Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

Вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних системи від прикладних атак на основі кореляційного аналізу логів та часових рядів

Виконав: здобувач 2-го курсу,
групи 1КІТС-24м
спеціальності 125– Кібербезпека
та захист інформації
Освітня програма – Кібербезпека
інформаційних технологій та систем
(шифр і назва напряму підготовки, спеціальності)
Колєсов І.С.
(прізвище та ініціали)
Керівник: к.ф.-м.н., доц. каф. МБІС
Шиян А.А.
(прізвище та ініціали)
«11» грудня 2025 р.
Опонент: к.т.н., доц. каф. ОТ
Крупельницький Л.В.
(прізвище та ініціали)
«11» грудня 2025 р.

Допущено до захисту
Голова секції УБ кафедри МБІС
Юрій ЯРЕМЧУК
“11” грудня 2025 р.

Вінниця ВНТУ - 2025 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека та захист інформації
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС
Юрій ЯРЕМЧУК
"24" вересня 2025 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу студенту
Колєсов Іван Сергійович
(прізвище, ім`я, по-батькові)

1. Тема роботи:
«Вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних системи від прикладних атак на основі кореляційного аналізу логів та часових рядів»
Керівник роботи: к.ф.-м.н., доц. каф. МБІС Шиян А. А.
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від "24" вересня 2025 року № 313
2. Строк подання студентом роботи за тиждень до захисту.
3. Вихідні дані до роботи: динамічний поведінковий аналіз, кореляційний аналіз логів, аналіз часових рядів, веб-додатки, захист від прикладних атак, машинне навчання.
4. Зміст текстової частини:
Робота складається з чотирьох розділів. У першому розділі розглянуто аналіз сучасних методів захисту веб-орієнтованих інформаційних систем від прикладних атак, систематизовано класифікацію атак та їхні вектори. У другому розділі розроблено удосконалений метод динамічного поведінкового аналізу, що базується на інтеграції математичної моделі кореляційного аналізу логів та методу аналізу часових рядів для виявлення атак. У третьому розділі представлена практична реалізація програмного комплексу, проведено експериментальне дослідження ефективності розробленого методу та порівняльний аналіз з існуючими рішеннями. У четвертому розділі наведено техніко-економічне обґрунтування розробки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень)
У другому розділі наведено 11 рисунків, в третьому розділі 5 рисунків.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина			
I	Шиян А. А. к.ф.-м.н., доц. каф. МБІС		
II	Шиян А. А. к.ф.-м.н., доц. каф. МБІС		
III	Шиян А. А. к.ф.-м.н., доц. каф. МБІС		
Економічна частина			
IV	Ратушняк О. Г. , доц. каф. ЕПВМ, к.т.н.		

7.Дата видачі завдання 24 вересня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Прим
1.	Визначення напрямку магістерської роботи, формулювання теми	24.09.2025	05.10.2025	
2.	Аналіз предметної області обраної теми	06.10.2025	20.10.2025	
3.	Розробка роботи	21.10.2025	31.10.2025	
4.	Написання магістерської роботи на основі розробленої теми	01.11.2025	20.11.2025	
5.	Передзахист магістерської роботи	21.11.2025	29.11.2025	
6.	Виправлення, уточнення, коригування магістерської кваліфікаційної роботи	30.11.2025	07.12.2025	
7.	Захист магістерської кваліфікаційної роботи	08.12.2025	11.12.2025	

Студент _____ (підпис) Колєсов І. С.

Керівник роботи _____ (підпис) Шиян А. А.

АНОТАЦІЯ

УДК: 004.056

Колєсов І.С. Удосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних систем. Магістерська кваліфікаційна робота зі спеціальності 125 — Кібербезпека та захист інформації, освітня програма — Кібербезпека інформаційних технологій та систем. Вінниця: ВНТУ, 2025. 128 с.

На укр. мові. Бібліогр.: 77 назв; рис.: 16; табл. 15.

У магістерській кваліфікаційній роботі розроблено програмний комплекс для вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних систем від прикладних атак на основі кореляційного аналізу логів та часових рядів. У загальній частині роботи розглянуто сучасні підходи до захисту веб-додатків, проведено аналіз векторів прикладних атак та існуючих методів їх виявлення, наведено аргументи для обґрунтування необхідності інтеграції кореляційного аналізу та методів обробки часових рядів, представлено концептуальну модель системи захисту.

