k – потужність $\{A\}_x$ і $\{A\}_y$ відповідно. Таким чином, співпадають тільки визначення атрибутів, а імена і значення їх можуть бути різні. Для множин $\{O\}_x$ і $\{O\}_y$ структурна еквівалентність означає, що первинні потужності Pp(x) і Pp(y) рівні і $O_{xi} = O_{yi}$, де i = 1,2...,m, де m = Pp(x) = Pp(y).

Відношення структурної еквівалентності є симетричним, антирефлексивним і транзитивним.

Функція визначення атрибута.

Оскільки при аналізі процесів функціонування буде потрібно порівняння різних характеристик ЛО, введемо функцію визначення атрибута I(p, q), де p – ім'я ЛО, q – ім'я атрибута.

$$I(p,q)=\begin{cases} x_0, \text{ якщо } \forall N_A\left(N_A \in p\right)\wedge\left(N_A \neq q\right) \\ O=<p_N_A, Aq> - \text{ у протилежному випадку} \end{cases}$$

Це означає, що у разі відсутності в множині {A} атрибута з ім'ям q, функція приймає значення порожнього логічного об'єкта x_0. Виходить, що функція I(p, q) приймає значення на множині $x_0 \cup \{O_{op}\}$, де $\{O_{op}\}^*$ – множина усіх можливих однопараметричних ЛО [1-14, 18-22, 51]. Необхідно зазначити апріорно, що взяття функції I від однопараметричного об'єкта дає той же самий однопараметричний об'єкт: I(p,q)=p, якщо p – однопараметричний ЛО.

Це дозволяє надалі ввести необхідні математичні операції над однопараметричними ЛО (ОПЛО). Почнемо цей розгляд з операції арифметичного підсумовування.

Нехай Σ(X,Y) = Z, де X = <p, A>, Y= <q, B> є ОПЛО з іменами p і q, а також атрибутами A і B, відповідно, при цьому $A = <N_A, S_A, V_A>$, $B = <N_B, S_B, V_B>$.

Введемо наступні аксіоми:

Аксіома B.1: $\forall x\left(\left(x=x_0\right)\rightarrow\left(\sum(x,y)=x_0\right)\right)$;

Аксіома B.2: $\forall x\left(\left(x=x_0\right)\rightarrow\left(\sum(y,x)=x_0\right)\right)$.

В результаті логічного виводу отримаємо, що $\Sigma(x_0, x_0) = x_0$. Об'єкт x_0 служить ознакою виникнення помилки.

Аксіома B.3: $\forall x\forall y\left(\neg\left(S_A=S_B\right)\rightarrow\left(\sum(X,Y)=x_0\right)\right)$;

Аксіома B.4: $\forall x\forall y\forall z\left(\left(S_A=S_B\right)\&\left(V_C\in S_A\right)\rightarrow\left(\sum(X,Y)=Z\right)\right)$;

де Z = <p-q, ΣA,B>,

де ΣA, B=C=$<N_C,S_C,V_C>$, $N_C = N_A - N_B$, $S_C = S_A = S_B$, $V_C = \Sigma V_A, V_B$;

Аксіома B.5: $\forall x\forall y\forall z\left(\left(S_A=S_B\right)\&\neg\left(V_C\in S_A\right)\rightarrow\left(\sum(X,Y)=x_0\right)\right)$.

Це випадок, коли V_C лежить за межами множини S_A.

$V_C = \Sigma V_A$, V_B розуміється як арифметична сума, яка може бути побудована конструктивно для відповідних видів чисел.

Операцію можна зробити і n-арною, використовуючи рекурсивне визначення:

$$\sum_{i=1}^{n} x_i = \Sigma\left(\sum_{i=1}^{n-1} x_i\right), x_n.$$

Легко відмітити, що для введеного таким чином додавання ОПІО не виконується комутативний закон: $\Sigma(X,Y) \neq \Sigma(Y,X)$, оскільки результатом лівої суми є $Z = <p-q, \Sigma A, B>$, а результатом правої суми $Z' = <q-p, \Sigma B, A>$.

Та зате для додавання ОПІО виконується асоціативний закон

$$\Sigma(X,Y,Z)=\Sigma(X,\Sigma(Y,Z))=\Sigma(\Sigma(X,Y),Z).$$

З рекурсивного визначення суми отримуємо: $\Sigma(X,Y,Z)= \Sigma(X,\Sigma(Y,Z))$. Покажемо, що це в точності дорівнює $\Sigma(\Sigma(X,Y),Z)$

Нехай $X=<p, A>$, $Y=<q, B>$, $Z=<l, C>$, тоді $\Sigma(X, \Sigma(Y, Z))$ набуває вигляду $\Sigma(X, \Sigma(Y, Z)) = \Sigma(X, <q-l, \Sigma B, C>) = \Sigma(X, <q-l, <N_B - N_C, S_B, \Sigma V_B, V_C>>)$.

У свою чергу

$\Sigma(\Sigma(X,Y),Z) = \Sigma<p-q, \Sigma A, B>$, $Z = \Sigma<p\text{-}q, N_A - N_B, S_A, \Sigma V_A, V_B>>$,

$Z = <p\text{-}q\text{-}l, N_A - N_B - N_C, S_A, \Sigma(\Sigma V_A, V_B), V_C>$

Оскільки V_A, V_B, V_C – це числа, то для них асоціативний закон виконується і $\Sigma V_A, \Sigma V_B, V_C = \Sigma(\Sigma V_A, V_B), V_C$. Таким чином, доведена справедливість асоціативного закону для ОПІО.

Покажемо справедливість закону нульової множини для складання ОПІО.

Теорема 2.4. Для всіх ОПІО виконується сувора рівність (еквівалентність):

$$\Sigma\left(\left(\sum_{i=1}^{n-1} x_i\right), x_0\right) = x_0$$

Для доказу скористаємося індукцією по довжині суми. При $n = 2$ маємо $\Sigma(x), x_0 = x_0$ згідно Аксіоми В.2. Припустимо, що наше твердження справедливе при довільному n, і покажемо, що з цього виходить його істинність при $n + 1$:

$$\Sigma\left(\left(\sum_{i=1}^{n-1} x_i\right), x_0\right) = \Sigma\left(\Sigma\left(\left(\sum_{i=1}^{n-1} x_i\right), x_{n+1}\right), x_0\right)$$

Розглянемо два можливі випадки:

а) сума $\sum_{i=1}^{n-1} x_i$ дає деякий ОПІО, який позначимо як М. Тоді отримуємо:

$$\Sigma\left(\Sigma\left(M, x_{n+1}\right), x_0\right) = \Sigma\left(M, \Sigma\left(x_{n+1}, x_0\right)\right),$$

$$\Sigma\left(\Sigma\left(\left(\sum_{i=1}^{n-1} x_i\right), x_{n+1}\right), x_0\right) = \Sigma\left(\Sigma\left(M, x_{n+1}\right), x_0\right)$$

згідно з асоціативним законом.

За аксіомою В.2 отримуємо $\Sigma(x_{n+1}, x_0)$ і тоді $\Sigma\left(M, x_0\right) = \Sigma\left(\sum_{i=1}^{n} x_i\right), x_0 = x_0$ за посилкою індукції;

б) сума $\sum_{i=1}^{n-1} x_i$ дає в результаті x_0. Тоді

$$\Sigma\left(\left(\sum_{i=1}^{n+1} x_i\right), x_0\right) = \Sigma\left(\Sigma\left(\left(\sum_{i=1}^{n} x_i\right), x_{n+1}\right), x_0\right) = \Sigma\left(\Sigma\left(x_0, x_{n+1}\right), x_0\right)$$

За аксіомою В.1 отримуємо:

$$\Sigma\left(\Sigma\left(x_0, x_{n+1}\right)\right), x_0 = \Sigma\left(x_0, x_0\right) = x_0$$

Теорема 2.5. Для всіх ОПІО виконується сувора рівність (еквівалентність): $\Sigma\left(x_1, \ldots, x_k, x_0, x_{k+2}, \ldots, x_n\right) = x_0$ при довільних k і n, що є натуральними числами.

Для доказу цього твердження перетворюватимемо n-арну суму відповідно до рекурсивного визначення:

$$\Sigma\left(x_1,\ldots,x_k,x_0,x_{k+2},\ldots,x_n\right)=\Sigma\left(\left(\sum_{i=1}^{n-1}x_i\right),x_n\right)=\Sigma\left(\Sigma\left(\sum_{i=1}^{n-2}x_i\right),x_{n-1}\right)=$$

$$=\Sigma\left(\Sigma\left(\ldots\left(\sum_{i=1}^{k}\left(x_k,x_0\right),x_{k+2}\right),\ldots\right),x_n\right)$$

Оскільки тут змінюються два індекси k і n, то необхідно використовувати подвійну індукцію. Оскільки k зобов'язаний знаходитися в межах 0 < k < n, то скористаємося зворотною індукцією по k:

а) k = 1, n = 3:

$$\Sigma x_0,x_2,x_3=\Sigma\left(\Sigma\left(x_0,x_2\right),x_3\right)=\Sigma x_0,x_3=x_0$$

Це витікає з асоціативного закону і Аксіоми В.1.

Слід довести, що теорема справедлива при k = 1 і довільному n. Тому робимо наступний крок;

б) k = 1, n = m, вважаючи, що твердження істинне при n = m, доведемо, що воно істинне і при n = m + 1:

$$\Sigma x_0,x_2,x_3,\ldots,x_{m+1}=\Sigma\left(\Sigma\left(x_0,\ldots,x_m\right),x_{m+1}\right)=\Sigma x_0,x_{m+1}=x_0\,.$$

Далі переходимо до зворотної індукції по k.

в) перевіряємо справедливість теореми при k = n – 1:

$$\Sigma\left(x_1,\ldots,x_0,x_n\right)=\Sigma\left(\Sigma\left(x_1,\ldots,x_0\right),x_n\right)=\Sigma x_0,x_n=x_0$$

відповідно до закону нульової множини.

г) покажемо, що якщо теорема вірна для довільного k, 1 < k < n, то вона вірна і для k – 1:

$$\Sigma\left(x_1,\ldots,x_k,x_0,x_{k+2},x_n\right)=\underbrace{\Sigma(\Sigma(\ldots\Sigma}_{n-k\ \text{раз}}\left(\sum_{i=1}^{k-1}x_i,x_k\right),x_0),x_{k+2}),\ldots),x_n)=x_0 \tag{2.1}$$

$$\Sigma\left(x_1,\ldots,x_{k-1},x_0,x_{k+1},\ldots,x_n\right)=\underbrace{\Sigma(\Sigma(\ldots\Sigma}_{n-k\ \text{раз}}\left(\sum_{i=1}^{k-1}x_i,x_0\right),x_{k+1}),x_{k+2}),\ldots),x_n)=$$

$$\underbrace{\Sigma(\Sigma(\ldots\Sigma}_{n-k\ \text{раз}}\left(x_0,x_{k+1}\right),x_{k+2}),\ldots),x_n)=\underbrace{\Sigma(\Sigma(\ldots\Sigma}_{n-k-1\ \text{раз}}\left(x_0,x_{k+2}\right),x_{k+3}),\ldots),x_n) \tag{2.2}$$

Суму (2.1) приведемо до вигляду (2.2), використовуючи закон нульової множини:

$$\underbrace{\Sigma(\Sigma(...\Sigma}_{n-k\ раз}\left(\sum_{i=1}^{k-1} x_i, x_k\right), x_0), x_{k+2}),...), x_n) =$$

$$= \underbrace{\Sigma(\Sigma(...\Sigma}_{n-k\ раз}\left(\sum_{i=1}^{k-1} x_i, x_0\right), x_0), x_{k+2}),...), x_n) =$$

$$= \underbrace{\Sigma(\Sigma(...\Sigma}_{n-k-1\ раз}\left(x_0, x_{k+2}\right), x_{k+3}),...), x_n)$$

Видно, що суми (2.1) і (2.2) набувають ідентичного вигляду, отже, твердження z справедливе, і, отже, істинна теорема в цілому.

Слід зазначити, що комутативний закон для додавання ОПЛО виконується для відношення структурної еквівалентності:

$$(\Sigma(X,Y) R_{ST} (\Sigma(Y,X)).$$

Опис функціонування логічного об’єкта інформаційної системи. Множина функцій логічного об’єкта. Введення поняття “множина функцій” дозволяє природним чином розділити ЛО на два класи: активні ($\{F\} \neq \varnothing$) і пасивні ($\{F\} = \varnothing$).

Активними ЛО (АЛО) називатимемо такі ЛО, у яких множина функцій не порожня, тобто вони володіють власною поведінкою і можуть виконувати деякі активні дії.

Пасивними ЛО (ПЛО) назвемо ЛО, що не володіють власною поведінкою, пасивно беруть участь в реалізації деяких дій.

Припустимо, що взаємодія АЛО здійснюється через прийом і передачу ПЛО, при цьому в множині функцій АЛО можуть породжуватися необхідні ПЛО і передаватися іншому ЛО. Цей інформаційний обмін можна трактувати як передачу повідомлень, обмін сигналами, зміну вхідних сигналів і тому

подібне, що дозволяє розглядати систему “джерело – приймач” в більш широкому сенсі ніж парадигма “клієнт – сервер”.

Традиційним засобом опису функціонування об’єктів в об’єктно-орієнтованих методах і системах є модель скінченого автомата. Як більш універсальний засіб визначення функціонування ЛО пропонується використовувати апарат канонічних числень Е. Поста [27].

Канонічним численням (КЧ) називатимемо четвірку вигляду

$$(A,a,P,G), \tag{2.3}$$

де A – алфавіт числення, a – список аксіом КЧ, P – алфавіт змінних, G – список правил виводу, кожне з яких має вигляд.2

$$\begin{array}{c} G_{11}p_{11}, G_{12}p_{12}, \ldots, G_{1n_1}p_{1n_1}G_{1n_{1+1}}; \\ G_{21}p_{21}, G_{22}p_{22}, \ldots, G_{2n_2}p_{2n_2}G_{2n_{2+1}}; \\ \underline{G_{m1}p_{m1}, G_{m2}p_{m2}, \ldots, G_{mn_m}p_{mn_m}} \\ G_1p_1, G_2p_2, \ldots, G_np_nG_{n+1} \end{array} \tag{2.4}$$

де $G_{ij}(i=1,\ldots,m, j=1,\ldots,n)$ і $G(k=1,\ldots,n+1)$ – деякі конкретні слова в алфавіті A, у тому числі і порожнє слово. Далі скористаємося визначеннями, введеними в [14-22]. Число т називається індексом схеми (2.4). Реалізуючим набором схеми (2.4) називатимемо вираз вигляду

$$\begin{array}{c} p^{(1)}, p^{(2)}, \ldots, p^{(f)}; \\ P_1, P_2, \ldots, P_f \end{array}$$

де $p^{(1)}, p^{(2)}, \ldots, p^{(f)}$ – список (без повторень) всіх змінних, що входять в (2.4), а P_i – слово в алфавіті A, яке називається значенням змінної в даному реалізуючому наборі (i=1,2...,f). Якщо замість кожного входження кожної змінної в схему (2.4) підставити значення цієї змінної в R (R – це конкретний

реалізуючий набір схеми (2.4)), то всі рядки схеми перетворяться на слова в алфавіті A. Так виходить реалізація продукційної схеми (2.4) реалізуючим набором R. Конструктивний об’єкт називатимемо реалізацією схеми (2.4), якщо він є реалізацією схеми яким-небудь реалізуючим набором. Слово Q називатимемо словом, що безпосередньо виводиться по схемі (2.4) із слів $Q_1, Q_2, ..., Q_m$, якщо вираз

$$Q_1, Q_2, ..., Q_m \Rightarrow Q$$

є реалізацією схеми (2.4). Список слів називається виводом в КЧ, якщо кожне слово списку є аксіомою даного КЧ або безпосередньо виводиться по якій-небудь схемі із попередніх слів в даному списку. Довжиною виводу називатимемо число слів у виводі. Слово P називатимемо таким, що виводиться в КЧ, якщо можна побудувати вивід в КЧ, останнім словом якого є P.

Поняття виводу і виводимості в КЧ дозволяють визначити еквівалентність числень і їх відношення з поняттям множини (2.4). Якщо алфавіт містить A, то воно називається численням над A. Два числення над A еквівалентні відносно A, якщо будь-яке слово в A виводиться в першому численні тоді і тільки тоді, коли воно виводиться в другому.

Пара

$$(A,T), \tag{2.5}$$

де T – канонічне числення над A, є поданням множини що виводяться в T слів алфавіту A. При цьому A називатимемо основним алфавітом подання, а доповнення до повного алфавіту числення T називатимемо допоміжним алфавітом, при цьому говорять, що T строго представляє множину слів, що виводяться в ньому. Множина M слів в A злічима, якщо для неї існує подання (2.5).

Для формалізації функціональної моделі ЛО введемо: {R} – множина ПЛО, що приймаються даним об’єктом; {T} – множина передаваних даним об’єктом ПЛО.

Вважаємо, що функціональна модель оперує з множинами {A}, {R} і {T}. Оскільки для кожного атрибута A_i множина, на якій він визначений, – S_{Ai}, може мати різну природу [22, 27], то елементи цієї множини можна інтерпретувати досить широко: як програмні коди, виклики функцій операційної системи, графічні структури та ін.

Також виділимо типи станів ЛО у функціональній моделі:

а)стани, в яких можливий прийом елементів множини {R}, надалі позначених як R_i**;**

б) стани, в яких неможливий прийом R_i.

Оскільки множина функцій повинна враховувати співвідношення атрибутів і зміст R_i**,** то необхідно ввести предикати, які утворюватимуть множину допустимих предикатів:

$$\{Pr\} = (Pr_l, Pr_2, \dots, Pr_\varphi).$$

Для аналізу складних умов і співвідношень будуватимемо формули над предикатами в мові числення висловлювань і позначати їх F(Pr) або просто F.

Алфавіт числення $K_{ФМ}$ визначимо таким чином: A = ({R}, {T}, {A}, {S}, {Pr}, &, $\vee$, (,), $\neg$, $\rightarrow$, $\Im$, ∇, $\varnothing$), де {S} – множина станів ЛО, $\Im$ – символ порожнього слова. До нього включаємо символи мови ЛО для побудови формул F. Алфавіт змінних включатиме змінні P = (p, q, f, hA), де p – послідовність вхідних ПЛО, q – послідовність вихідних ПЛО, f – послідовність формул з предикатами Pr в ЛО, hA – список атрибутів ЛО, для якого будується функціональна модель.

Аксіому числення задамо як A = ($\varnothing$ $\Im$ S_0 $\Im$ hA(0) $\Im$ $\varnothing$ $\Im$ $\varnothing$), де $\varnothing$ означає порожній стан змінної, а під hA(0) розуміється список вигляду

$$hA(0) = \langle\langle N_{A1},\ S_{A1}, V_{A1}(0)\rangle; \langle N_{A2},\ S_{A2}, V_{A2}(0)\rangle; \ldots \langle N_{An},\ S_{An}, V_{An}(0)\rangle\rangle,$$

де $V_{Ai}(0)$ позначає значення і-го атрибуту у момент часу t = 0, тобто у момент початку функціонування ЛО.