У технологічній частині розроблений програмний засіб для реалізації вдосконаленого методу динамічного поведінкового аналізу з використанням алгоритмів машинного навчання (Random Forest та LSTM). Проведено експериментальне дослідження ефективності розробленого рішення, за результатами якого досягнуто показника точності F1-score 0.969, а також виконано порівняльний аналіз з існуючими аналогами.

Економічна частина містить обґрунтування доцільності розробки власної системи захисту.

Ключові слова: веб-орієнтована система, поведінковий аналіз, кореляційний аналіз, часові ряди, машинне навчання, кібербезпека, детектування атак.

ABSTRACT

Koliesov I.S. Improvement of the dynamic behavioral analysis method for protecting web-oriented information systems against application attacks based on log correlation analysis and time series. Master's Thesis in Specialty 125 — Cybersecurity and Information Protection, Educational Program — Cybersecurity of Information Technologies and Systems. Vinnytsia: VNTU, 2025. 128 p.

In Ukrainian. Refs.: 77 titles; figures: 16; tables: 15.

In the master's qualification work, a software complex has been developed to improve the method of dynamic behavioral analysis for protecting web-oriented information systems from application attacks based on log correlation analysis and time series. In the general part of the thesis, modern approaches to protecting web applications are considered, an analysis of application attack vectors and existing detection methods is conducted, arguments justifying the necessity of integrating log correlation analysis and time series processing methods are provided, and a conceptual model of the protection system is presented.

In the technological part, a software tool has been developed to implement the improved method of dynamic behavioral analysis using machine learning algorithms (Random Forest and LSTM). An experimental study of the effectiveness of the developed solution was conducted, resulting in an F1-score of 0.969, and a comparative analysis with existing analogs was performed.

In the economic part, the feasibility of developing this software tool is substantiated.

Keywords: web-oriented system, behavioral analysis, correlation analysis, time series, machine learning, cybersecurity, attack detection.

ЗМІСТ

ВСТУП

Актуальність. Веб-орієнтовані інформаційні системи становлять критичну інфраструктуру сучасного цифрового суспільства, забезпечуючи функціонування електронної комерції, банківських сервісів, державних порталів, корпоративних додатків. Статистика Verizon Data Breach Investigations Report 2024 фіксує 8.4% ймовірність успішної кібератаки на веб-додатки щорічно з середніми збитками $4.45 мільйона за інцидент згідно IBM Cost of Data Breach Report. Традиційні методи захисту базуються переважно на сигнатурному аналізі та статичних правилах фільтрації трафіку, що виявляється недостатнім для детектування zero-day атак, складних багатоетапних сценаріїв, атак з повільним розгортанням у часі.

Дослідження 2023-2024 років демонструють зростання частки атак на прикладному рівні до 67% від загальної кількості інцидентів безпеки, при цьому середній час виявлення атаки складає 207 днів за даними Mandiant M-Trends 2024. Регуляторні вимоги GDPR, PCI DSS, SOC 2 встановлюють жорсткі штрафи до 4% річного обороту за неналежний захист персональних даних, що робить питання забезпечення безпеки критичним з економічної перспективи. Аналіз наукових публікацій 2018-2025 років виявив зростання інтересу дослідників до методів машинного навчання у кібербезпеці з 18% у 2018 до 64% у 2024, проте лише 12% робіт інтегрують кореляційний аналіз логів з аналізом часових рядів для комплексного детектування загроз.

Динамічний поведінковий аналіз дозволяє виявляти аномалії через відхилення від профілів нормальної поведінки без залежності від баз сигнатур атак. Кореляційний аналіз логів ідентифікує нетипові комбінації атрибутів у запитах користувачів, тоді як аналіз часових рядів виявляє аномальні паттерни у динаміці агрегованих метрик системи. Ізольоване застосування цих підходів не забезпечує достатньої точності — кореляційні методи пропускають атаки з розподілом у часі, методи часових рядів втрачають деталізацію на рівні

окремих запитів. Інтеграція обох підходів на множинних рівнях абстракції формує синергетичний ефект через взаємне підтвердження детекцій з різних джерел інформації.