Правила виводу для числення $К_{ФМ}$ будуватимемо як схеми правил, оскільки в конкретній функціональній моделі виходитиме різна кількість правил виводу, які мають вигляд, що задовольняє запропонованим схемам:

$$\text{Схема 1: } R_i p \Im S_0 \Im hA(0) \Im q \Im f \Rightarrow p \Im S_i \Im hA(R_i) \Im q, T_i \Im f, F_i; \qquad (2.6)$$

$$\text{Схема 2: } R_i p \Im S_0 \Im hA(0) \Im q \Im f \Rightarrow \nabla p \Im S_i \Im hA(R_i) \Im q, T_i \Im f, F_i. \qquad (2.7)$$

Схеми (2.6) і (2.7) задають правила, що виводять зі стану S_0 в стани типу а) і б) відповідно. Для позначення неможливості обробки вхідної послідовності R_i використовується службовий символ ∇. У цих схемах породжується вихідний ПЛО T_i, і формула F_i, при цьому ми допускаємо можливість завдання $T_i = \varnothing$ і $F_i = \varnothing$, що дозволяє уникнути зайвих схем виводу [18, 51].

$$\text{Схема 3: } R_i p \Im S_i \Im hA \Im q \Im f \Rightarrow p \Im S_j \Im hA(R_i) \Im q, T_j \Im f, F_j; \qquad (2.8)$$

$$\text{Схема 4: } R_i p \Im S_i \Im hA \Im q \Im f, F_i \Rightarrow p \Im S_j \Im hA(R_i) \Im q, T_j \Im f. \qquad (2.9)$$

Схеми (2.8) і (2.9) визначають переходи із станів типу а) в стани типу а), з аналізом істинності F_i або без аналізу. Допускаємо також, що в F_i може бути задана формула ($\neg F_i$), тобто перевіряється істинність заперечення деякої формули. Таке розширення допустиме, оскільки у численні висловлювань істинність або помилковість будь-якого вислову може бути точно встановлена. Оброблена формула F_i виключається з подальшого процесу виводу. При циклічній поведінці ЛО необхідна формула може знову породжуватися схемами (2.8).

$$\text{Схема 5: } R_i p \Im S_i \Im hA \Im q \Im f \Rightarrow \nabla p \Im S_j \Im hA(R_i) \Im q, T_j \Im f, F_j; \quad (2.10)$$

$$\text{Схема 6: } R_i p \Im S_i \Im hA \Im q \Im f, F_i \Rightarrow \nabla p \Im S_j \Im hA(R_i) \Im q \Im f\ . \quad (2.11)$$

Схема (2.10) задає перехід із стану типу а) в стан типу б) без аналізу F, а схема (2.11) – з аналізом F.

$$\text{Схема 7: } \nabla p \Im S_i \Im hA \Im q \Im f \Rightarrow p \Im S_j \Im hA(R_i) \Im q, T_j \Im f, F_j; \quad (2.12)$$

$$\text{Схема 8: } \nabla p \Im S_i \Im hA \Im q \Im f, F_j \Rightarrow p \Im S_j \Im hA(S_i) \Im q, T_j \Im f\ ; \quad (2.13)$$

$$\text{Схема 9: } \nabla p \Im S_i \Im hA \Im q \Im f \Rightarrow \nabla p \Im S_j \Im hA(S_i) \Im q, T_j \Im f, F_j; \quad (2.14)$$

$$\text{Схема 10: } \nabla p \Im S_i \Im hA \Im q \Im f, F_i \Rightarrow \nabla p \Im S_j \Im hA(S_i) \Im q, T_j \Im f\ . \quad (2.15)$$

Схеми (2.12), (2.13) визначають переходи із станів типу б) в стани типу а), а схеми (2.14) і (2.15) із станів типу б) в стани типу б). У цих схемах правил закладається можливість повернення в стан S_0 і зупинки при переході в такий стан S_j з якого немає можливості подальшого виводу.

У схемах правил виводу (2.6) – (2.15) задаються в загальному вигляді функціональні перетворення $hA(S_i)$, які можна визначити як перетворення над значеннями атрибутів: $V_{Ai} := f_{(k)}(V_{A\phi 1}, ..., V_{A\phi k})$, де k – арність функціонального символу, $V_{A\phi i}$ – і-й аргумент функції f(k), узятий із списку значень атрибутів.

Слід зазначити, що носієм будь-якого процесу в інформаційній системі є окрема функціональна система, яка утворена вибірковою участю вироблених у цьому процесі функціональних компонентів і зв’язків між ними.

Загалом завдання забезпечення кібербезпеки інформаційних систем полягатиме у створенні умов для функціонування організаційної інформаційної системи під впливом кібернетики, щоб у ній відбувалися лише

ті процеси і лише ті функціональні утворення системи відповідали цілям для якою була створена ця система. Якщо процес відбувається тільки у функціональному компоненті системи, який задовольняє мету створення системи, то компонент буде в дозволеному стані - безпечному стані. Наявність функціонального компонента в дозволеному стані визначає його стан безпеки. У свою чергу, стан безпеки інформаційної системи визначається станом безпеки її функціональних компонентів.

Неможливо оцінити стан функціональних компонентів системи в рамках самої системи. Вирішити цю задачу можна шляхом моніторингу стану безпеки функціональних компонентів системи. Для цього необхідно розробити та створити зовнішню систему – агент моніторингу стану безпеки функціональних компонентів системи, яка реалізовуватиме специфічні функції ідентифікації та діагностики стану безпеки. Зовнішня система означає, що агент не є частиною сформованої цільової функціональної системи.

Слід підкреслити, що основою методу виявлення шкідливого процесу на основі аналізу стану логічних об’єктів має бути динамічна модель функціонування виділених логічних об’єктів. За допомогою цієї моделі ми зможемо визначати поведінку вибраних логічних об’єктів, аналізувати їх для визначення початкової системи даних для встановлення стану безпеки логічних об’єктів для завдань ідентифікації та діагностичної інформації.

Розглянемо зміст запропонованого методу моделювання функціональної динаміки логічних об’єктів та визначення закономірностей їх функціонування.

Передбачається, що динамічна модель функції логічного об'єкта буде представлена системою алгебраїчних рівнянь. Розв’язком системи рівнянь буде регулярний вираз, який визначає поведінку логічного об’єкта. Застосовуючи цей метод побудови динамічної моделі функціонування логічного об’єкта та визначення законів його функціонування, можна буде скласти коректні твердження та розв’язати для визначення необхідних процесів (згідно з відповідним протоколом) у просторі станів виділений

логічний об'єкт як скінченна автоматична система. Правильне визначення структури простору станів майбутніх логічних об’єктів дозволить нам вирішувати різні прикладні задачі, особливо ідентифікації та діагностики станів безпеки, а також визначати ефективні алгоритми для вирішення цих проблем.

На першому етапі підходу ми аналізуємо вимоги програми, транспорту, мережевого рівня або служби захисту вибраного протоколу, щоб визначити частковий набір функцій f, які повинен реалізувати логічний об’єкт. Виходячи з того, що кожна часткова функція fn реалізується відповідною процедурою, визначення та розгляд цих процедур є принциповою відмінністю нашого підходу.

Слід зазначити, що системні компоненти (логічні об’єкти), що працюють відповідно до протоколів на прикладному, транспортному, мережевому рівні та службах захисту, реалізують обмежений набір функціональних можливостей, визначених протоколами, і мають поняття стану. Кожна функція реалізується процесом, який відображається у відповідних станах системи та переходах між ними. Тому для дослідження функцій логічних об’єктів інформаційних систем за відповідними протоколами скористаємося положеннями теорії скінченних автоматів для абстрагування та моделювання.

Задача аналізу скінченних автоматизованих систем полягає в знаходженні регулярного виразу для множини прийнятних процесів. Вихідними даними для задачі аналізу є множина перетворених співвідношень і допустимих процесів [99]. Процеси, які реалізують набір функцій згідно з відповідним протоколом, будуть формувати допустиму множину процесів логічного об'єкта. Сукупність прийнятних процесів для дискретної системи є мовою в державному алфавіті [99]. Тому для опису цих процесів можна застосувати методи, визначені лінгвістичною алгеброю. На другому кроці методу визначається множина станів Sn і множина процесу Pn виділеного логічного об’єкта, що призводить до реалізації функції fn.

Для аналізу логічного об’єкта за певним протоколом необхідно визначити набір запущених процесів у просторі станів дискретної системи (другий крок у методі побудови динамічної моделі логічної системи). об'єкт є функцією логічного об'єкта). Правильне визначення структури простору станів, тобто набору станів і функції переходу, визначає ефективність алгоритму для подальшого вирішення завдань моніторингу, ідентифікації та діагностики безпечного (небезпечного) стану. Слід підкреслити, що функціональний процес логічного об'єкта змінює його стан. Цю послідовність змін можна представити у вигляді орієнтованого графа (графа переходів). Вершини такого графа є станами системи, а дуги з'єднують вершини, зумовлені можливістю переходу з одного стану в інший. Крім того, динамічні моделі логічних об'єктів можуть бути описані системами алгебраїчних рівнянь у просторі станів.

На третьому кроці методу ми моделюємо окремі процеси функцій вибраних логічних об’єктів, які призводять до реалізації кожної функції множини f через формалізацію у вигляді алгебраїчних рівнянь. Слід зазначити, що множина допустимих процесів для скінченних автоматичних систем є мовою в алфавіті стану і є регулярною. Зв’язок між станом s кінцевої автоматичної системи та допустимим процесом Fs описується канонічною системою лінійних рівнянь алгебри регулярних мов [99], наприклад,

$$F_s=\alpha(s)\vee\bigvee_{s'\to s} F_{s'}\, s,$$

де $\alpha(s) = (\text{якщо } s \in S_0, \text{то } s, \text{інакше } \emptyset)$, $S_0 -$ множина початкових станів.

На четвертому кроці ми будуємо динамічну модель функції вибраного логічного об’єкта шляхом об’єднання окремих алгебраїчних рівнянь у їх систему. На цьому кроці ми розглянемо проблему аналізу кінцевих автоматичних систем. Розв’язок системи алгебраїчних рівнянь у просторі станів як динамічної моделі логічного об’єкта є набором регулярних виразів

для прийнятного процесу. Цей регулярний вираз визначає його можливу траєкторію, задану в просторі станів, що становить закон функціонування цього логічного об’єкта. За умови, що стан і допустимий процес скінченної автоматичної системи повністю визначені, система лінійних рівнянь має єдиний розв’язок. Слід підкреслити, що повне визначення стану та процесу дозволу розкриває сутність цього логічного об’єкта та є необхідною умовою (орієнтиром) для визначення стану його безпеки.

На п’ятому кроці методу шляхом розв’язування системи алгебраїчних рівнянь визначається алгоритм для обраного логічного об’єкта. Закони поведінки вибраних логічних об'єктів є початковим джерелом даних для постановки та вирішення проблем ідентифікації та діагностики стану їх безпеки. У результаті аналізу формалізованих законів роботи обраного логічного об’єкта ми отримаємо точні вихідні дані, тобто допустимий початковий стан, набір наступних допустимих станів, набір подій і послідовність станів переходи.

На шостому кроці проводимо аналіз формалізованого закону функціонування обраного логічного об’єкта з метою формування вихідних даних для постановки наступних задач.

Попередня оцінка адекватності функціональної динамічної моделі логічних об'єктів. Слід зазначити, що при оцінці адекватності моделей у реалізаціях дискретних систем використовується поняття гомоморфізму [99]. Розкриємо суть цього поняття. Нехай S і S' — дискретні системи з набором прийнятних процесів F і F' відповідно. Розглянемо відображення γ: S → S'. Це відображення природно можна продовжити до відображення γ: P(S) → P(S'), припускаючи γ(s1 … sn) = γ(s1) … γ(sn). Відображення γ називається гомоморфізмом систем S у S', якщо образ допустимого процесу системи S при цьому відображенні є допустимий процес системи S'. У цьому випадку S' називається гомоморфною моделлю, а S - гомоморфною реалізацією системи S'. Цей термін пояснюється практичними проблемами [1, 9, 99].

Вивчаючи властивості системи S в результаті абстракції, ми створюємо модель S', яка є гомоморфною моделлю. Якщо ми встановимо модель S', то створена за цією моделлю система S є гомоморфною реалізацією S'. У той же час, спостерігаючи, що система S запускає прийнятний процес p, який реалізує систему S' за допомогою гомоморфізму γ, ми явно відновлюємо систему S', що виконує процес q = γ(p). Гомоморфна реалізація S системи S' називається повною, якщо кожен допустимий процес системи S' має реалізацію, тобто γ(F) = F'.

Зробимо деякі суттєві зауваження.

По-перше, специфікація протоколів прикладного, транспортного, мережевого рівнів та послуг захисту визначає сутність відповідних логічних об’єктів та зміст їх динамічних моделей.

По-друге, рівень деталізації динамічної моделі відповідних логічних об’єктів інформаційних систем, яка описується в термінах станів та процесів, є достатнім для її дослідження та постановки задач ідентифікації та діагностування його стану безпеки.

2.4. Розробка моделі процесу функціонування ідентифікації та діагностування стану логічних об’єктів

В ході аналізу динамічної моделі функціонування ЛО встановлено той факт, що в загальних моделях ідентифікації та діагностування (рис. 2.5, рис. 2.6) застосовуються різні способи функціонування. Під функціонуванням слід розуміти знаходження шляху досягнення мети або цілей даними. Оскільки різні логічні об’єкти володіють своєю специфікою, то навряд чи є можливим застосування деякого універсального методу функціонування для всіх підсистем.

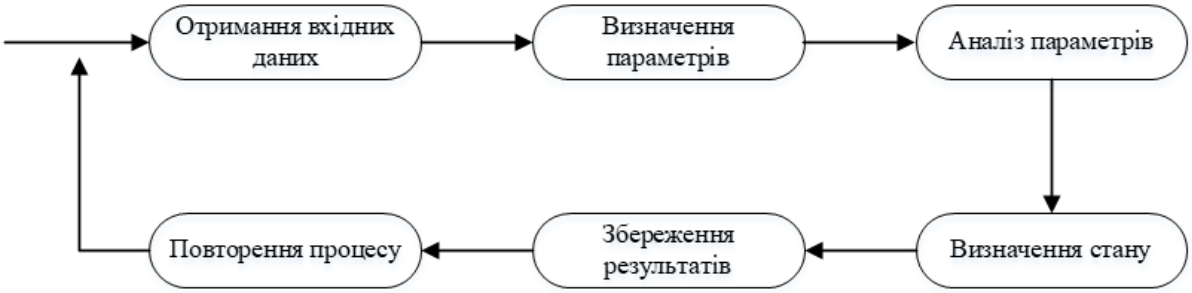

Рис. 2.5. Загальна модель функціонування ідентифікації стану логічних об’єктів

Модель процесу функціонування ідентифікації стану логічних об'єктів може бути описана наступним чином:

отримання вхідних даних: процес починається з отримання вхідних даних про логічні об'єкти, для яких необхідно визначити стан;

визначення параметрів: наступним кроком є визначення параметрів, що характеризують кожний логічний об'єкт. Ці параметри можуть бути залежні від типу об'єкту і можуть включати такі параметри, як стан, рівень доступу, наявність або відсутність певних функцій та інші;

аналіз параметрів: після визначення параметрів, що характеризують логічний об'єкт, проводиться їх аналіз з метою визначення стану цього об'єкту. Аналіз може включати перевірку наявності певних функцій, порівняння параметрів з попередніми значеннями і т.д.;

визначення стану: на основі аналізу параметрів визначається стан кожного логічного об'єкту. Стан може бути визначений за допомогою різних маркерів, таких як "активний", "неактивний", "підключений", "відключений" і т.д.;

збереження результатів: результати ідентифікації стану логічних об'єктів зберігаються для подальшого використання. Це може включати створення журналів, що містять інформацію про стан кожного об'єкта, створення сповіщень про зміну стану та інші дії;

повторення процесу: процес ідентифікації стану логічних об'єктів може бути повторено з регулярним інтервалом для забезпечення актуальності інформації про стани об'єктів. Крім того, процес може бути запущений відповідно до певної події, наприклад, якщо змінюється параметр об'єкта або відбувається певна дія з об'єктом.

Ця модель може бути використана для автоматизації процесу моніторингу логічних об'єктів, що дозволить ефективніше виявляти проблеми та швидше реагувати на них. Крім того, цей підхід дозволяє покращити безпеку системи, оскільки виявлення зміни стану об'єкта може бути попереджено до того, як ця зміна стане критичною.

Рис. 2.6. Загальна модель функціонування діагностування стану логічних об’єктів

Модель процесу функціонування діагностування стану логічних об’єктів може включати такі етапи:

Збір даних про стан логічних об'єктів: на цьому етапі здійснюється збір даних про стан об'єктів за допомогою моніторингових систем. Ці дані можуть включати параметри об'єктів, такі як розмір, тип, стан, час відповіді та ін.

Аналіз даних: на цьому етапі аналізуються зібрані дані про стан об'єктів з метою виявлення аномалій та невідповідностей, що можуть свідчити про несправність об'єкта.

Визначення проблемних об'єктів: на цьому етапі визначаються ті

об'єкти, стан яких може бути відхиленням від норми. Цей етап може включати порівняння стану об'єктів з попередніми даними або зі стандартними значеннями.

Діагностика причин несправностей: на цьому етапі виявляються причини, які можуть призвести до відхилень у стані об'єктів. Цей етап може включати використання методів машинного навчання або аналізу даних для виявлення кореляцій між параметрами об'єктів та їх станом.

Вирішення проблем: на цьому етапі приймаються рішення щодо вирішення проблем, що були виявлені на попередніх етапах. Це може включати виклик фахівців для ремонту об'єктів або відновлення роботи системи.

Ця модель може бути використана для автоматизації процесу діагностування стану логічних об'єктів, що дозволить ефективніше виявляти проблеми та швидше реагувати на них. Для реалізації цієї моделі можуть бути використані різноманітні технології та інструменти, такі як моніторингові системи, системи автоматичного аналізу даних, системи машинного навчання та інші.

Важливим елементом у розробці такої моделі є також визначення критеріїв, за якими буде визначатися стан об'єктів та їх нормальна робота. Для цього можуть бути використані стандарти та рекомендації, які визначають параметри та характеристики об'єктів, що вважаються нормальними для їх роботи.

Загалом, модель процесу функціонування діагностування стану логічних об'єктів дозволяє автоматизувати та оптимізувати процес виявлення та вирішення проблем у системах, що включають багато логічних об'єктів. Це дозволяє забезпечити високу ефективність та надійність роботи таких систем та зменшити ризик їх відмови.

Більш доцільним є розробка на базі динамічної моделі функціонування ЛО, що дозволить створювати різні моделі ідентифікації та діагностування (МІД).

З погляду реалізації дій моделі слід розбити на три класи:

1) з визначеною кінцевою множиною елементарних дій [1-35];
2) з множиною планів [51-58];
3) з довільними повідомленнями і діями в логічній мові [27].

Узагальнюючи ці моделі функціонування, в узагальненій моделі МІД пропонується наступний варіант функціонування.

Вважаємо, що МІД має визначену множину статичних цілей $GS = \{gs^i | i = 1,...,n\}$. Апріорі відомі шляхи досягнення цілей, тобто побудовані логічні об'єкти $(IO^i | i = 1,...,n)$, функціонування яких повинне вести до gs^i. Тут кожен ЛО покриває деякий план, слідуючи термінології [51-58]. Усередині ж цього плану, тобто в моделі поведінки ЛО, можуть бути сформовані довільні повідомлення і довільні послідовності дій, як показано у розд. 2.3.

Тоді модель функціонування задається функцією функціонування $SR : GS \to VO$ де VO – множина вкладених ЛО і-го МІД. Це відображення однозначне, але не взаємно, оскільки можливо, що декілька цілей досягаються одним і тим же ЛО. Модель активних дій визначається відображенням $AD : GA \to VO$, яке вибирає необхідні для запуску у нинішній момент ЛО. Процес запуску ЛО реалізується на стадії функціонування СБ. Зміна активних цілей (зміна множини GA) приводитиме до зупинки і запуску відповідних ЛО.

Далі розробляється продукційна формалізація функціонування агента моніторингу стану логічних об'єктів в узагальненій моделі МІД. Введемо наступні позначення: SA(0) – початкова конфігурація структури атрибутів МІД, Lvo – список вкладених ЛО, SMA(0) – початкова конфігурація МІД-оточення, Ls – список стратегій МІД, Lsg – список статичних цілей, Ltg – список цілей, що отримуються, Ldg – список цілей, що передаються, Ldag – список автономних динамічних цілей, Lag – список активних цілей.

Аксіома числення K_{MVB} у момент первинного запуску МІД приймає вигляд

$$AA = \{SA(0); Lvo; SMA(0); Ls(0); Lsg(0); \varnothing; \varnothing; \varnothing; \varnothing\} \qquad (2.16)$$

Для спрощення запису інформаційного простору V і його стану SV введемо

$$Lia = \{NIA_j, \{A^{\psi}_{IA_j} \,|\, \psi = 1,...,k\} \,|\, j = 1,...,l\},$$

$$Lio = \{NIO_j, \{A^{\psi}_{IO_j} \,|\, \psi = 1,...,k\} \,|\, j = 1,...,l\}.$$

V представляємо двома списками – Lia, Lio, а S_V – списками атрибутів Laia, Laio. Тоді (V,SV) $\Leftrightarrow$ Lia, Lio, Laia, Laio.

Формування першої ситуації МІД-оточення описуємо правилом

(1) $AA \Rightarrow (2)Lia, Lio, \varnothing, \varnothing; AA$

Виконання обсервації визначиться правилом (2)

(2) $Lia, Lio, \varnothing, \varnothing; AA \Rightarrow (3)FV(lia, lio, \varnothing, \varnothing); AA$.

Далі для скорочення запису позначатимемо складові антецедента кутовими дужками з нижнім індексом як номер, і, якщо вміст даного компоненту не змінюється в даному правилі, то просто повторюватимемо $\Re_i$. В правилах числення приписуємо на початок антецедента номер правила в круглих дужках (...), а в консеквенті формуватимемо номер наступного правила.

Тоді формування початкового стану МІД-оточення опишеться правилом

$$(3)\ \ Lia, Lio, Laia, Laio; AA \Rightarrow (4)\ \ \Re_1 \langle S_A \rangle_2 \langle lvo \rangle_3$$
$$\langle \varnothing; Lia' := FP(\Re_1), Lio' := FP(\Re_1), Laia' := FP(\Re_1), Laio' := FP(\Re_1) \rangle_4$$
$$\langle ls \rangle_5 \langle lsg \rangle_6 \langle \varnothing \rangle_7 \langle \varnothing \rangle_8 \langle \varnothing \rangle_9 \langle \varnothing \rangle_{10}$$

Вибір стратегії задається правилом (4)

$$(4)\ \ \Re_1\Re_2\Re_3\Re_4 \langle ls \rangle_5 \Re_6\Re_7\Re_8\Re_9\Re_{10} \Rightarrow$$
$$\Rightarrow (5)\ \ \Re_1\Re_2\Re_3\Re_4 \langle s' := FSS(ls, \Re_4, \Re_{10}); ls \rangle_5 \Re_6\Re_7\Re_8\Re_9\Re_{10}$$

Для прийому цілей, що отримуються введемо операцію читання буфера цілей – read(bltg). Відповідне правило приймає вигляд

$$(5)\ \ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10} \Rightarrow$$
$$\Rightarrow (6)\ \ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7 \langle Ltg := read(bltg) \rangle_8 \Re_9\Re_{10}$$

Далі можна сформувати цілі передавані, динамічні цілі з множини $G^{AVT(D)}$ і вибрати статичні цілі

$$(6)\ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10} \Rightarrow$$
$$\Rightarrow (7)\ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7 \left\langle Ldg := h^{DOWN}(\Re_4, s', U, Lag) \right\rangle_8$$
$$\left\langle Ldag := h^{AVT(D)}(\Re_4, s', U, Lag) \right\rangle_9 \Re_{10} \left\langle Lavtgs := h^{AVT(D)}(\Re_4, s', U, Lag) \right\rangle_{11}$$

Тепер може бути побудована множина активних цілей

$$(7)\ \ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow (8)\ \Re_1\Re_2\Re_3\Re_4\Re_5\Re_6\Re_7\Re_8$$
$$\Re_9 \left\langle Lag := h^{A}(s', \Re_9, \Re_{10}, \Re_{11}, Z) \right\rangle_{10} \Re_{11}$$

Для кожної активної мети ga^j вибираємо підмножину ЛО для яких апріорі припускаємо, що вони можуть вести до даної активної мети. Вважаємо, що цілі, що містяться в Lag не суперечливі, виконання цієї вимоги повинне забезпечити перетворення h^A що використовує онтологію активізації цілей Z. Отже, множина VO розбивається на n непересічних підмножин, таких що

$$VO \subseteq \bigcup_{j=1}^{n} VOga^j .$$

Такі підмножини $VOga^j$ називатимемо модельною множиною вкладених ЛО і позначати $ModelSet^j$. ЛО, що входять в j-ту модельну множину позначатимемо $IO_m^{j,i}$ де нижній індекс *t* показує [18, 51], що даний ЛО моделюється в процесі функціонування. Для здійснення такого розбиття потрібна функція $FMVO = h^{VO}(Lvo, Lag, X)$, де X – онтологія, що описує логіку такого розбиття і необхідна тому, що потенційна множина динамічних цілей може бути нескінченною.

Формулюємо правило

$$(8)\ \Re_1\Re_2\langle lvo\rangle_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\langle Lag \neq \varnothing\rangle_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow \Re_1\Re_2\left\langle h^A(Lvo, Lag, X\right\rangle_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\langle Lag \neq \varnothing\rangle_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow (9)\ \Re_1\Re_2\langle ModelSet, Lvo\rangle_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Логічні об’єкти відповідно до своєї моделі поведінки формують ПЛО (пасивні логічні об’єкти) і передають їх як повідомлення; а також виконують перетворення h^A над структурою своїх атрибутів. Тому в процесі функціонування ми моделюємо поведінку всіх ЛО, що належать даній модельній множині для пошуку шляху до мети ga^j на глибину в *gp* кроків.

Вважаємо, що параметр *gp* визначається вибраною стратегією s^i і вводимо два лічильники: лічильник модельних кроків Cmgp і лічильник актуальних кроків *Cagp*. Виконуємо модельні кроки для модельних множин на глибину *gp*. Після кожного кроку обчислюємо передбачувану оцінку стану МА-оточення для всіх модельованих ЛО і робимо наступний модельний крок. Повторюємо модельні кроки до досягнення *Cmgp* = *gp*.

Далі обчислюємо передбачувану оцінку перетворень цілей для всіх ЛО і відхилення від цілей (міру близькості до мети). Визначаємо найменше відхилення

$$\Delta_{\min} = \min(\Delta(IO_m^{j,i}) \big| IO_m^{j,i}) \in ModelSet^j .$$

Ці дії формалізуємо правилами (9)

$$(9)\ \Re_1\Re_2 \langle LModelSet; Lvo\rangle_3 \Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow \Re_1\Re_2 \langle LModelSet; Lvo\rangle_3 \Re_4 \left\langle s'; Ls; Cmgp := h^{gp}(s'); Cagp := h^{gp}(s')\right\rangle_5$$
$$\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow \Re_1\Re_2 \left\langle LModelSet(K_{МП}(IO_m^{j,i}) \big| IO_m^{j,i} \in ModelSet^j; j = 1,...,n); Lvo\right\rangle_3$$
$$\Re_4 \langle s'; Ls; Cmgp := Cmgp - 1; Cagp\rangle_5 \Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow \Re_1\Re_2$$
$$\left\langle LModelSet(K_{МП}(IO_m^{j,i}), FP(Rt, K_{МП}(IO_m^{j,i})) \big| IO_m^{j,i} \in ModelSet^j; j = 1,...,n); Lvo\right\rangle_3$$
$$\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow$$
$$\Rightarrow \Re_1\Re_2$$
$$\left\langle LModelSet(K_{МП}(K_{МП}(IO_m^{j,i}), Ri) \big| IO_m^{j,i} \in ModelSet^j; j = 1,...,n); Lvo\right\rangle_3$$
$$\Re_4 \langle s'; Ls; Cmgp := Cmgp - 1; Cagp\rangle_5 \Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11} \Rightarrow ... \Rightarrow ... \Rightarrow ... \Rightarrow$$

$$\Rightarrow \Re_1\Re_2\left\langle \mathrm{LModelSet}(K_{МП}(K_{МП}(IO_m^{j,i}),Ri)\middle| IO_m^{j,i}\in \mathrm{ModelSet}^j;\, j=1,...,n);Lvo\right\rangle_3$$
$$\Re_4\langle s';Ls;Cmgp=0;Cagp\rangle_5\,\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow \Re_1\Re_2$$
$$\left\langle \mathrm{LModelSet}(h^{ga_j}(K_{МП}(K_{МП}(IO_m^{j,i}),Ri))\middle| IO_m^{j,i}\in \mathrm{ModelSet}^j;\, j=1,...,n);Lvo\right\rangle_3$$
$$\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$

$$\Rightarrow \Re_1\Re_2\left\langle \mathrm{LModelSet}(\Delta(ga^j,ga^j)\middle| IO_m^{j,i}\in \mathrm{ModelSet}^j;\, j=1,...,n);Lvo\right\rangle_3$$
$$\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow (10)\ \Re_1\Re_2\left\langle \mathrm{LModelSet}(\Delta_{min}:=\min(\Delta(ga^j,ga^j))\middle| j=1,...,n);Lvo\right\rangle_3$$
$$\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Після цього кроку слід сформувати множину актуальних ЛО, які діятимуть і посилатимуть повідомлення, таким чином, переходячи до фази активних дій і реалізації моделі ідентифікації та діагностування:

$$(10)\ \Re_1\Re_2\left\langle (\Delta_{min}^j\,|\,j=1,...,n),lvo\right\rangle_3\,\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow \Re_1\Re_2\left\langle (IO_m^{1,1},...,IO_m^{1,min},...,IO_m^{1,k_1})(IO_m^{2,1},...,IO_m^{2,min},...,IO_m^{2,k_2})...\right.$$
$$\left.(IO_m^{n,1},...,IO_m^{n,min},...,IO_m^{n,k_n});lvo\right\rangle_3\,\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow (11)\ \Re_1\Re_2\left\langle lvo:=h^R(lvo)\right\rangle_3\,\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Зміст цього правила полягає в тому, що ЛО, відповідні Δ_{min}^j позначаються як актуальні, а решта $IO_m^{j,i}$ заміщають своїх батьків в списку Lvo за допомогою перетворення зворотньої підстановки h^R. Вміст компонента $\Re_3$ приймає вигляд $IO_1,...,IO_i^a,...,IO_j^a,...,IO_I$, де I – потужність множини вкладених логічних об’єктів $\{V_O\}$.

Далі включаємо в цю множину і виділений логічний об’єкт IO_0, що представляє дії ІА над своїми атрибутами, який активізується за умовчанням.

Описуємо фазу активних дій наступним правилом:

$$(11)\ \Re_1\Re_2\left\langle IO_0^a,...,IO_j^a,...\right\rangle_3\Re_4\langle s';Ls;Cagp\rangle_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow\Re_1\Re_2\left\langle K_{МП}(IO_0^a),...,K_{МП}(IO_j^a),...\right\rangle_3\Re_4\langle s';Ls;Cagp:=Cagp-1\rangle_5$$
$$\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow...\Rightarrow...\Rightarrow...\Rightarrow$$
$$\Rightarrow(12)\ \Re_1\Re_2\left\langle K_{МП}(K_{МП}...(IO_0^a)),...,K_{МП}(K_{МП}...(IO_j^a)),...\right\rangle_3\Re_4$$
$$\langle s';Ls;Cagp=0\rangle_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Після того, як активні дії закінчилися, можна виконати перевірку і зняти мітки активності з ЛО:

$$(12)\ \langle Lia,Lio,Laia,Laio\rangle_1\Re_2\left\langle IO_0^a,...,IO_j^a,...\right\rangle_3\Re_4\langle s';Ls;Cagp=0\rangle_5$$
$$\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow(13)\ [\langle Lia',Lio',Laia',Laio':=FV(Lia,Lio,Laia,Laio)\rangle_1$$
$$\Re_2\left\langle(IO_0,...,IO_j,...\right\rangle_3\Re_4\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Формуємо новий поточний стан МІД-оточення за допомогою правил (13) і (14)

$$(13)\ \langle Lia',Lio',Laia',Laio'\rangle_1\Re_2\Re_3\langle Pa;Rt;Fu\rangle_4\langle s';Ls\rangle_5$$
$$\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow(14)\ \Re_1\Re_2\Re_3$$
$$\langle Rt,Pa;Lia',Lio',Laia',Laio';Fu:=FP(Lia',Lio',Laia',Laio')\rangle_4$$
$$\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

$$(14)\ \Re_1\Re_2\Re_3\langle Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow\Re_1\Re_2\Re_3\left\langle(g\hat{a}^j := h^{ga_j}(Pa,Rt,Fu)\,|\,j=1,\ldots,n);Pa;Rt;Fu\right\rangle_4$$
$$\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow(15)\ \Re_1\Re_2\Re_3\left\langle\Delta(ga^j,g\hat{a}^j)\,|\,j=1,\ldots,n);Pa;Rt;Fu\right\rangle_4$$
$$\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Обчислюємо оцінки стану цілей від досягнутого стану МІД-оточення і визначаємо відхилення від мети [18]. Оскільки цілей може бути декілька (у разі j > 1), необхідно сформувати комплексну оцінку положення МІД в просторі цілей

$$\Delta_k = \sum_{j=1}^{n} a_i\Delta(ga^j,g\hat{a}^j),$$

що враховує значущість цілей, a_i – вагові коефіцієнти важливості цілей, n – потужність множини активних цілей. Тому вводимо правило (15)

$$(15)\ \Re_1\Re_2\Re_3\left\langle\Delta(ga^j,g\hat{a}^j)\,|\,j=1,\ldots,n);Pa;Rt;Fu\right\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow\nabla\Re_1\Re_2\Re_3\langle\Delta k;Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

Проводимо порівняння з допустимим комплексним відхиленням Δd за допомогою наступних правил:

$$(16)\ \nabla\Re_1\Re_2\Re_3\langle\Delta k>\Delta d;Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow(4)\ \Re_1\Re_2\Re_3\langle Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

$$(17)\ \nabla\Re_1\Re_2\Re_3\langle\Delta k\le\Delta d;Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}\Rightarrow$$
$$\Rightarrow(5)\ \Re_1\Re_2\Re_3\langle Pa;Rt;Fu\rangle_4\,\Re_5\Re_6\Re_7\Re_8\Re_9\Re_{10}\Re_{11}$$

При виконанні умови $\Delta k > \Delta d$ переходимо до правила (4), тобто переоцінюємо стратегію формування цілей; якщо ж $\Delta k \leq \Delta d$ то функціонування МІД продовжується у тій же самій стратегії.

Побудована в даному розділі модель функціонування в узагальненій моделі ідентифікації та діагностування (ІД) дозволяє описати такі відомі класи моделей реалізації поведінки як моделі із зумовленою кінцевою множиною елементарних дій; моделі з множиною планів; моделі з довільними повідомленнями і діями. На базі даної моделі можуть створюватися нові моделі реалізації поведінки МІД, що поєднують механізми різних класів.

На основі вищезазначених етапах та матриці переходів побудуємо функцональну модель (рис. 2.7) та розширену матрицю допустимих переходів станів логічного об’єкта (табл. 2.5), що дає нам можливість знизити вірогідність помилки та відобразити допустимі флаги функціональної моделі TCP протоколу.

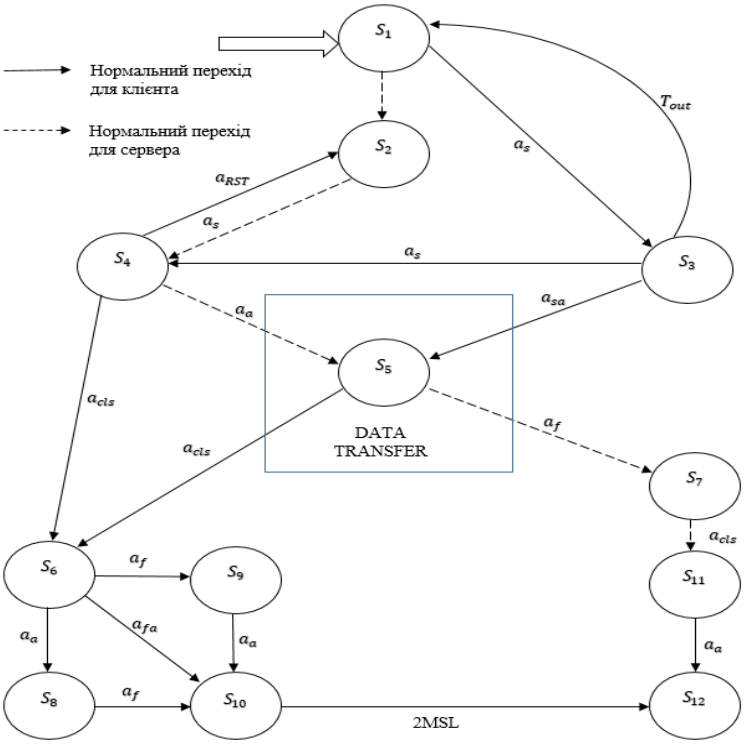

Рис. 2.7. Об’єднана функціональна модель TCP протоколу зі сторони клієнта та сервера

S_1 — CLOSED — з’єднання закрите, це вихідний стан.

S_2 — LISTEN — стан в якому знаходиться сервер, що очікує запити на встановлення з'єднання (passive open).

S_3 — SYN SENT — стан в якому знаходиться клієнт, що відправив серверу запит для встановлення з’єднання, це перший етап «трьохстороннього рукостискання» (active open).

S_4 — SYN RCVD — стан в якому знаходиться сервер при отриманні запиту на створення з’єднання і відправка його підтвердження. Другий етап «трьохстороннього рукостискання».

S_5 — ESTAB — в цьому стані з’єднання вважається встановленим, сервер переходить в стан S_5 та приймає підтвердження від клієнта на прийнятий від сервера відповідь, а клієнт переходить в стан S_5 та відправляє підтвердження про отримання відповіді від сервера.

S_6 — FIN WAIT-1 — перехід в стан S_6 виконується при досягненні кінця потоку даних a_{cls}, що передаються, при цьому надсилається сегмент зі встановленим прапором FIN.

S_7 — CLOSE WAIT — перехід в цей стан виконується при отриманні сегмента з встановленим прапором FIN, у відповідь надсилається підтвердження.

S_8 — FIN WAIT-2 — перехід в цей стан виконується при отриманні підтвердження для раніше відправленого сегмента зі встановленим прапором FIN.

S_9 — CLOSING — перехід в цей стан виконується при отриманні сегмента зі встановленим прапором FIN.

S_{10} — TIME WAIT — виконується або при отриманні сегмента зі встановленим прапором FIN і в такому випадку надсилається підтвердження, або при отриманні підтвердження раніше надісланого сегмента зі встановленим прапором FIN.

S_{11} — LAST-ACK — перехід виконується при закінченні вихідного потоку даних, надсилається сегмент зі встановленим прапором FIN.

S_{12} — CLOSED — перехід виконується або при отриманні підтвердження для раніше надісланого сегмента зі встановленим прапором FIN [1,9,58], або при закінченні тайм-ауту, який гарантує отримання підтвердження відправленого для сегмента зі встановленим прапором FIN.

T_{out}, a_{RST} — відбувається при виявленні помилок в стані S_2 та S_3.

Для виявлення атаки в TCP протоколі на основі EFSM використовуємо кілька критеріїв з матриці переходів та формул ідентифікації та діагностики станів:

Кількість SYN-пакетів, які надійшли на сервер в перші кілька секунд після встановлення з'єднання. Цей критерій може бути визначений за допомогою формули:

$$\mathrm{SYN_{prate}} = (\mathrm{SYN_c} \div \mathrm{TPC}) \times 100\%$$

Де $\mathrm{SYN_c}$ - кількість SYN-пакетів, TPC (total_packets_count) - загальна кількість пакетів, які надійшли на сервер.

Кількість відповідей на SYN-пакети, що містять підтвердження (ACK).

Якщо після прийняття SYN-пакету сервер відправляє підтвердження (ACK), то такий пакет не є частиною атаки. Тому, якщо кількість підтверджень на SYN-пакети є дуже низькою, то можна підозрювати наявність атаки. Цей критерій можна визначити за допомогою формули:

$$\mathrm{ACK_{rrate}} = (\mathrm{ACK_c} \div \mathrm{SYN_c}) \times 100\%$$

Де $\mathrm{ACK_c}$ - кількість підтверджень на SYN-пакети, $\mathrm{SYN_c}$- кількість SYN-пакетів.

Таблиця 2.5.

Розширена матриця допустимих переходів станів логічного об’єкта функціональної моделі TCP

Подія/стан	CLOSED	LISTEN	SYN_RCVD	SYN_SENT	ESTABLISHED	FIN_WAIT_1	FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
ACTIVE	SYN			SYN							
PASSIVE		SYN	SYN		SYN_ACK	FIN_WAIT_1	CLOSING	CLOSW_WAIT	LAST_ACK	TIME_WAIT	
SYN		SYN		SYN	SYN_ACK						
SYN_ACK			ESTABLISHED								
ESTABLISHED						FIN_WAIT_1	FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
FIN_WAIT_1			FIN_WAIT_1	FIN_WAIT_2		FIN_WAIT_1	CLOSING	TIME_WAIT		LAST_ACK	TIME_WAIT
FIN_WAIT_2					TIME_WAIT		FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
CLOSE_WAIT					LAST_ACK	FIN_WAIT_1	FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
CLOSING					TIME_WAIT	FIN_WAIT_2	CLOSING	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
LAST_ACK					TIME_WAIT	FIN_WAIT_2	FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	TIME_WAIT
TIME_WAIT						FIN_WAIT_2	FIN_WAIT_2	CLOSE_WAIT	CLOSING	LAST_ACK	

Кількість SYN-пакетів, що мають неправильні номери портів відправника та отримувача. Якщо відправник використовує випадкові номери портів, то можна припустити, що це частина атаки. Цей критерій можна визначити за допомогою формули:

$$\mathrm{INVP_{nrate}} = (\mathrm{INVP_c} \div \mathrm{SYN_c}) \times 100\%$$

Де $\mathrm{INVP_c}$ (invalid_port_count) - кількість SYN-пакетів з неправильними номерами портів, $\mathrm{SYN_c}$- кількість SYN-пакетів.

Отже, розроблена роширена матриця функціонування ЛО та модель ідентифікації та діагностування станів ЛО дозволяє підвищити точність та швидкість процесу діагностики. Дана модель може бути використана в системах моніторину та діагностики стану ЛО в режимі реального часу.

Висновки до розділу 2

1. Системи виявлення вторгнень, які базуються на моніторингу поведінки протоколу окремо від інших протоколів, залучених до сеансу, вже недостатньо для боротьби зі складною природою атак. Підхід, що ґрунтується на скінченних автоматах TCP є інноваційним з точки зору створення основи для протоколів для обміну корисною інформацією. Стани та змінні стану доступні між розширеними скінченними автоматами різних протоколів, щоб допомогти системі точно виявляти атаки. Система виявлення також є основою для всіх додатків, які базуються на TCP як протоколі транспортного рівня.

2. Встановлено, що виявлення шкідливих процесів, що реалізовуються на протокол TCP полягає в пошуку ненормальних шаблонів пакетів TCP RST. У типовому TCP-з'єднанні пакет RST надсилається, щоб вказати, що з'єднання

припиняється. Однак під час атаки зловмисник може надіслати велику кількість RST-пакетів, щоб порушити з’єднання. Відстежуючи прапори TCP у вхідних пакетах, потрібно шукати шаблони пакетів RST, які вказують на те, що відбувається атака. Загалом виявлення шкідливих процесів на рівні протоколу TCP вимагає ретельного моніторингу вхідного трафіку та глибокого розуміння нормальних шаблонів трафіку TCP.

Отже, необхідно розуміти функціонування протоколу транспортного рівня відповідно до стандартів RFC щодо використання скінченних автоматів, які визначають основні стани логічних об’єктів TCP протоколу у стані спокою.

3. Для постійного моніторингу є необхідність проведення постійного аналізу функціональних складових моделі ідентифікації та діагностування, на основі яких формалізуються компоненти моделі ідентифікації та діагностики логічного об’єкта та визначається функціонування логічних об’єктів інформаційної системи.

4. В результаті розроблено модель процесу функціонування ідентифікації та діагностування стану логічних об’єктів TCP протоколу на основі скінченних автоматів та розширеної матриці допустимих переходів станів логічного об’єкта функціональної моделі TCP, що дозволяє відслідковувати стани логічних об’єктів в режимі реального часу та відображає переходи в невідповідний стан, що символізує шкідливий процес в роботі протоколу.

5. Вперше розроблено модель ідентифікації та діагностування станів логічних об’єктів, наукова новизна якого полягає в тому, що вона ґрунтується на теорії скінченних автоматів на основі протоколу TCP, що дозволяє на стадії створення з’єднання ідентифікувати порушення пакету зі встановленими параметрами та діагностувати роботу транспортного протоколу.

РОЗДІЛ 3 РОЗРОБКА МЕТОДУ ВИЯВЛЕННЯ ШКІДЛИВИХ ПРОЦЕСІВ В ІНФОРМАЦІЙНІЙ СИСТЕМІ ОРГАНІЗАЦІЇ ТА РЕКОМЕНДАЦІЙ ЩОДО ЙОГО ЗАСТОСУВАННЯ

3.1. Метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі алгоритму аналізу основних компонентів

Аналіз основних компонентів (Principal Component Analysis - PCA) – це алгоритм неконтрольованого навчання, який використовується для зменшення розмірності в машинному навчанні [69]. Це статистичний процес, який перетворює спостереження корельованих ознак у набір лінійно некорельованих ознак за допомогою ортогонального перетворення. Ці нові трансформовані функції називаються Основними компонентами. Це один із популярних інструментів, який використовується для дослідницького аналізу даних і прогнозованого моделювання. Це техніка для виведення сильних закономірностей із заданого набору даних шляхом зменшення відхилень.

PCA зазвичай намагається знайти поверхню нижчої розмірності для проектування даних великої розмірності [69].

PCA працює, враховуючи дисперсію кожного атрибута, оскільки високий атрибут показує хороший розподіл між класами, а отже, він зменшує розмірність. Деякі реальні застосування PCA – це обробка зображень, система рекомендацій фільмів, оптимізація розподілу потужності в різних каналах зв’язку [69]. Це метод виділення ознак, тому він містить важливі змінні та відкидає найменш важливу змінну.

Алгоритм PCA базується на деяких математичних концепціях, таких як:

Дисперсія та коваріація

Власні значення та власні фактори

Деякі загальні терміни, що використовуються в алгоритмі PCA:

Розмірність: це кількість ознак або змінних, присутніх у даному наборі даних. Простіше кажучи, це кількість стовпців у наборі даних.

Кореляція: це означає, наскільки сильно дві змінні пов’язані одна з одною. Наприклад, якщо змінюється одна, інша змінна також змінюється. Значення кореляції коливається від -1 до +1. Тут -1 виникає, якщо змінні обернено пропорційні одна одній, а +1 вказує на те, що змінні прямо пропорційні одна одній.

Ортогональний: він визначає, що змінні не корельовані одна з одною, і, отже, кореляція між парою змінних дорівнює нулю.

Власні вектори: якщо існує квадратна матриця M і дано ненульовий вектор v. Тоді v буде власним вектором, якщо Av є скалярним кратним v.

Коваріаційна матриця: матриця, що містить коваріацію між парою змінних, називається коваріаційною матрицею [79, 88].

Основні компоненти PCA

Як описано вище, перетворені нові функції або вихід PCA є Основними компонентами. Кількість цих комп’ютерів дорівнює або менше оригінальних функцій, наявних у наборі даних. Нижче наведено деякі властивості цих основних компонентів:

Головний компонент має бути лінійною комбінацією оригінальних характеристик.

Ці компоненти є ортогональними, тобто кореляція між парою змінних дорівнює нулю.

Важливість кожного компонента зменшується при переході від 1 до n, це означає, що 1 ПК має найбільше значення, а n ПК матиме найменше значення.

Метод вибору критеріїв логічних об'єктів протоколу TCP на основі розширеного скінченного автомата (EFSM) буде використаний для ідентифікації та діагностування шкідливих процесів в мережі [58].

Алгоритм PCA буде використаний для збору даних про трафік мережі і виділення характеристик цього трафіку, таких як розмір пакетів, інтервали між пакетами, підсумковий обсяг даних тощо.

Для виявлення шкідливих процесів в протоколі TCP використовуються різні критерії, такі як кількість спроб з'єднання, час очікування відповіді, зміна розміру пакетів тощо.

Метод вибору критеріїв заснований на аналізі EFSM та відповідних станів, які пов'язані з певними критеріями. Наприклад, якщо під час взаємодії з протоколом TCP виникає помилка з певною ймовірністю, створено стан EFSM в попередньому науковому результаті, який відповідає цій помилці, та встановлюються критерії, які вказують на відповідність цього стану помилці [79, 88].

Формула вибору критеріїв логічного об'єкта щодо виявлення шкідливих процесів на основі алгоритму PCA має вигляд:

$$score = w_1 \times x_1 + w_2 \times x_2 + \ldots + w_n \times x_n \quad (3.1)$$

де:

$score$ — оцінка шкідливості процесу;

$w_1, w_2, \ldots w_n$ — вагові коефіцієнти, що відповідають кожному критерію;

$x_1, x_2, \ldots x_n$ — значення кожного критерію.

У формулі вагові коефіцієнти відображають важливість кожного критерію у виявленні шкідливих процесів. Чим більшу вагу отримує певний критерій, тим більше його вплив на оцінку шкідливості процесу. Значення кожного критерію ($x_1, x_2, \ldots x_n$) можуть бути числовими або бінарними (0 або 1), залежно від природи критерію.

Формула вагових коефіцієнтів за допомогою кореляційного аналізу полягає у визначенні кореляційного коефіцієнта Пірсона між кожним з критеріїв і міткою (або цільовою змінною), а потім визначенні вагового

коефіцієнта для кожного критерію, що враховує його кореляцію з міткою та кореляції з іншими критеріями.

Формула вагових коефіцієнтів:

$$w_i = r_i \times (\sum|r_i| \div \sum|r|) \tag{3.2}$$

де:

w_i — ваговий коефіцієнт для критерію;

r_i — кореляційний коефіцієнт Пірсона між критерієм та міткою;

$\sum|r_i|$ — сума абсолютних значень кореляційних коефіцієнтів для всіх критеріїв;

$\sum|r|$ — сума абсолютних значень кореляційних коефіцієнтів між міткою та всіма критеріями.

Отримані вагові коефіцієнти використовуються для вибору найбільш інформативних критеріїв для PCA методу. Критерії з вищими ваговими коефіцієнтами вважаються більш інформативними та використовуються для аналізу даних [69].

Мітка і критерій можуть бути будь-якими ознаками, які використовуються для класифікації об'єктів (рис. 3.1). Для кожного критерію ми можемо обчислити кореляційний коефіцієнт Пірсона між цим критерієм та міткою.

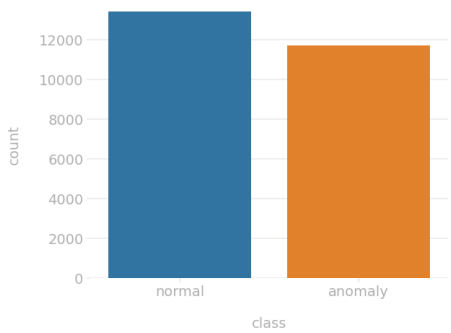

Рис. 3.1. Класифікація даних

Кореляційний коефіцієнт Пірсона - це числове значення, що вказує на силу та напрямок лінійної залежності між двома змінними. В контексті міток та критеріїв, кореляційний коефіцієнт Пірсона може використовуватися для визначення ступеня зв'язку між міткою та критерієм.

Якщо кореляційний коефіцієнт Пірсона дорівнює 1, це означає, що між міткою та критерієм існує дуже сильна позитивна залежність - тобто, чим більше значення критерію, тим більше значення мітки [69, 79, 88].

Якщо кореляційний коефіцієнт Пірсона дорівнює -1, це означає, що між міткою та критерієм існує дуже сильна негативна залежність - тобто, чим менше значення критерію, тим більше значення мітки.

Якщо кореляційний коефіцієнт Пірсона дорівнює 0, це означає, що між міткою та критерієм не існує лінійної залежності.

$$r = \frac{n(\sum xy)-(\sum n)(\sum y)}{\sqrt{[n\sum x^2-(\sum x)^2][n\sum y^2-(\sum y)^2]}} \tag{3.3}$$

де:

r — коефіцієнт кореляції;

n — кількість параметрів;

x — 1-а змінна;

y — 2-а змінна.

При використанні методу PCA для вибору критеріїв логічного об'єкта, спочатку проводиться аналіз різних параметрів (рис. 3.2) і вибираються найбільш інформативні критерії. Далі вагові коефіцієнти можуть бути визначені експертним шляхом.

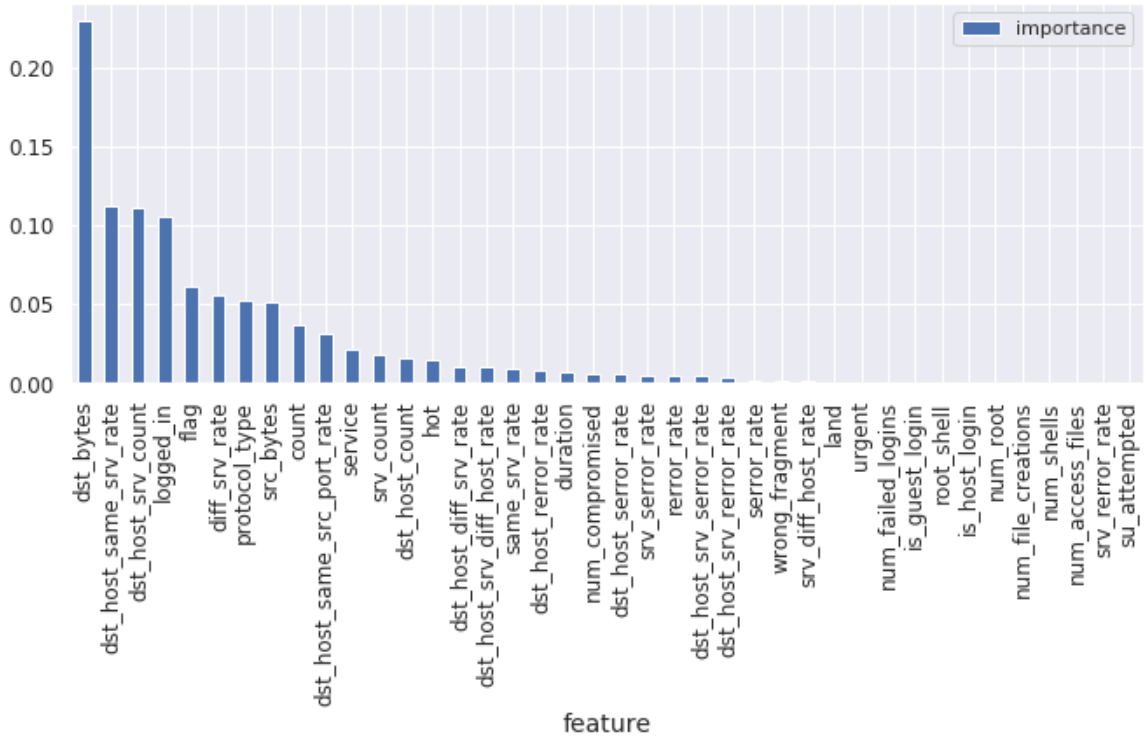

Рис. 3.2. Вибір найважливіших критеріїв

Щоб подати дані для PCA, спочатку необхідно їх проскалювати та вибрати необхідну кількість компонентів для алгоритму (рис. 3.3).

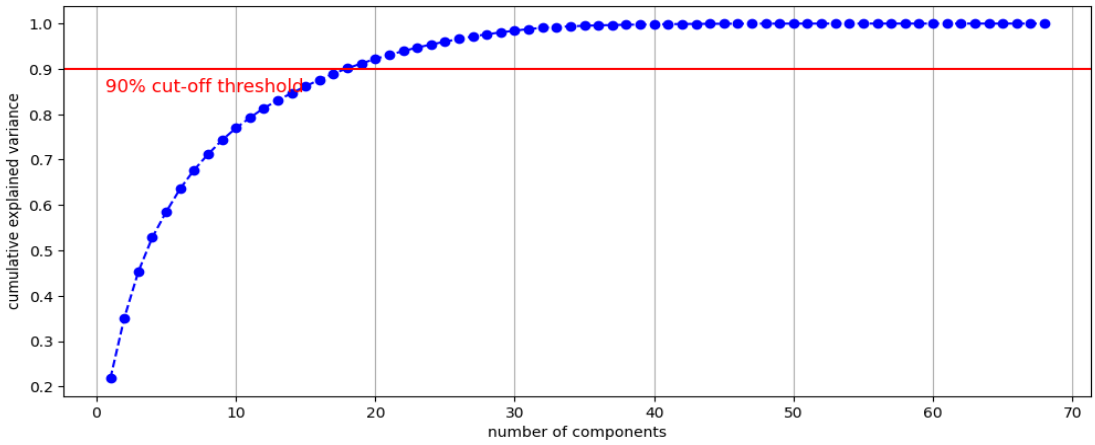

Рис. 3.3. Кількість основних компонентів необхідних для метода PCA

Найбільш інформативні критерії для виявлення ШП за допомогою PCA:

'src_bytes': кількість байтів, що надходять від джерела до призначення.

'dst_bytes': кількість байтів, що надходять від призначення до джерела.

'hot': кількість "хостів" у мережі, які звернулися до цільового сервера за час життя мережевого потоку.

'count': кількість звернень до цільового сервера протягом короткого періоду часу.

'srv_count': кількість звернень до конкретного сервісу на цільовому сервері протягом короткого періоду часу.

'diff_srv_rate': частота різних сервісів, доступних на цільовому сервері, відносно загальної кількості звернень.

'dst_host_count': кількість різних цільових хостів, до яких звертається джерело.

'dst_host_srv_count': кількість сервісів на цільовому хості, до яких звертається джерело.

'dst_host_same_srv_rate': частота звернень до тих же самих сервісів на цільовому хості, відносно загальної кількості звернень.

'dst_host_diff_srv_rate': частота звернень до різних сервісів на цільовому хості, відносно загальної кількості звернень.

'dst_host_same_src_port_rate': частота звернень з тих же самих портів джерела на цільовий хост, відносно загальної кількості звернень.

'dst_host_serror_rate': частота помилок взаємодії на цільовому хості, таких як відмови з'єднань або таймаути.

'protocol_type': тип протоколу мережі, використовуваного у мережевому трафіку, наприклад, TCP, UDP або ICMP.

'service': сервіс, до якого здійснюється доступ у мережевому трафіку, наприклад, http, ftp або ssh.

'flag': стан прапору у заголовку мережевого пакета, наприклад, "SYN", "FIN" або "RST"

Таким чином удосконалений метод вибору критеріїв логічних об’єктів на основі алгоритму PCA, дозволяє прискорити обробку даних при виявленні

шкідливих процесів в реальному часі та заощаджує використання інформаційних ресурсів для навчання методів виявлення.

3.2. Метод виявлення шкідливих процесів в інформаційній системі організації на основі алгоритмів машинного навчання

Для експерименту було використано datasets CIC-DDoS2019 та Stratosphere IPS - Malware Capture Facility Project

(MCFP) (2020, 2021) (DDoS 2019 | Canadian Institute for Cybersecurity | MCFP Dataset (Malware Capture Facility Project - CTU University) [43].

Атака розподіленої відмови в обслуговуванні (DDoS) є загрозою безпеці мережі, яка спрямована на виснаження цільових мереж шкідливим трафіком. Незважаючи на те, що для виявлення атак DDoS розроблено багато статистичних методів, розробка детектора в реальному часі з низькими обчислювальними витратами все ще залишається однією з головних проблем. З іншого боку, оцінка нових алгоритмів і методів виявлення значною мірою залежить від наявності добре розроблених наборів даних.

CICDDoS2019 містить безпечні та найсучасніші поширені DDoS-атаки, які нагадують справжні дані реального часу (PCAP). CICDDoS2019 також містить результати аналізу мережевого трафіку за допомогою CICFlowMeter-V3 із позначеними потоками на основі міток часу, IP-адрес джерела та призначення, портів джерела та призначення, протоколів і атак (файли CSV).

Створення реалістичного фонового трафіку було головним пріоритетом CIC у створенні цього набору даних [43]. Для цього набору даних використано абстрактну поведінку 25 користувачів на основі протоколів HTTP, HTTPS, FTP, SSH і електронної пошти система B-Profile.

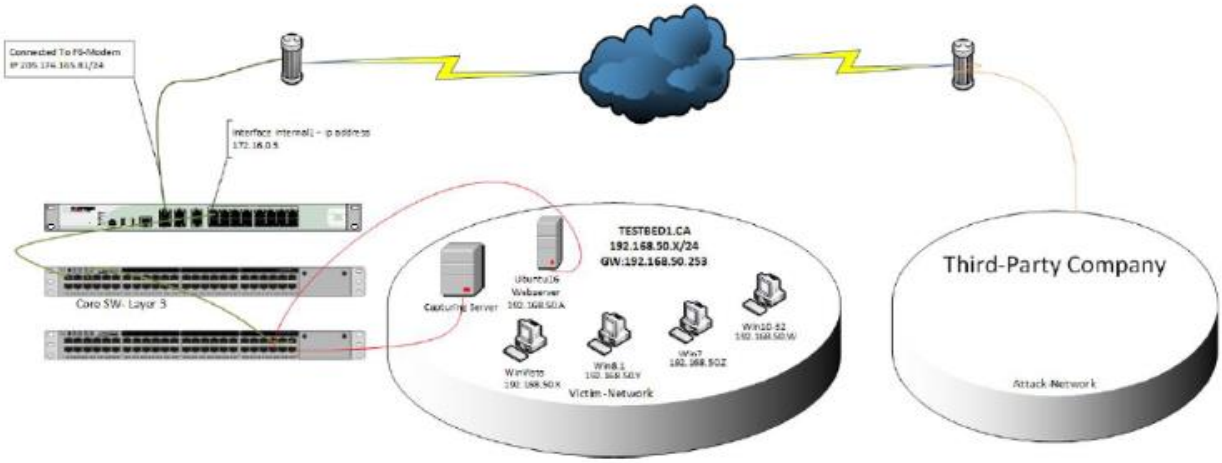

Рис. 3.4. Тестова архітектура CICDDoS2019 [43]

В наборі даних використані різні сучасні рефлексивні DDoS-атаки, такі як PortMap, NetBIOS, LDAP, MSSQL, UDP, UDP-Lag, SYN, NTP, DNS і SNMP. Згодом протягом цього періоду були здійснені напади 12 DDoS-атак, включаючи NTP, DNS, LDAP, MSSQL, NetBIOS, SNMP, SSDP, UDP, UDP-Lag, WebDDoS, SYN і TFTP [43].

Метод опорних векторів (Support Vector Machine Algorithm) або SVM є одним із найпопулярніших алгоритмів керованого навчання, який використовується для задач класифікації та регресії. Однак, в основному, він використовується для проблем класифікації в машинному навчанні.

Мета алгоритму SVM — створити найкращу лінію або межу рішення, яка може розділити n-вимірний простір на класи, щоб ми могли легко помістити нову точку даних у правильну категорію в майбутньому. Ця границя найкращого рішення називається гіперплощиною [32, 86].

SVM вибирає крайні точки/вектори, які допомагають створити гіперплощину. Ці екстремальні випадки називають опорними векторами, а отже, алгоритм називають методом опорних векторів. Розглянемо наведену нижче діаграму, на якій є дві різні категорії, які класифікуються за допомогою границі рішення або гіперплощини:

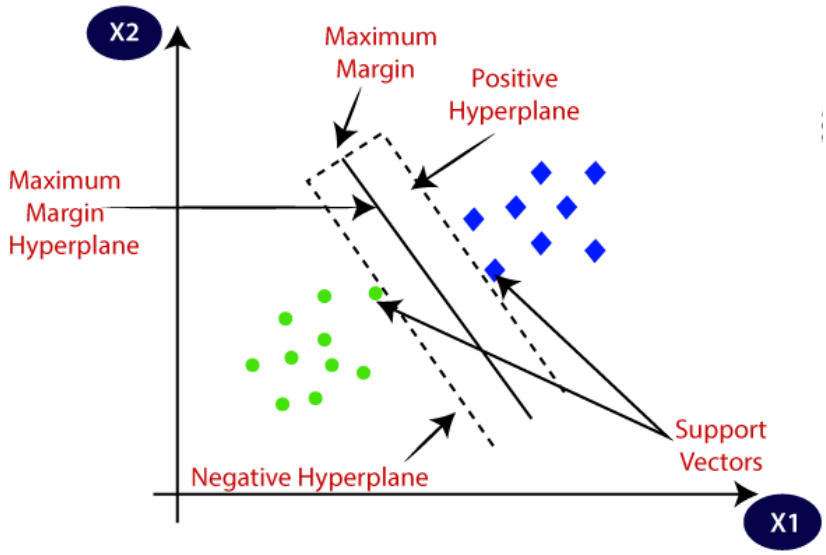

Рис. 3.5. Метод опорних векторів [86]

На практиці алгоритм SVM реалізується за допомогою ядра. Він використовує техніку, яка називається «трюк ядра». Простими словами, ядро — це лише функція, яка відображає дані у вищий вимір, де дані можна розділити. Ядро перетворює низькорозмірний простір вхідних даних у більш вимірний простір. Отже, він перетворює нелінійні сепарабельні задачі на лінійні сепарабельні задачі, додаючи до них більше розмірів. Таким чином, трюк ядра допомагає нам побудувати більш точний класифікатор. Отже, це корисно в задачах нелінійного розділення [32, 86].

Для експерименту використаємо datasets CIC-DDoS2019 та Stratosphere IPS - Malware Capture Facility Project [43, 75, 77] та за допомогою удосконаленого метода PCA на основі CA в яких знаходяться щонайменше 80 критеріїв. Виділяємо найвагоміші для машинного навчання виходячи з HP2.

src_bytes', 'dst_bytes', 'hot', 'count', 'srv_count', 'diff_srv_rate', 'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate', 'dst_host_serror_rate', 'protocol_type', 'service', 'flag'

[6]:

n	protocol_type	service	flag	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot	...	dst_host_srv_count
0	tcp	ftp_data	SF	491	0	0	0	0	0	...	25
0	udp	other	SF	146	0	0	0	0	0	...	1
0	tcp	private	S0	0	0	0	0	0	0	...	26
0	tcp	http	SF	232	8153	0	0	0	0	...	255
0	tcp	http	SF	199	420	0	0	0	0	...	255
...	...	...	...	...	...	...	...	...	...	...	..
0	tcp	exec	RSTO	0	0	0	0	0	0	...	7
0	tcp	ftp_data	SF	334	0	0	0	0	0	...	39
0	tcp	private	REJ	0	0	0	0	0	0	...	13

Рис. 3.5. Вміст датасетів

На основі вищезазначених наукових результатах розподіляємо наший набір критеріїв оцінювання на два датасети тренувальний та тестовий. навчаємо систему виявлення ШП на основі РСА СА TCP протоколу використовуючи різні гіперпараметри *C* та ядра. Отримані результати відображені в табл. 3.1. та 3.2 в яких:

mas — Model accuracy score (оцінка точності моделі) є метрикою, що використовується для оцінки ефективності класифікаційної моделі на наборі даних.

Hyperparameter C — однин з ключових параметрів, який контролює компроміс між мінімізацією помилок класифікації на навчальному наборі та максимізацією ширини розділяючої гіперплощини [85, 86, 91].

Таблиця 3.1.

Отримані значення Train датасету

Train	**hyperparameter C=**					
	1		**100**		**1000**	
Kernel	**t(s)**	**mas**	**t(s)**	**mas**	**t(s)**	**mas**
rbf	0,62	0,9827	1,91	0,9832	7,93	0,9816
linear	0,53	0,983	5,49	0,9832	40,45	0,9832
polinomial	0,85	0,9807	13,82	0,9824		
sigmoid	1,45	0,8858	1,49	0,8855		

Таблиця 3.2.

Перевірка навченої моделі Test датасету

Test	**hyperparameter C=**					
	1		**100**		**1000**	
Kernel	**t(s)**	**mas**	**t(s)**	**mas**	**t(s)**	**mas**
rbf	0,29	0,9827	0,26	0,9832	0,27	0,9816
linear	0,05	0,983	0,05	0,9832	0,05	0,9832
polinomial	0,06	0,9807	0,06	0,9824		
sigmoid	0,24	0,8858	0,24	0,8855		

Значення параметру *C* в SVM визначає міру "жорсткості" або "м'якості" розділяючої гіперплощини. Малий параметр *C* означає більш м'яку розділяючу гіперплощину, що дозволяє деяким прикладам належати до неправильного класу, але зменшує ризик перенавчання моделі.

З іншого боку, великий параметр C означає більш жорстку розділяючу гіперплощину, що зменшує помилки класифікації на навчальному наборі, але може призвести до перенавчання та погіршення універсальності моделі на нових даних [47, 48, 70, 71].

Kernel (ядро) — це функція, яка визначає відстань між точками в просторі ознак. Ця функція дозволяє розширити набір функцій, які можна використовувати для розділення класів, тобто знайти розділяючу гіперплощину, що максимально відокремлює класи.

У SVM, ядра можуть бути лінійними або не лінійними. Лінійне ядро використовується для розділення класів за допомогою простої лінійної розділяючої гіперплощини. Нелінійні ядра дозволяють розділяти класи, що не можуть бути розділені простою лінійною гіперплощиною.

Для використання нелінійних ядер, дані з ознаками спочатку проекціонуються на вищу розмірність (зазвичай більшу), де можна знайти гіперплощину для розділення класів, після чого ці дані повертаються назад до вихідної розмірності. Такі ядра використовуються для розділення класів, що не є лінійно роздільними у вихідному просторі ознак [21, 25, 30, 33-35, 46].

Найбільш поширеними нелінійними ядрами в SVM є радіальне базисне ядро (RBF - Radial Basis Function) та поліноміальне ядро (Polynomial kernel).

Кожне з цих ядер має свої параметри, які можуть бути налаштовані для покращення результатів класифікації. Вибір ядра та його параметрів залежить від характеристик набору даних та завдання класифікації.

Необхідно визначити альтернативні показники, які забезпечують більш точні дані. Однією з таких метрик для аналізу продуктивності моделі проблеми є матриця плутанини.

Матриця плутанини — це інструмент для узагальнення продуктивності алгоритму класифікації. Матриця плутанини дасть нам чітке уявлення про продуктивність моделі класифікації та типи помилок, створених моделлю. Він дає нам зведення правильних і неправильних прогнозів, розбитих за кожною категорією (рис. 3.6).

Під час оцінки продуктивності моделі класифікації можливі чотири типи результатів. Ці чотири результати описані нижче:

Справжні позитивні результати (TP) – Справжні позитивні результати виникають, коли ми передбачаємо, що спостереження належить до певного класу, і спостереження насправді належить до цього класу.

Справжні негативи (TN) – Справжні негативи виникають, коли ми прогнозуємо, що спостереження не належить до певного класу, а спостереження насправді не належить до цього класу [24].

Помилкові позитивні результати (FP) – помилкові позитивні результати виникають, коли ми передбачаємо, що спостереження належить до певного класу, але спостереження насправді не належить до цього класу. Цей тип помилки називається помилкою I типу.

Помилкові негативні результати (FN) – помилкові негативні результати виникають, коли ми передбачаємо, що спостереження не належить до певного класу, але спостереження насправді належить до цього класу. Це дуже серйозна помилка, і вона називається помилкою типу II [24, 57, 95].

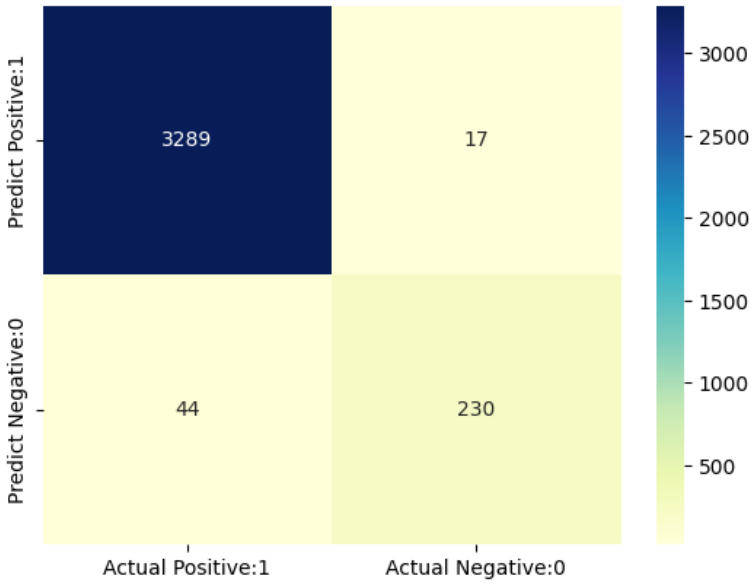

Рис. 3.6. Матриця плутанини

Для більш детально розуміння отриманих результатів розглянемо наші показники в звіті класифікації (Classification report) табл. 3.4 та показники сучасного методу класифікації в табл 3.3.

Таблиця 3.3.

Звіт класифікації сучасного методу

	precision	recall	f1-score	support
0	0,87	0,87	0,87	3306
1	0,83	0,74	0,77	274
accuracy			0,86	3580
macro avg	0,84	0,81	0,83	3580
weighted avg	0,86	0,86	0,86	3580

Таблиця 3.4.

Звіт класифікації удосконаленого методу

	precision	recall	f1-score	support
0	0,99	0,99	0,99	3306
1	0,93	0,84	0,88	274
accuracy			0,98	3580
macro avg	0,96	0,92	0,94	3580
weighted avg	0,98	0,98	0,98	3580

Classification report - це інструмент, який використовується для оцінки якості моделі машинного навчання для багатокласової класифікації в SVM. Цей звіт містить такі параметри:

— Precision (точність) - це відношення кількості правильно класифікованих позитивних прикладів до загальної кількості прикладів, які були класифіковані як позитивні. Точність вказує, наскільки точно модель ідентифікує позитивні приклади.

— Recall (чутливість) - це відношення кількості правильно класифікованих позитивних прикладів до загальної кількості позитивних прикладів в тестовому наборі. Чутливість вказує, наскільки ефективно модель ідентифікує позитивні приклади [24].

— F1-score - це середнє гармонічне між точністю та чутливістю. Це є мірою загальної ефективності моделі в класифікації.

— Support - це кількість прикладів у кожному класі, яку використовується для обчислення precision, recall та F1-score.

— 0 та 1 відображають два класи, які були задані під час тренування моделі. Клас 0 відповідає негативному класу (тобто, відсутність певної ознаки або події), а клас 1 - позитивному класу (тобто, наявність певної ознаки або події).

Precision, recall та F1-score для кожного класу розраховуються окремо. Precision - це відношення кількості правильно класифікованих елементів до загальної кількості елементів, що були визначені як позитивний клас. Recall - це відношення кількості правильно класифікованих елементів до загальної кількості елементів, які насправді належать до позитивного класу. F1-score - це середнє гармонійне між precision та recall.

Таким чином, показники precision, recall та F1-score для класу 0 відображають ефективність моделі в правильному визначенні негативних елементів, а показники для класу 1 - ефективність моделі в правильному визначенні позитивних елементів. Взагалі, чим більше ці показники для обох класів, тим краще працює модель [24, 46-48, 57, 95].

— Accuracy (точність) - це відношення кількості правильно класифікованих прикладів до загальної кількості прикладів в тестовому наборі.

— Macro average - це середнє арифметичне precision, recall та F1-score для кожного класу. Це означає, що кожен клас має однакову вагу для підсумовування показників, незалежно від кількості прикладів у кожному класі. Macro average корисний в тих випадках, коли кількість прикладів у кожному класі досить збалансована.

— Weighted average - це середнє арифметичне precision, recall та F1-score для кожного класу з урахуванням кількості прикладів у кожному класі. Це означає, що показники для більш великих класів мають більшу вагу, оскільки вони мають більше прикладів. Weighted average корисний в тих випадках, коли кількість прикладів у кожному класі не збалансована.

Після оформлення звіту отримуємо точність даних класифікацій табл. 3.5 для сучасного методу та табл. 3.6 для удосконаленого методу.
Отримавши точність класифікацій методів створемо порівняльна таблиця точності класифікації сучасного методу та удосконаленого (табл. 3.7).

Точність класифікації (Classification accuracy) - це відношення кількості правильно класифікованих елементів (TP) до загальної кількості елементів у вибірці (TP + FP). Цей показник дозволяє оцінити загальну ефективність моделі.

Таблиця 3.5.

Точність класифікації сучасного методу

Точність класифікації:	0,865
Класифікаційна помилка :	0,135
Точність:	0,8758
Пригадування або чутливість:	0,8681
Істино позитивний рейтинг :	0,8681
Хибнопозитивний рейтинг:	0,0605
Специфікація:	0,8206

Таблиця 3.6.

Точність класифікації удосконаленого методу

Точність класифікації:	0,984
Класифікаційна помилка :	0,016
Точність:	0,9948
Пригадування або чутливість:	0,9869
Істино позитивний рейтинг :	0,9869
Хибнопозитивний рейтинг:	0,0687
Специфікація:	0,9312

Класифікаційна помилка (Classification error) - це відношення кількості неправильно класифікованих елементів до загальної кількості елементів у вибірці. Цей показник відображає кількість помилок, які робить модель.

Точність (Precision) — це відношення кількості правильно класифікованих позитивних елементів до загальної кількості елементів, які були визначені як позитивний клас. Цей показник дозволяє оцінити точність класифікації позитивних елементів. Математично точність визначається як відношення TP до (TP + FP) [24].

Пригадування або чутливість (Recall або Sensitivity) - це відношення кількості правильно класифікованих позитивних елементів (TP) до загальної кількості елементів, які насправді належать до позитивного класу (TP + FN). Цей показник відображає здатність моделі виявляти позитивні елементи.

Істинно позитивний рейтинг (True Positive Rate) - це відношення кількості правильно класифікованих позитивних елементів до загальної кількості елементів, які насправді належать до позитивного класу. Цей показник є аналогічним до показника recall або sensitivity.

Хибнопозитивний рейтинг (False Positive Rate) - це відношення кількості неправильно класифікованих негативних елементів до загальної кількості елементів, які насправді належать до негативного класу. Цей показник відображає частоту помилкової класифікації негативних елементів як позитивних.

Специфікація (Specificity) - це зважене гармонічне середнє значення точності та запам’ятовування. Найкраща можлива оцінка f1 буде 1,0, а найгірша – 0,0. f1-оцінка є середнім гармонійним значенням точності та запам’ятовування. Таким чином, показник f1 завжди нижчий, ніж вимірювання точності, оскільки вони вбудовують точність і пам’ять у свої обчислення. Для порівняння моделей класифікаторів слід використовувати середнє зважене значення f1, а не глобальну точність.

Таблиця 3.7

Порівняльна таблиця точності класифікації сучасного методу та удосконаленого

	SVM	Удосконал.	Відсоток покращення
Точність класифікації:	0,865	0,984	13,76
Класифікаційна помилка :	0,135	0,016	88,15
Точність:	0,8758	0,9948	13,59
Пригадування або чутливість:	0,8681	0,9869	13,69
Істино позитивний рейтинг :	0,8681	0,9869	13,69
Хибнопозитивний рейтинг:	0,0605	0,0687	13,55
Специфікація:	0,8206	0,9312	13,48

Отже, удосконалений метод машинного навчання щодо застосування алгоритмів, які використовуються для задач класифікації на основі обраних параметрів за допомогою розширеного скінченного автомату транспортного протоколу TCP, що дозволяє підвищити ефективність виявлення шкідливих процесів в інформаційній системі підприємства зменшуючи хибно-позитивні спрацювання на 13%. Тобто, точність удосконаленого методу збільшилася на 13-14% в порівнянні із сучасними методами на основі машинного навчання.

3.3. Дослідження ефективності та достовірності застосування методу виявлення шкідливих процесів в інформаційній системі підприємства

Для перевірки запропонованих в дисертаційному дослідженні підходів щодо виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об’єктів проведено практичну реалізацію на підприємстві, зміст якої полягає у наступному. У відповідності до розробленого методу виявлення ШП було розроблено наступну методологію яка наведена на рис. 3.6. В якості даних для перевірки та оцінки ефективності розробленого методу було прийнято рішення використовувати вже розроблений в 3.2 тестовий стенд (див. рис 3.3).

Експериментальне дослідження та оцінка ефективності розробленого методу виявлення ШП в ІС підприємства на основі ідентифікації та діагностування станів логічних об’єктів проводилися з використанням мови програмування Python в середовищі jupyter-lab використовуючи метод вибору головних компонент PCA та машинного навчання SVM на основі СА TCP протоколу [21-95].

Загалом, під ефективністю виявлення ШП в ІС підприємства розуміється швидкість виявлення ШП на СА протоколу TCP та ідентифікація і діагностування станів логічних об’єктів в режимі реального часу.

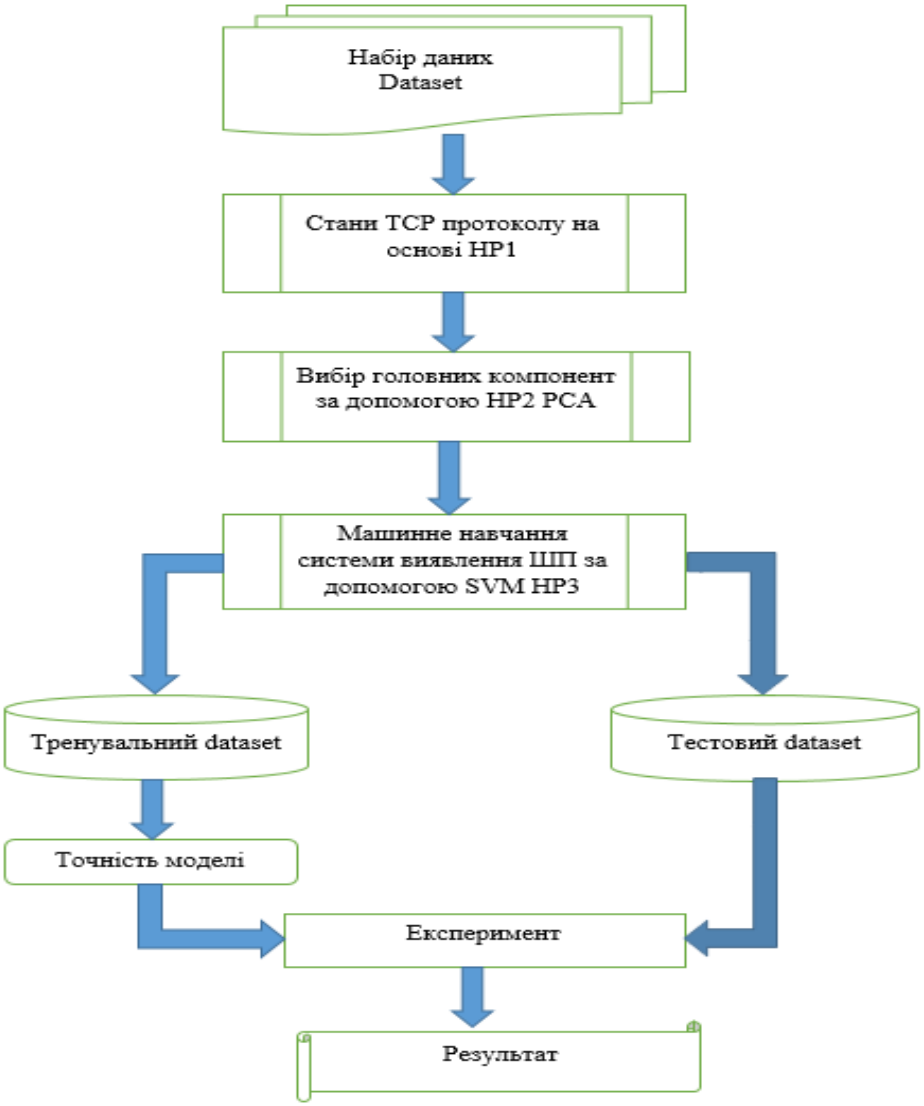

Рис. 3.6. Методологія дослідження ефективності наукових результатів

1-й етап: вибір набору даних для дослідження ефективності.

2-й етап: визначення станів логічного об’єкта за допомогою розширеної матриці функціонування TCP протоколу на основі скінченного автомата (НР1).

3-й етап: вибір основних компонентів за допомогою методу PCA, що удосконалений на основі СА TCP протоколу та зменшення критерії для машинного навчання (НР2).

4-й етап: використання SVM для навчання системи виявлення шкідливих процесів, що базується на попередніх наукових результатів (НР3).

5-й етап: розподілення набору даних на тренувальний та тестові dataset-и.

6-й етап: визначення точності моделі навченої системи виявлення ШП НР3).

7-й етап: проведення експерименту.

9-й етап: відображення результату дослідженої ефективності.

Використовуючи дані наукових результатів дисертаційного дослідження та вже навченої моделі виявлення шкідливих процесів переглянемо результати проведеного експерименту на тестовому стенду продемонстрованому в п. 3.2 та порівняємо з удосконаленим методом виявлення ШП ІС підприємства на основі ідентифікації та діагностування станів логічних об’єктів. Отримані результати відображаємо в таблицях 3.8 та 3.9. Далі знаходимо відсоткове значення на скільки удосконалений метод краще за сучасний SVM, дані заносимо до табл. 3.10.

Таблиця 3.8.

Отримані результати методом SVM

Test1	**hyperparameter C=**		
	1	**100**	**1000**
rbf	0,72	2,01	8,95
linear	0,63	5,65	41,55

Таблиця 3.9

Отримані результати удосконаленого методу

Test2	**hyperparameter C=**		
	1	**100**	**1000**
rbf	0,25	0,25	0,26
linear	0,07	0,07	0,06

Графічне представлення отриманих результатів на рис. 3.7 де по основній осі графік отриманих результатів SVM з основним діленням в 10 с. та на додатковій осі розробленого методу виявлення ШП з діленням в 0,5 с.

В таблиці 3.7 відображено в скільки разів зменшився час виявлення ШП на основі методу розробленого в даному дисертаційному дослідженні.

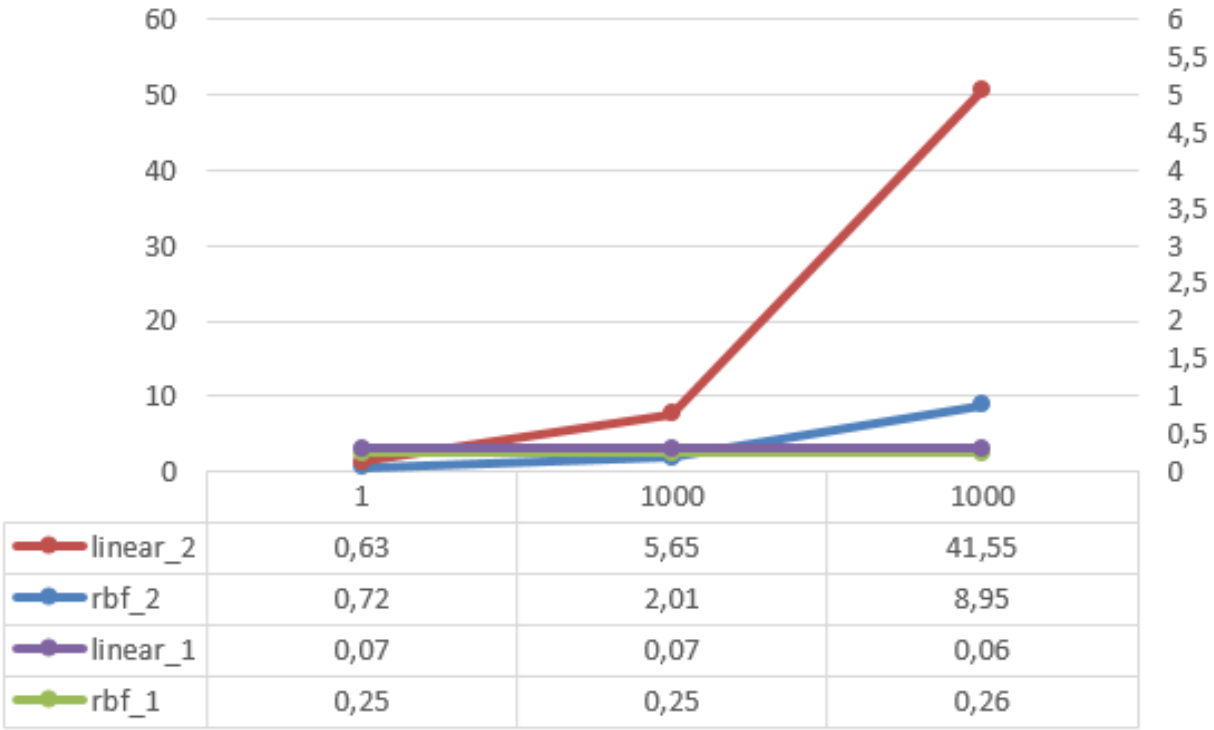

Рис. 3.7. Порівняльний графік отриманих результатів

Таблиця 3.10.

Ефективність виявлення (в часі) розробленого методу за звичайний SVM

	hyperparameter C=		
	1	**100**	**1000**
rbf	65,3%	87,6%	97,1%
linear	88,9%	98,8%	99,86%

В результаті дослідження порівнявши звичайний метод SVM з удосконаленим методом в ході роботи швидкість виявлення ШП зменшилася в:

1. 65,3% з гіперпараметром *C=1* та використання ядра *rbf*
2. 87,6% з гіперпараметром *C=100* та використання ядра *rbf*
3. 97,1% з гіперпараметром *C=1000* та використання ядра *rbf*
4. 88.9% з гіперпараметром *C=1* та використання ядра *linear*
5. 98,8% з гіперпараметром *C=100* та використання ядра *linear*

6. 99,86% з гіперпараметром *C=1000* та використання ядра *linear*

Отже, на основі проведеного дослідження можна зробити висновок, що вдосконалений метод виявлення шкідливих процесів на основі алгоритму методу опорних векторів з використанням лінійного ядра та гіперпараметром C=1000 є найефективнішим. Застосування цього методу дозволяє значно знизити час виявлення шкідливих процесів, що сприяє підвищенню рівня безпеки інформаційної систем. Результати дослідження можуть бути використані в подальших наукових дослідженнях в галузі захисту інформації та забезпечення безпеки технічних систем.

3.4. Розробка рекомендацій щодо впровадження методу виявлення шкідливих процесів в інформаційній системі підприємства

На основі результатів дослідження ефективності та достовірності застосування методу виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів використовуючи скінченні автомати TCP протоколу вибору головних компонент PCA та машинного навчання SVM, можна рекомендувати впровадження цього методу в інформаційну систему підприємства після мережевого екрану рис. 3.8.

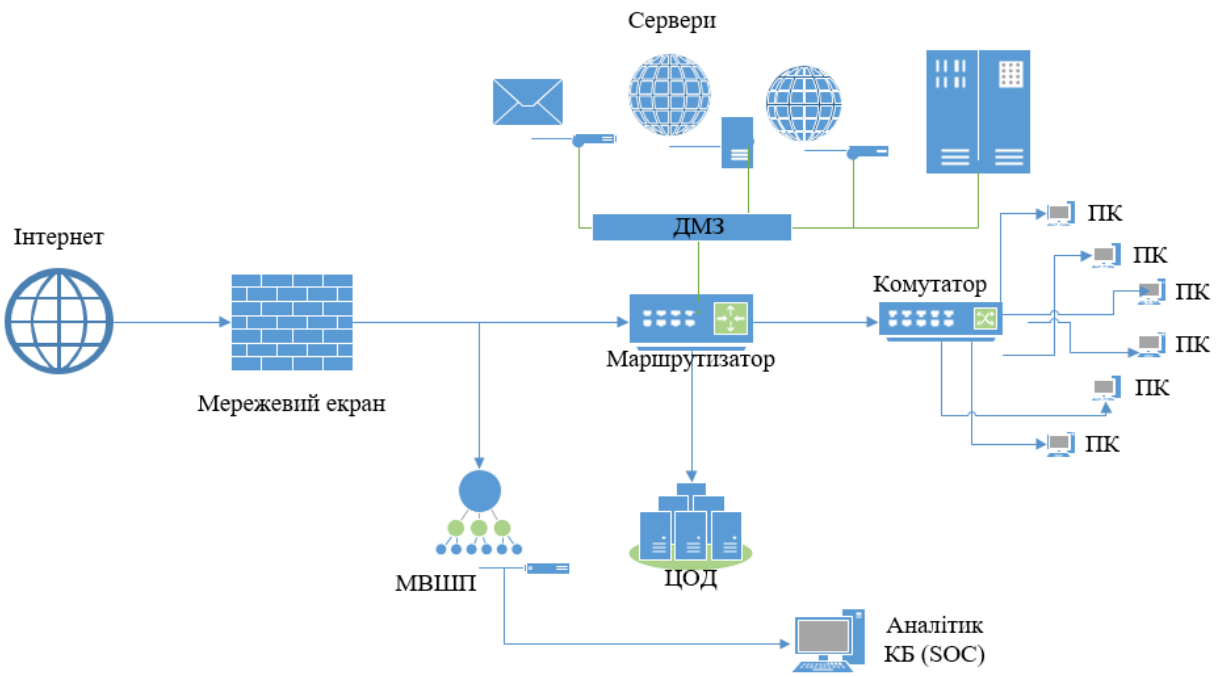

Рис. 3.8. Розташування методу виявлення ШП в ІС підприємства

Опис роботи алгоритму відповідно методу виявлення шкідливих процесів (МВШП) в інформаційній системі підприємства зображений на рис. 3.9.

1. Діагностування стану ТСР: На початку роботи методу необхідно діагностувати стан технічної системи розумінням її функціональності та взаємодії логічних об'єктів в межах системи.

2. Збір даних: Після діагностики стану ТСР, метод збирає велику кількість даних про функціонування логічних об'єктів, які входять до складу інформаційної системи підприємства.

3. Ідентифікація стану ТСР: Зібрані дані аналізуються для виявлення потенційних загроз, що можуть виникнути в інформаційній системі.

4. Вибір основних компонент: На цьому етапі метод використовує алгоритми аналізу основних компонентів для вибору ключових факторів, які найбільше впливають на функціонування системи.

5. Метод опорних векторів: Вибрані ключові фактори подаються на вхід до методу опорних векторів, який використовується для побудови моделі, що дозволяє виявляти відхилення від нормального функціонування.

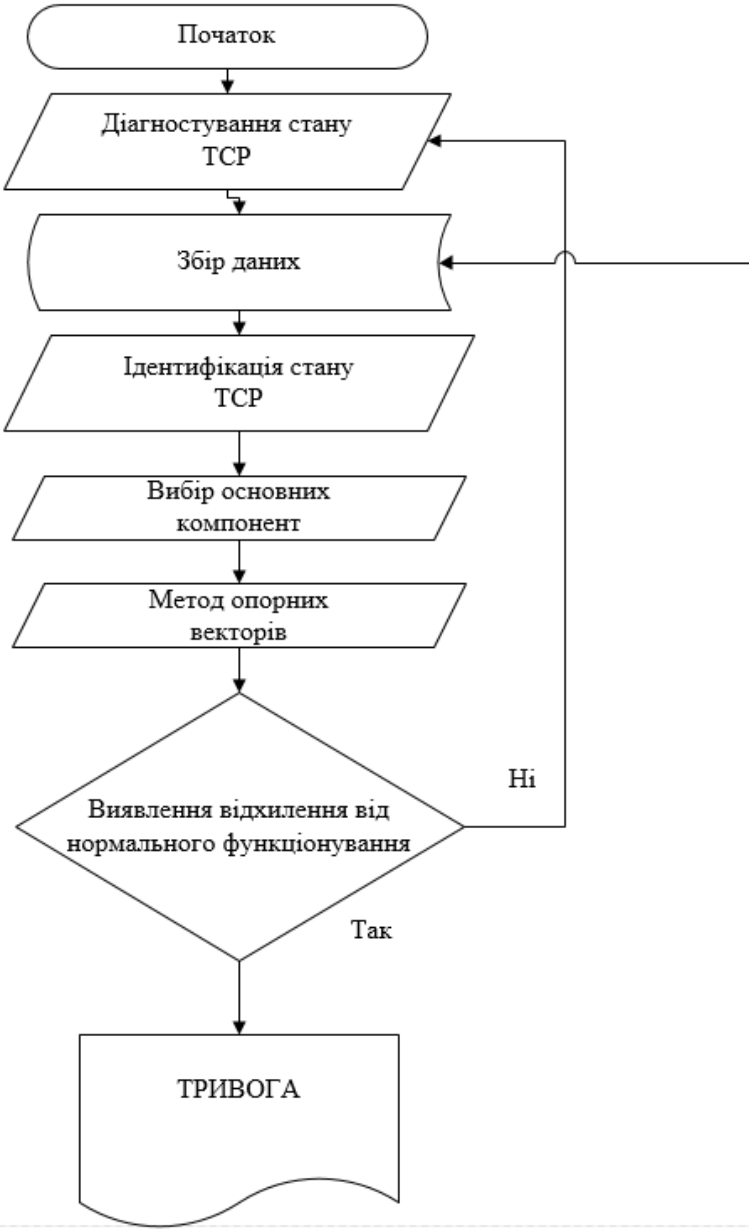

Рис. 3.9. Алгоритм роботи методу виявлення шкідливих процесів

6. Виявлення відхилення від нормального функціонування: на цьому етапі проводиться аналіз отриманих даних і визначення, чи є відхилення від нормального функціонування інформаційної системи. Для цього застосовується метод опорних векторів. Метод аналізує нові дані про

функціонування системи та порівнює їх з результатами, отриманими під час тренування моделі. Якщо відхилення від нормального функціонування буде виявлено, метод спрацьовує і генерує тривогу.

7. Виявлення тривоги: якщо на етапі виявлення відхилення від нормального функціонування було виявлено потенційну загрозу безпеці інформаційної системи, то на цьому етапі генерується тривога. Тривога може бути надіслана адміністратору системи або відповідному відділу, щоб вжити заходів щодо подальшого аналізу інциденту та його припинення.

Якщо на етапі виявлення відхилення від нормального функціонування відсутні будь-які ознаки потенційних атак або інших шкідливих процесів, то процес повертається до початку, і цикл збору та аналізу даних розпочинається знову.

Таким чином, алгоритм методу виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів дозволяє автоматично виявляти потенційні загрози безпеці інформаційної системи та реагувати на них вчасно. Це значно зменшує ризик виникнення інцидентів та збільшує рівень безпеки інформаційних систем підприємства та її функціонування в цілому.

Загальні рекомендації щодо впровадження методу виявлення шкідливих процесів:

1. Забезпечити належне збирання та аналіз даних для побудови моделі ідентифікації та діагностування шкідливих процесів.

2. Використовувати надійні алгоритми для обробки даних, щоб уникнути помилок та впевнитись у точності результатів.

3. Ретельно відібрати набір ознак для побудови моделі, щоб уникнути перенавчання та забезпечити її ефективність.

4. Використовувати стандарти та протоколи для забезпечення взаємодії між системами та компонентами.

5. Забезпечити постійний моніторинг за станом системи та її компонентів.

6. Регулярно проводити навчання моделі на нових даних для забезпечення її актуальності та ефективності.

7. Налагодити процедуру автоматичного оповіщення у разі виявлення шкідливих процесів для швидкого реагування та мінімізації можливих наслідків.

8. Врахувати особливості конкретної інформаційної системи та її використання при впровадженні методу ідентифікації та діагностування шкідливих процесів.

9. Забезпечити відповідний рівень захисту даних та конфіденційності, пов'язаний з використанням методу ідентифікації та діагностування шкідливих процесів.

10. Проводити систематичний аналіз ефективності та результативності методу для постійного вдосконалення та оптимізації його використання.

Однак, перед впровадженням методу слід врахувати наступні рекомендації:

для впровадження методу необхідно підготувати відповідний датасет, що містить достатню кількість відомостей про стани логічних об'єктів. Якщо немає достатньої кількості даних, то слід провести додаткові дослідження та зібрати необхідну інформацію;

слід забезпечити достатній рівень організаційної культури, що дозволяє працівникам використовувати систему ефективно та безпечно;

необхідно провести тестування системи перед її впровадженням, щоб переконатися в її ефективності та достовірності в конкретному підприємстві;

слід забезпечити можливість редагування параметрів системи в разі необхідності їх зміни;

необхідно забезпечити інтеграцію системи в інформаційну інфраструктуру підприємства та її взаємодію з іншими системами;

слід забезпечити навчання працівників підприємства з використання системи та її можливостями;

необхідно забезпечити регулярне оновлення системи та її параметрів в залежності від змін у логіці роботи підприємства та можливих нових загроз безпеці;

перед впровадженням методу слід провести аналіз вартості розробки та впровадження системи. Може виявитися, що вартість розробки та впровадження системи перевищує потенційні користі від її використання;

слід забезпечити захист від можливих атак на систему виявлення шкідливих процесів. Для цього можна використовувати захисні механізми, такі як шифрування даних, автентифікацію, контроль доступу тощо;

для впровадження методу слід мати достатній рівень технічної підтримки, яка забезпечує вчасне виявлення та усунення проблем у системі.

Висновки до розділу 3

1. Удосконалено метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі методу основних компонентів з усіх параметрів вхідних даних, що на відміну від існуючих, визначає найбільш важливі параметри логічного об’єкту моделі розширеного скінченного автомату TCP протоколу. Такий підхід дозволяє прискорити обробку даних при виявленні шкідливих процесів в інформаційній системі організації в реальному часі.

2. Удосконалено метод виявлення шкідливих процесів в інформаційній системі організації, який на відміну від існуючих, базується на алгоритмах машинного навчання та враховує обрані параметри які підлягають ідентифікації та діагностуванню. Такий метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі.

3. У результаті дослідження було встановлено, що запропонований метод метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі на 65-99% (в залежності від використаного ядра та гіперпараметру *C*) від існуючих сучасних методів та зменшення кількості хибно-позитивних спрацювань на 13-14%.

4. Розроблено рекомендації щодо впровадження методу виявлення шкідливих процесів в інформаційну систему підприємства та описаний алгоритм роботи розробленого методу.

Таким чином, розроблений та досліджений метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностуванні станів логічних об’єктів дозволяє швидше знаходити шкідливі процеси в режимі реального часу та використовувати менше ресурсів для ідентифікації та діагностування станів логічних об’єктів, що може позитивно відобразитися на діяльності підприємства в цілому.

ВИСНОВКИ

В результаті дисертаційних досліджень, виконаних автором, вирішено актуальне наукове завдання, що полягає в розробці методу виявлення шкідливих процесів в реальному часі на основі ідентифікації та діагностування станів логічних об’єктів інформаційної системи підприємства. Зазначене наукове завдання має суттєве значення для забезпечення безпеки інформаційної системи підприємства від шкідливих процесів, а також зменшення хибно-позитивних спрацювань в системах виявлення вторгнень. Відсутність аналогічних методів у світі робить результати роботи дослідження актульними та приорітетними.

В дисертаційному дослідженні отримані такі основні результати:

1.На підставі проведеного аналізу встановлено, що сучасний етап економічного та технологічного розвитку більшості країн світу супроводжується стрімким розвитком інформаційно-комунікаційних технологій, на які націлені основні атаки та шкідливі процеси для виведення її з ладу, спотворення інформації та несанкціонованого доступу то даних. Будь-який шкідливий процес може бути направлений на інформаційну систему підприємства. Тому існує необхідність в створенні надійних систем захисту, а також методів виявлення та реагування на шкідливі процеси в реальному часі. В результаті аналізу існуючих методів виявлення шкідливих процесів, було визначено, що існує необхідність в розробці методів виявлення шкідливих процесів в реальному часі. Основне протиріччя, яке лежить в основі наукового дослідження полягає в тому, що існує необхідність оперативного ідентифікування в режимі реального часу шкідливих процесів та якісного виявлення зі зменшенням показника хибно-позитивних спрацювань. Отже, актуальним є наукове завдання, щодо розробки методу виявлення шкідливих процесів в реальному часі на основі ідентифікації та діагностування станів логічних об’єктів інформаційної системи підприємства.

2. Вперше розроблено модель ідентифікації та діагностування станів логічних об’єктів, наукова новизна якого полягає в тому, що вона ґрунтується на теорії скінченних автоматів на основі протоколу TCP, що дозволяє на стадії створення з’єднання ідентифікувати порушення пакету зі встановленими параметрами та діагностувати роботу транспортного протоколу.

3. Удосконалено метод вибору критеріїв логічних об’єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі методу основних компонентів з усіх параметрів вхідних даних, що на відміну від існуючих, визначає найбільш важливі параметри логічного об’єкту моделі розширеного скінченного автомату TCP протоколу. Такий підхід дозволяє прискорити обробку даних при виявленні шкідливих процесів в інформаційній системі організації в реальному часі.

4. Удосконалено метод виявлення шкідливих процесів в інформаційній системі організації, який на відміну від існуючих, базується на алгоритмах машинного навчання та враховує обрані параметри які підлягають ідентифікації та діагностуванню. Такий метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об’єктів в інформаційній системі організації в реальному часі.

5. У результаті дослідження та проведення практичного експерименту щодо виявлення шкідливих процесів в інформаційній системі підприємства було встановлено, що запропонований метод є ефективним, щодо роботи в режимі реального часу на 65-99% (в залежності від використаного ядра та гіперпараметру C) від існуючих сучасних методів та зменшення кількості хибно-позитивних спрацювань на 13-14%.

6. Розроблено рекомендації щодо впровадження методу виявлення шкідливих процесів в інформаційну систему підприємства та описаний алгоритм роботи розробленого методу.

7. Таким чином, мета дослідження яка полягає у підвищенні ефективності роботи інформаційної системи підприємства за рахунок зменшення часу на пошук та вирішення проблем, що виникають у процесі її роботи шляхом

вчасного виявлення можливих проблем та шкідливих процесів, що можуть впливати на її функціонування на основі ідентифікації та діагностування станів логічних об’єктів досягнута та всі часткові завдання вирішено повністю. Наукові результати є внеском у розвиток науково-методичних основ, технологій та засобів виявлення шкідливих процесів в інформаційні системи підприємств.

8. Основні наукові результати дослідження реалізовані у практичній діяльності в ТОВ «Смартс» та в ТОВ «ЄВРОТЕЛЕКОМ», які використані для удосконалення програмного забезпечення систем моніторингу та виявлення шкідливих процесів та в освітній процес Державного університету телекомунікацій.

9. Напрями подальших досліджень в зазначеній галузі можуть ґрунтуватися на більш глибокому вивчені та на вдосконаленні методів машинного навчання, що використовуються для виявлення шкідливих процесів в інформаційних системах підприємств.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abu-Ein, A. A.-K., Hatamleh, H., & Sharadqeh, A. A. M. (2022, October). *Using finite state machine at the testing of network protocols*. ResearchGate. Retrieved 2022, from https://www.researchgate.net/publication/294553580_Using_finite_state_machine_at_the_testing_of_network_protocols.

2. Abraham, A., & Thomas, J. (2006). Distributed intrusion detection systems. *Applications of Information Systems to Homeland Security and Defense*, 107–137. https://doi.org/10.4018/978-1-59140-640-2.ch005.

3. Ageyev, D., Radivilova, T., Mulesa, O., Bondarenko, O., & Mohammed, O. (2022). Traffic monitoring and abnormality detection methods for decentralized distributed networks. *Information Security Technologies in the Decentralized Distributed Networks*, 287–305. https://doi.org/10.1007/978-3-030-95161-0_13.

4. Ahmadi, M., Ulyanov, D., Semenov, S., Trofimov, M., & Giacinto, G. (2016). Novel feature extraction, selection and fusion for effective malware family classification. *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*. https://doi.org/10.1145/2857705.2857713.

5. Ahmed, A. A. E., & Traore, I. (2005). Anomaly intrusion detection based on biometrics. *Proceedings from the Sixth Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop, 2005.* https://doi.org/10.1109/iaw.2005.1495997.

6. Ahmed, M., Naser Mahmood, A., & Hu, J. (2016). A survey of Network Anomaly Detection Techniques. *Journal of Network and Computer Applications*, *60*, 19–31. https://doi.org/10.1016/j.jnca.2015.11.016.

7. Akgun, D., Hizal, S., & Cavusoglu, U. (2022). A new ddos attacks intrusion detection model based on Deep Learning for Cybersecurity. *Computers & Security*, *118*, 102748. https://doi.org/10.1016/j.cose.2022.102748.

8. AL-Hawawreh, M., Moustafa, N., & Sitnikova, E. (2018). Identification of malicious activities in industrial internet of things based on Deep Learning Models. *Journal of Information Security and Applications*, *41*, 1–11. https://doi.org/10.1016/j.jisa.2018.05.002.

9. Aljeaid, D., Ma, X., & Langensiepen, C. (2015). Analysis of security protocols using finite-state machines. *International Journal of Advanced Research in Artificial Intelligence*, *4*(4). https://doi.org/10.14569/ijarai.2015.040407.

10. Allman, M., Paxson, V., & Stevens, W. (1999, April 1). *TCP Congestion Control*. RFC Editor. Retrieved January 12, 2022, from https://www.rfc-editor.org/rfc/rfc2581.

11. Ates, C., Ozdel, S., & Anarim, E. (2019). A new network anomaly detection method based on header information using greedy algorithm. *2019 6th International Conference on Control, Decision and Information Technologies (CoDIT)*. https://doi.org/10.1109/codit.2019.8820501.

12. Ates, C., Ozdel, S., & Anarim, E. (2019). Clustering based DDoS attack detection using the relationship between packet headers. *2019 Innovations in Intelligent Systems and Applications Conference (ASYU)*. https://doi.org/10.1109/asyu48272.2019.8946331.

13. Ates, Ç., Özdel, S., & Anarim, E. (2020). Graph-based fuzzy approach against ddos ATTACKS1. *Journal of Intelligent & Fuzzy Systems*, *39*(5), 6315–6324. https://doi.org/10.3233/jifs-189099.

14. Baranov, S. I. (2018). *Finite State Machines and algorithmic state machines: Fast and simple design of complex finite state machines*. Wydawca nieznany.

15. Barry, B. I., & Chan, H. A. (2007). A cross-protocol approach to detect TCP hijacking attacks. *2007 IEEE International Conference on Signal Processing and Communications*. https://doi.org/10.1109/icspc.2007.4728254.

16. Bekerman, D., Shapira, B., Rokach, L., & Bar, A. (2015). Unknown malware detection using network traffic classification. *2015 IEEE Conference on*

Communications and Network Security (CNS). https://doi.org/10.1109/cns.2015.7346821.

17. Bhatkar, S., Chaturvedi, A., & Sekar, R. (2006). Dataflow anomaly detection. *2006 IEEE Symposium on Security and Privacy (S&P'06)*. https://doi.org/10.1109/sp.2006.12.

18. Bokr, J., & Jáneš, V. (2006). State of a logical object. *Acta Polytechnica*, *46*(1). https://doi.org/10.14311/808.

19. Bulakh, V., Radivilova, T., & Kirichenko, L. (2018). Classification of multifractal time series by decision tree methods. In *14th International Conference on ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer. CEUR Workshop Proceedings* (Vol. 2005, pp. 457–460).

20. Cahyo, A. N., Hidayat, R., & Adhipta, D. (2016). Performance comparison of intrusion detection system based anomaly detection using artificial neural network and support vector machine. *AIP Conference Proceedings*. https://doi.org/10.1063/1.4958506.

21. Cao, C., Blaise, A., Verwer, S., & Rebecchi, F. (2022). Learning State Machines to monitor and detect anomalies on a kubernetes cluster. *Proceedings of the 17th International Conference on Availability, Reliability and Security*. https://doi.org/10.1145/3538969.3543810.

22. Cassel, S., Howar, F., Jonsson, B., & Steffen, B. (2014). Learning extended finite state machines. *Software Engineering and Formal Methods*, 250–264. https://doi.org/10.1007/978-3-319-10431-7_18.

23. Chen, L., Ye, Y., & Bourlai, T. (2017). Adversarial machine learning in malware detection: Arms race between evasion attack and Defense. *2017 European Intelligence and Security Informatics Conference (EISIC)*. https://doi.org/10.1109/eisic.2017.21.

24. *Confusion matrix in machine learning*. javatpoint. (2021). Retrieved 2022, from https://www.javatpoint.com/confusion-matrix-in-machine-learning.

25. Demetrio, L., Biggio, B., Lagorio, G., Roli, F., & Armando, A. (2019, January 24). *Explaining vulnerabilities of deep learning to adversarial malware binaries*. arXiv.org. Retrieved October 9, 2022, from https://arxiv.org/abs/1901.03583.

26. Dorokhin, O. O., Marchenko, V. V., & Semenova, I. D. (2021). Threat Intelligence Technology and methods of its use to protect the company from Cyber Threats. Modern Information Security, 3(47). https://doi.org/10.31673/2409-7292.2021.031216.

27. El-Fakih, K., Yevtushenko, N., Bozga, M., & Bensalem, S. (2016). Distinguishing extended finite state machine configurations using predicate abstraction. *Journal of Software Engineering Research and Development*, *4*(1). https://doi.org/10.1186/s40411-016-0027-4.

28. Erhan, D., & Anarım, E. (2020). Boğaziçi University Distributed Denial of Service Dataset. *Data in Brief*, *32*, 106187. https://doi.org/10.1016/j.dib.2020.106187.

29. Erhan, D., & Anarim, E. (2020). Hybrid ddos detection framework using matching pursuit algorithm. *IEEE Access*, *8*, 118912–118923. https://doi.org/10.1109/access.2020.3005781.

30. Fraley, J. B., & Cannady, J. (2017). The promise of machine learning in Cybersecurity. *SoutheastCon 2017*. https://doi.org/10.1109/secon.2017.7925283.

31. GeeksforGeeks. (2022, May 12). *TCP connection termination*. GeeksforGeeks. Retrieved January 14, 2023, from https://www.geeksforgeeks.org/tcp-connection-termination/?ref=rp.

32. GeeksforGeeks. (2022, November 24). *Support Vector Machine algorithm*. GeeksforGeeks. Retrieved January 14, 2023, from https://www.geeksforgeeks.org/support-vector-machine-algorithm/.

33. Gibert, D., Mateu, C., & Planes, J. (2018). An end-to-end deep learning architecture for classification of malware's binary content. *Artificial Neural Networks and Machine Learning – ICANN 2018*, 383–391. https://doi.org/10.1007/978-3-030-01424-7_38.

34. Gibert, D., Mateu, C., & Planes, J. (2020). The rise of machine learning for detection and classification of malware: Research developments, trends and challenges. *Journal of Network and Computer Applications*, *153*, 102526. https://doi.org/10.1016/j.jnca.2019.102526.

35. Gilpin, L. H., Bau, D., Yuan, B. Z., Bajwa, A., Specter, M., & Kagal, L. (2018, May 31). *Explaining explanations: An approach to evaluating interpretability of machine learning*. arXiv.org. Retrieved February 1, 2022, from https://arxiv.org/abs/1806.00069v1.

36. Han, W., Xue, J., Wang, Y., Huang, L., Kong, Z., & Mao, L. (2019). Maldae: Detecting and explaining malware based on correlation and fusion of static and dynamic characteristics. *Computers & Security*, *83*, 208–233. https://doi.org/10.1016/j.cose.2019.02.007.

37. Homoliak, I., Malinka, K., & Hanacek, P. (2020). ASNM datasets: A collection of network attacks for testing of adversarial classifiers and intrusion detectors. *IEEE Access*, *8*, 112427–112453. https://doi.org/10.1109/access.2020.3001768.

38. Hopcroft, J. E., Motwani, R., & Ullman, J. D. (2022). *Introduction to automata theory, languages, and computation*. Pearson Education.

39. Iorliam, A., Tirunagari, S., Ho, A. T. S., Li, S., Waller, A., & Poh, N. (2017, January 20). *"flow size difference" can make a difference: Detecting malicious TCP Network flows based on Benford's law*. arXiv.org. Retrieved December 5, 2022, from https://arxiv.org/abs/1609.04214v2.

40. Ivanisenko, I., Kirichenko, L., & Radivilova, T. (2016). Investigation of multifractal properties of Additive Data Stream. *2016 IEEE First International Conference on Data Stream Mining & Processing (DSMP)*. https://doi.org/10.1109/dsmp.2016.7583564.

41. Jacobson, V., Braden, R., & Borman, D. (1992, May 1). *TCP extensions for high performance*. RFC Editor. Retrieved October 16, 2022, from https://www.rfc-editor.org/rfc/rfc1323.

42. A K, H. (2021). Techniques for detecting malware traffic: A comprehensive approach to feature selection and classification. *International Journal for Research in Applied Science and Engineering Technology*, *9*(12), 1–10. https://doi.org/10.22214/ijraset.2021.39088.

43. Kancherla, K., & Mukkamala, S. (2013). Image visualization based malware detection. *2013 IEEE Symposium on Computational Intelligence in Cyber Security (CICS)*. https://doi.org/10.1109/cicybs.2013.6597204.

44. Kirichenko, L., & Radivilova, T. (2017). Analyzes of the distributed system load with multifractal input data flows. *2017 14th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM)*. https://doi.org/10.1109/cadsm.2017.7916130.

45. Kirichenko, L., Radivilova, T., & Alghawli, A. (2013, January). *MATHEMATICAL SIMULATION OF SELF-SIMILAR NETWORK TRAFFIC WITH AIMED PARAMETERS*. ResearchGate. Retrieved May 25, 2022, from https://www.researchgate.net/publication/301220628_MATHEMATICAL_SIMULATION_OF_SELF-SIMILAR_NETWORK_TRAFFIC_WITH_AIMED_PARAMETERS.

46. Kirichenko, L., Radivilova, T., & Bulakh, V. (2018). Machine learning in Classification Time Series with fractal properties. *Data*, *4*(1), 5. https://doi.org/10.3390/data4010005.

47. Kirichenko, L., Radivilova, T., & Bulakh, V. (2019). Binary classification of fractal time series by machine learning methods. *Advances in Intelligent Systems and Computing*, 701–711. https://doi.org/10.1007/978-3-030-26474-1_49.

48. Kirichenko, L., Zinchenko, P., Radivilova, T., & Tavalbeh, M. (2019). *Machine Learning Detection of DDoS Attacks Based on Visualization of Recurrence Plots*. Ceur-ws. Retrieved June 20, 2022, from https://ceur-ws.org/Vol-2588/paper3.pdf.

49. Kolias, C., Kambourakis, G., Stavrou, A., & Voas, J. (2017). DDoS in the IOT: Mirai and other botnets. *Computer*, *50*(7), 80–84. https://doi.org/10.1109/mc.2017.201.

50. Kumar, N., Mukhopadhyay, S., Gupta, M., Handa, A., & K. Shukla, S. (2019). Malware classification using early stage behavioral analysis. *2019 14th Asia Joint Conference on Information Security (AsiaJCIS)*. https://doi.org/10.1109/asiajcis.2019.00-10.

51. Lakhno, V., Kazmirchuk, S., Kovalenko, Y., Myrutenko, L., & Zhmurko, T. (2016). Design of adaptive system of detection of cyber-attacks, based on the model of logical procedures and the coverage matrices of features. *Eastern-European Journal of Enterprise Technologies*, *3*(9(81), 30. https://doi.org/10.15587/1729-4061.2016.71769

52. Laskar, S., & Mishra, D. (2016). Qualified Vector Match and merge algorithm (QVMMA) for ddos prevention and Mitigation. *Procedia Computer Science*, *79*, 41–52. https://doi.org/10.1016/j.procs.2016.03.007.

53. Lazar, A., Wu, K., & Sim, A. (2018). Predicting network traffic using TCP ANOMALIES. *2018 IEEE International Conference on Big Data (Big Data)*. https://doi.org/10.1109/bigdata.2018.8622522.

54. Letteri, I., Della Penna, G., Di Vita, L., & Grifa, M. (2020). *MTA-KDD'19: A Dataset for Malware Traffic Detection*. Ceur-ws. Retrieved August 27, 2022, from https://ceur-ws.org/Vol-2597/paper-14.pdf.

55. Lopez, A. D., Mohan, A. P., & Nair, S. (2019). *Network traffic behavioral analytics for detection of ddos attacks*. SMU Scholar. Retrieved September 23, 2022, from https://scholar.smu.edu/datasciencereview/vol2/iss1/14/.

56. Mohaisen, A., & Alrawi, O. (2013). Unveiling zeus: automated classification of malware samples. *Proceedings of the 22nd International Conference on World Wide Web*. https://doi.org/10.1145/2487788.2488056.

57. Monnappa, K. A. (2018). *Learning malware analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows Malware*. Packt Publishing.

58. Moser, A., Kruegel, C., & Kirda, E. (2007). Limits of static analysis for malware detection. *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*. https://doi.org/10.1109/acsac.2007.21.

59. N. Ahmed, M., Abdullah, A. H., & Kaiwartya, O. (2016). FSM-F: Finite State Machine based framework for denial of service and intrusion detection in Manet. *PLOS ONE*, *11*(6). https://doi.org/10.1371/journal.pone.0156885.

60. Nashat, D., & Hussain, F. A. (2021). Multifractal detrended fluctuation analysis based detection for SYN flooding attack. *Computers & Security*, *107*, 102315. https://doi.org/10.1016/j.cose.2021.102315.

61. Nataraj, L., Karthikeyan, S., Jacob, G., & Manjunath, B. S. (2011). Malware images: visualization and automatic classification. *Proceedings of the 8th International Symposium on Visualization for Cyber Security*. https://doi.org/10.1145/2016904.2016908.

62. O'Kane, P., Sezer, S., & McLaughlin, K. (2011). Obfuscation: The hidden malware. *IEEE Security & Privacy*, *9*(5), 41–47. https://doi.org/10.1109/msp.2011.98.

63. Orset, J.-M., Alcalde, B., & Cavalli, A. (2005). An EFSM-based intrusion detection system for ad hoc networks. *Automated Technology for Verification and Analysis*, 400–413. https://doi.org/10.1007/11562948_30.

64. Passeri, P. (2023, February 2). *Q3 2022 cyber attacks statistics*. HACKMAGEDDON. Retrieved January 14, 2023, from https://www.hackmageddon.com/2022/11/03/q3-2022-cyber-attacks-statistics/.

65. Paxson, V., Allman, M., Dawson, S., Fenner, W., Griner, J., Heavens, I., Lahey, K., Semke, J., & Volz, B. (1999). Known TCP implementation problems. https://doi.org/10.17487/rfc2525.

66. Postel, J. (1980). *RFC 768: User datagram protocol*. Rfc-Editor. Retrieved May 12, 2021, from https://www.rfc-editor.org/rfc/rfc768.

67. Postel, J. (1981, September). *RFC 793: Transmission control protocol*. Rfc-Editor. Retrieved August 9, 2022, from https://www.rfc-editor.org/rfc/rfc793.html.

68. Prasse, P., Machlica, L., Pevny, T., Havelka, J., & Scheffer, T. (2017). Malware detection by analysing network traffic with Neural Networks. *2017 IEEE Security and Privacy Workshops (SPW)*. https://doi.org/10.1109/spw.2017.8.

69. *Principal component analysis*. javatpoint. (2021). Retrieved January 10, 2022, from https://www.javatpoint.com/principal-component-analysis.

70. Rezende, E., Ruppert, G., Carvalho, T., Ramos, F., & de Geus, P. (2017). Malicious software classification using transfer learning of resnet-50 deep neural network. *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*. https://doi.org/10.1109/icmla.2017.00-19.

71. Rieck, K., Trinius, P., Willems, C., & Holz, T. (2011). Automatic analysis of malware behavior using machine learning. *Journal of Computer Security*, *19*(4), 639–668. https://doi.org/10.3233/jcs-2010-0410.

72. Santhosh Kumar, B. J., & Karthik Gowda, K. R. (2022). Detection and prevention of TCP SYN flooding attack in WSN using protocol dependent detection and classification system. *2022 IEEE International Conference on Data Science and Information System (ICDSIS)*. https://doi.org/10.1109/icdsis55133.2022.9915949.

73. Savchenko, V., Haidur, H., Dzyuba, T., Marchenko, V., Matsko, O., & Havryliuk, I. (2021). Providing of data center information security on the basis of Performance Balance. *2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT)*. https://doi.org/10.1109/atit54053.2021.9678679.

74. Sekar, R., Gupta, A., Frullo, J., Shanbhag, T., Tiwari, A., Yang, H., & Zhou, S. (2002). Specification-based anomaly detection. *Proceedings of the 9th ACM Conference on Computer and Communications Security*. https://doi.org/10.1145/586110.586146.

75. Shah, A. A., Khan, Y. D., & Ashraf, M. A. (2018). Attacks analysis of TCP and UDP OF UNCW-NB15 dataset. *VAWKUM Transactions on Computer Sciences*, *15*(3), 143. https://doi.org/10.21015/vtcs.v15i3.528.

76. Sharafaldin, I., Habibi Lashkari, A., & Ghorbani, A. A. (2018). Toward generating a new intrusion detection dataset and intrusion traffic characterization.

Proceedings of the 4th International Conference on Information Systems Security and Privacy. https://doi.org/10.5220/0006639801080116.

77. Sharafaldin, I., Lashkari, A. H., Hakak, S., & Ghorbani, A. A. (2019). Developing realistic distributed denial of service (ddos) attack dataset and taxonomy. *2019 International Carnahan Conference on Security Technology (ICCST)*. https://doi.org/10.1109/ccst.2019.8888419.

78. She, C., Wen, W., Zheng, K., & Lyu, Y. (2016). Application-layer ddos detection by K-means algorithm. *Proceedings of the 2016 4th International Conference on Electrical & Electronics Engineering and Computer Science (ICEEECS 2016)*. https://doi.org/10.2991/iceeecs-16.2016.16.

79. Shirataki, S., & Yamaguchi, S. (2017). A study on interpretability of decision of machine learning. *2017 IEEE International Conference on Big Data (Big Data)*. https://doi.org/10.1109/bigdata.2017.8258557.

80. Shirey, R. (2007). Internet security glossary, version 2. https://doi.org/10.17487/rfc4949.

81. Socolofsky, T. J., & Kale, C. J. (1991). TCP/IP tutorial. https://doi.org/10.17487/rfc1180.

82. Sorokin, I. (2011). Comparing files using structural entropy. *Journal in Computer Virology*, *7*(4), 259–265. https://doi.org/10.1007/s11416-011-0153-9.

83. Stevens, W. (1997). RFC 2001: TCP slow start, congestion avoidance, fast retransmit, and fast recovery algorithms. https://doi.org/10.17487/rfc2001.

84. Stiawan, D., Wahyudi, D., Heryanto, A., Samsuryadi, S., Idris, M. Y., Muchtar, F., Abdullah Alzahrani, M., & Budiarto, R. (2019). TCP Fin Flood attack pattern recognition on internet of things with rule based signature analysis. *International Journal of Online and Biomedical Engineering (IJOE)*, *15*(07), 124. https://doi.org/10.3991/ijoe.v15i07.9848.

85. Stryczek, S., & Natkaniec, M. (2022). Internet threat detection in smart grids based on network traffic analysis using LSTM, if, and SVM. *Energies*, *16*(1), 329. https://doi.org/10.3390/en16010329.

86. *Support Vector Machines (svms)*. ritchieng. (2021). Retrieved July 13, 2022, from https://www.ritchieng.com/machine-learning-svms-support-vector-machines/.

87. Szymoniak, S., Siedlecka-Lamch, O., Zbrzezny, A. M., Zbrzezny, A., & Kurkowski, M. (2021). Sat and SMT-based verification of security protocols including Time Aspects. *Sensors*, *21*(9), 3055. https://doi.org/10.3390/s21093055.

88. Ucci, D., Aniello, L., & Baldoni, R. (2019). Survey of machine learning techniques for malware analysis. *Computers & Security*, *81*, 123–147. https://doi.org/10.1016/j.cose.2018.11.001.

89. *What is a protocol? - definition from Techopedia*. Techopedia.com. (2020). Retrieved June 11, 2022, from https://www.techopedia.com/definition/4528/protocol.

90. Wojnowicz, M., Chisholm, G., Wolff, M., & Zhao, X. (2016). Wavelet decomposition of software entropy reveals symptoms of malicious code. *Journal of Innovation in Digital Ecosystems*, *3*(2), 130–140. https://doi.org/10.1016/j.jides.2016.10.009.

91. Ye, Y., Chen, L., Wang, D., Li, T., Jiang, Q., & Zhao, M. (2008). SBMDS: An interpretable string based malware detection system using SVM Ensemble with bagging. *Journal in Computer Virology*, *5*(4), 283–293. https://doi.org/10.1007/s11416-008-0108-y.

92. Yokkampon, U., Chumkamon, S., Mowshowitz, A., Fujisawa, R., & Hayashi, E. (2021). Anomaly detection using support vector machines for time series data. *Journal of Robotics, Networking and Artificial Life*, *8*(1), 41. https://doi.org/10.2991/jrnal.k.210521.010.

93. You, I., & Yim, K. (2010). Malware obfuscation techniques: A brief survey. *2010 International Conference on Broadband, Wireless Computing, Communication and Applications*. https://doi.org/10.1109/bwcca.2010.85.

94. Yuan, X., Li, C., & Li, X. (2017). DeepDefense: Identifying ddos attack via Deep Learning. *2017 IEEE International Conference on Smart Computing (SMARTCOMP)*. https://doi.org/10.1109/smartcomp.2017.7946998.

95. Yuxin, D., & Siyi, Z. (2017). Malware detection based on Deep Learning algorithm. *Neural Computing and Applications*, *31*(2), 461–472. https://doi.org/10.1007/s00521-017-3077-6.

96. Zaghal, R., & Khan, J. (2005, January). *EFSM/SDL modeling of the original TCP standard (RFC793) and the Congestion Control Mechanism of TCP Reno*. ResearchGate. Retrieved August 20, 2022, from https://www.researchgate.net/publication/228575536_EFSMSDL_modeling_of_the_original_TCP_standard_RFC793_and_the_Congestion_Control_Mechanism_of_TCP_Reno.Zhang, X.

97. Zhen, Chen, W., Cui, J., & Xiao, W. (2022). Research on multi-path Network Balancing Method for cloud computing. *International Journal of Embedded Systems*, *15*(5), 421. https://doi.org/10.1504/ijes.2022.127165.

98. Веселков, Н., & Марченко, В. (2021). Можливості та перспективи використання Open-Source технологій виявлення вторгнень в сферах малого та середнього бізнесу України. Сучасний Захист Інформації, 1(45). https://doi.org/10.31673/2409-7292.2021.011925.

99. Гайдур, Г. І., Гахов, С. О., & Марченко, В. В. (2022). Метод побудови динамічної моделі логічного об’єкта інформаційної системи та визначення закону його функціонування. *RADIOELECTRONIC AND COMPUTER SYSTEMS*, (1), 129–140. https://doi.org/10.32620/reks.2022.1.10.

100. Гапоненко, О., Марченко, В., & Гайдур, Г. (2020). Переваги та недоліки HoneyPot — приманки для хакерів. Сучасний Захист Інформації, 1(42). https://doi.org/10.31673/2409-7292.2020.025968.

101. Гладун, А. Я., Пучков, О. О., Субач, І. Ю., & Хала, К. О. (2018). *Англо-український СЛОВНИК ТЕРМІНІВ з інформаційних технологій та кібербезпеки*. kpi. Retrieved January 14, 2022, from https://ela.kpi.ua/bitstream/123456789/45895/1/Anhlo-ukrainskyi_slovnyk_kiberbezpeka.pdf.

102. Киричок, Р., Зінченко, О., Срібна, І., Марченко, В., & Кітура, О. (2021). Удосконалений метод автоматичного активного аналізу захищеності

корпоративної мережі. *Захист Інформації*, *2*(2021). https://doi.org/https://doi.org/doi.org/10.18372/2410-7840.23.15725.

103. Корченко, А. О. (2019, February 22). *Методи ідентифікації аномальних станів для систем виявлення вторгнень*. Репозитарій Національного Авіаційного Університету: Home. Retrieved April 11, 2022, from https://er.nau.edu.ua/handle/NAU/38772.

104. Корченко, А. О., Гізун, А. І., Волянська, В. В., Казмірчук, С. В., & Гнатюк, С. О. (2013). Метод виявлення та ідентифікації порушника в інформаційно-комунікаційних системах. *Захист Інформації*, *15*(4). https://doi.org/10.18372/2410-7840.15.5733.

105. Корченко, О. Г., Терейковський, І. А., & Казмірчук, С. В. (2015). Верифікація нейромережевих методів розпізнавання кібератак. *Управління Розвитком Складних Систем*, *17*(2014). https://doi.org/10.32347/2412-9933.2014.17.%25p.

106. Лаптєв, С., Лаптєва, Т., Гребенніков, А., Кітура, О., Марченко, В., & Луценко, М. (2021). Удосконалення моделі протидії витоку інсайдерської інформації з урахуванням нестабільності функціонування окремих елементів системи захисту інформації. Journal of Science. Lyon, (20/2021). ISSN 3475-3281.

107. Марченко, В. (2019). What is security intelligence and why does it matter today? (thesis). IX Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Київ.

108. Марченко, В., & Крючкова, Л. (2019). Радіомоніторинг частотних каналів мережі Wi-Fi. Сучасний Захист Інформації, 2(38). https://doi.org/10.31673/2409-7292.2019.023236.

109. Марченко, В., & Шевчук, Я. (2017). Захист інформації на підприємстві (thesis). V Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Київ.

110. Савченко, В., Мацько, О., Легомінова, С., Полторак, І., & Марченко, В. (2019). Моделювання кібератак засобами теорії графів.

Сучасний Захист Інформації, 4(40). https://doi.org/10.31673/2409-7292.2019.040611.

111. Юрченко, В., & Марченко, В. (2018). Методика організації мережевої розвідки на основі Sql-ін’єкції (thesis). V міжнародна науково-практична конференція «актуальні проблеми забезпечення Інформаційної та кібернетичної безпеки», Київ.

ДОДАТОК А

Про впровадження результатів
дисертаційної роботи

ДОВІДКА

про впровадження результатів отриманих при виконанні дисертаційного дослідження
Марченка Віталія Вікторовича
на тему: «Метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів»

Основні положення, рекомендації дисертаційного дослідження на здобуття наукового ступеня доктора філософії за спеціальністю 125 «Кібербезпека» Марченка Віталія Вікторовича можуть використовуватись в аналітичній роботі ТОВ «Смартс».

Для підприємства заслуговує на увагу та представляє цінність запропонована методика виявлення шкідливих процесів в інформаційній системі на основі ідентифікації та діагностування станів логічних об'єктів. Ефективність виявлення шкідливих процесів основана на моделі протоколу TCP у вигляді станів логічних об'єктів. Отримано покращену модель ідентифікації та діагностування на основі машинного навчання, яке виконується на транспортному рівні шляхом аналізу трафіка ще на ранній стадії встановлення з'єднання та передачі даних на транспортному рівні моделі OSI.

Розроблений метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів підвищує рівень виявлення шкідливих процесів, що дозволить ефективніше запобігати втручанню в інформаційний простір організації на ранній стадії встановлення з'єднання. Тому даний метод може бути використаний для удосконалення систем виявлення та протидії вторгненням в інформаційну систему підприємства.

Директор
ТОВ «Смартс» В. Соболь

ЗАТВЕРДЖУЮ

Проректор з навчально-виховної та наукової роботи Державного університету телекомунікацій доктор технічних наук, професор

_____ Любов БЕРКМАН

«_____» _____ 2023 р.

АКТ

впровадження в освітній процес Державного університету телекомунікацій наукових результатів **Марченка Віталія Вікторовича**, одержаних під час проведення дисертаційного дослідження на здобуття наукового ступеня доктора філософії за спеціальністю 125 Кібербезпека, галузі знань 12 Інформаційні технології на тему: «Метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів».

Комісія у складі:

голови комісії – директора Навчально-наукового інституту захисту інформації, д.т.н., професора Савченка В.А.;

членів комісії:

завідувача кафедри Управління інформаційною та кібернетичною безпекою, д.е.н., професора Легомінова С.В.;

завідувача кафедри Штучного інтелекту, д.т.н., доцента Зінченко О.В.;

доцент кафедри Інформаційної та кібернетичної безпеки, к.військ.н., доцента Гахова С.О.;

провела роботу щодо визначення фактичного впровадження результатів наукового дослідження здобувача наукового ступеня доктора філософії Марченко В.В. в освітній процес Державного університету телекомунікацій.

У результаті проведеної роботи комісія встановила:

1. Нові наукові результати, одержані Марченко В.В.., використовуються під час навчальних занять зі здобувачами вищої освіти освітніх рівнів Бакалавр та Магістр у наступних дисциплінах:

1.1. Вперше розроблену модель ідентифікації та діагностування станів логічних об'єктів, що реалізована на базі розширеного кінцевого автомату на основі протоколу TCP – під час проведення лекційних та практичних занять з дисциплін «Захист від шкідливого програмного забезпечення», «Технології виявлення уразливостей та протидії злоякісному програмному забезпеченню» при підготовці здобувачів освіти за спеціальністю 125Кібербезпека.

1.2. Удосконалений метод вибору критеріїв логічних об'єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі методу головних компонент – під час проведення лекційних занять з дисциплін «Цифрова криміналістика», «Технології виявлення уразливостей мережевих ресурсів» при підготовці здобувачів освіти за спеціальністю 125 Кібербезпека.

1.3. Удосконалений метод виявлення шкідливих процесів в інформаційній системі організації, який базується на алгоритмах машинного навчання – під час проведення лекційних та практичних занять з дисциплін «Штучний інтелект», «Організація проведення наукових досліджень» при підготовці здобувачів освіти за спеціальністю 125 Кібербезпека.

2. Зазначені наукові результати Марченка В.В. представлені у формі окремих навчальних питань та включені до методичних розробок для лекційних та практичних занять, які проводяться у Державному університеті телекомунікацій кафедрами: Інформаційної та кібернетичної безпеки, Штучного інтелекту.

Голова комісії:

Директор Навчально-наукового інституту
захисту інформації
д.т.н., професор — Віталій САВЧЕНКО

Члени комісії:

Завідувач кафедри Управління і
нформаційною та кібернетичною
безпекою
д.е.н., професор — Світлана ЛЕГОМІНОВА

Завідувач кафедри Штучного
Інтелекту
д.т.н., доцент — Ольга ЗІНЧЕНКО

Доцент кафедри Інформаційної та
кібернетичної безпеки
к.військ.н., доцент — Сергій ГАХОВ

eurotelecom

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ «ЄВРОТЕЛЕКОМ» 34837389 | +38 099 915-91-59 office@etele.com.ua | 03151, Україна, м.Київ, вул.Волинська, буд.10

№ 16/1 2023-2/1
від 16 01 2023 р.

Про впровадження результатів
дисертаційної роботи

ДОВІДКА
про впровадження результатів отриманих при виконанні дисертаційного дослідження Марченка Віталія Вікторовича на тему: «Метод виявлення шкідливих процесів в інформаційній системі підприємства на основі ідентифікації та діагностування станів логічних об'єктів»

Ця довідка свідчить про те, що результати, отримані в дисертаційному дослідженні старшого викладача кафедри Інформаційної та кібернетичної безпеки Державного університету телекомунікацій Марченка Віталія Вікторовича, були використані в ході роботи ТОВ «ЄВРОТЕЛЕКОМ» щодо підвищення ефективності виявлення шкідливих процесів в інформаційній системі.

Перелік реалізації та впровадження результатів дослідження:

1. Удосконалено метод вибору критеріїв логічних об'єктів, які підлягають ідентифікації та діагностуванню щодо виявлення шкідливих процесів на основі методу основних компонентів з усіх параметрів вхідних даних, що на відміну від існуючих, визначає найбільш важливі параметри логічного об'єкту моделі розширеного скінченного автомату TCP протоколу. Такий підхід дозволяє прискорити обробку даних при виявленні шкідливих процесів в інформаційній системі організації в реальному часі.

2. Удосконалено метод виявлення шкідливих процесів в інформаційній системі організації, який базується на методі опорних векторів та, на відміну від існуючих, враховує обрані параметри які підлягають ідентифікації та діагностуванню. Такий метод дозволяє підвищити ефективність ідентифікації та діагностування станів логічних об'єктів в інформаційній системі організації в реальному часі та зменшити хибно-позитивні спрацювання.

Наукові та практичні результати дослідження можуть знайти подальше практичне застосування в процесі розробки та удосконалення існуючих превентивних механізмів забезпечення кібербезпеки інформаційних систем в сучасних системах кібербезпеки.

Дана довідка не є підставою для фінансових взаєморозрахунків.

Директор **Павло БУЛАВІН**

ДОДАТОК Б
СПИСОК ОПУБЛІКОВАНИХ ПРАЦЬ ЗА ТЕМОЮ ДИСЕРТАЦІЇ

1. Savchenko, V., Haidur, H., Dzyuba, T., Marchenko, V., Matsko, O., & Havryliuk, I. (2021). Providing of data center information security on the basis of Performance Balance. *2021 IEEE 3rd International Conference on Advanced Trends in Information Theory (ATIT)*. https://doi.org/10.1109/atit54053.2021.9678679.

2. Гайдур, Г. І., Гахов, С. О., & Марченко, В. В. (2022). Метод побудови динамічної моделі логічного об’єкта інформаційної системи та визначення закону його функціонування. *RADIOELECTRONIC AND COMPUTER SYSTEMS*, (1), 129–140. https://doi.org/10.32620/reks.2022.1.10.

3. Лаптєв, С., Лаптєва, Т., Гребенніков, А., Кітура, О., Марченко, В., & Луценко, М. (2021). Удосконалення моделі протидії витоку інсайдерської інформації з урахуванням нестабільності функціонування окремих елементів системи захисту інформації. *Journal of Science. Lyon*, (20/2021). ISSN 3475-3281.

4. Dorokhin, O. O., Marchenko, V. V., & Semenova, I. D. (2021). Threat Intelligence Technology and methods of its use to protect the company from Cyber Threats. *Modern Information Security*, *3*(47). https://doi.org/10.31673/2409-7292.2021.031216.

5. Киричок, Р., Зінченко, О., Срібна, І., Марченко, В., & Кітура, О. (2021). Удосконалений метод автоматичного активного аналізу захищеності корпоративної мережі. *Захист Інформації*, *2*(2021). https://doi.org/doi.org/10.18372/2410-7840.23.15725.

6. Веселков, Н., & Марченко, В. (2021). Можливості та перспективи використання Open-Source технологій виявлення вторгнень в сферах малого та середнього бізнесу України. *Сучасний Захист Інформації*, *45*(1). https://doi.org/10.31673/2409-7292.2021.011925.

7. Гапоненко, О., Марченко, В., & Гайдур, Г. (2020). Переваги та недоліки HoneyPot — приманки для хакерів. *Сучасний Захист Інформації, 1*(42). https://doi.org/10.31673/2409-7292.2020.025968.

8. Марченко, В., & Крючкова, Л. (2019). Радіомоніторинг частотних каналів мережі Wi-Fi. *Сучасний Захист Інформації, 2*(38). https://doi.org/10.31673/2409-7292.2019.023236.

9. Савченко, В., Мацько, О., Легомінова, С., Полторак, І., & Марченко, В. (2019). Моделювання кібератак засобами теорії графів. *Сучасний Захист Інформації, 4*(40). https://doi.org/10.31673/2409-7292.2019.040611.

10. Марченко, В. (2019). *What is security intelligence and why does it matter today?* (thesis). IX Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Київ.

11. Марченко, В., & Шевчук, Я. (2017). *Захист інформації на підприємстві* (thesis). V Міжнародна науково-технічна конференція студентства та молоді «Світ інформації та телекомунікацій», Киїів.

12. Юрченко, В., & Марченко, В. (2018). *Методика організації мережевої розвідки на основі Sql-ін'єкції* (thesis). V міжнародна науково-практична конференція «актуальні проблеми забезпечення Інформаційної та кібернетичної безпеки», Київ.