Метою роботи є підвищення ефективності захисту веб-орієнтованих інформаційних систем через розробку удосконаленого методу динамічного поведінкового аналізу на основі інтеграції кореляційного аналізу логів та аналізу часових рядів.

Задачі дослідження:

1. Проаналізувати сучасні вектори прикладних атак на веб-орієнтовані системи та існуючі методи їх захисту, виявивши недоліки традиційних підходів до виявлення загроз у логах та трафіку.
2. Удосконалити метод динамічного поведінкового аналізу шляхом інтеграції математичних моделей кореляційного аналізу логів та методів обробки часових рядів для підвищення точності детектування аномалій.
3. Розробити програмний засіб для реалізації запропонованого удосконалення та перевірити його ефективність шляхом експериментального дослідження на еталонних датасетах мережевого трафіку..

Об'єкт дослідження — процеси виявлення та класифікації кібератак на веб-орієнтовані інформаційні системи.

Предмет дослідження — методи динамічного поведінкового аналізу на основі кореляційного аналізу логів та аналізу часових рядів для захисту веб-додатків.

Методи дослідження. У роботі використано методи математичної статистики для обчислення коефіцієнтів кореляції та побудови базових профілів поведінки, методи аналізу часових рядів (декомпозиція STL, ARIMA моделювання, спектральний аналіз) для виявлення аномалій у динаміці метрик, методи машинного навчання (Random Forest, LSTM-мережі, ансамблювання) для класифікації типів атак, методи теорії ймовірностей для оцінювання ризиків та очікуваних збитків, експериментальні методи для валідації ефективності розроблених алгоритмів.

Новизна роботи:

1. Запропоновано новий метод захисту веб-додатків, який, на відміну від існуючих, одночасно аналізує і окремі запити (через кореляцію логів), і загальну динаміку системи (через часові ряди). Таке поєднання дозволило знизити кількість хибних спрацювань (False Positive) на 73%.

2. Удосконалено класифікацію атак за допомогою гібридної моделі, що поєднує алгоритм Random Forest (аналізує статистику) та нейромережу LSTM (аналізує послідовність дій). Завдяки цьому точність виявлення (F1-score) зросла до 0.969 порівняно з 0.897 при використанні лише LSTM.

3. Розвинуто математичний апарат адаптивного налаштування параметрів детектування через функцію $W(t) = \max(W_{min}, \min(W_{max}, \lambda(t) \times T_{target}))$ для розміру ковзного вікна кореляційного аналізу залежно від поточної інтенсивності трафіку та $\theta(t) = \mu(t) + k\sigma(t)$ для порогу виявлення аномалій з експоненційним згладжуванням $\mu(t) = \alpha x(t) + (1-\alpha)\mu(t-1)$, що забезпечує стабільність статистичних оцінок при змінному навантаженні та природних коливаннях метрик.

Практичне значення одержаних результатів полягає у створенні функціонуючого програмного комплексу захисту веб-орієнтованих інформаційних систем з Python, навченими моделями Random Forest (200 дерев) та LSTM (архітектура 128-64-32). Система забезпечує продуктивність 850 запитів на секунду при середній латентності детектування 12.4 мілісекунди, досягає F1-score 0.969 на стандартних датасетах CICIDS2017 та CSE-CIC-IDS2018.

В економічному розділі обґрунтовано доцільність створення даного програмного засобу.

РОЗДІЛ 1. АНАЛІЗ СУЧАСНИХ МЕТОДІВ ЗАХИСТУ ВЕБ-ОРІЄНТОВАНИХ ІНФОРМАЦІЙНИХ СИСТЕМ ВІД ПРИКЛАДНИХ АТАК

1.1 Класифікація прикладних атак на веб-орієнтовані інформаційні системи та аналіз їх векторів

Веб-орієнтовані інформаційні системи становлять фундаментальну основу функціонування організацій різного масштабу та спрямування діяльності, водночас піддаючись систематичним атакам з боку зловмисників. З. Абуабед, А. Альсаде та А. Тавіл досліджували вразливості транспортних засобів через призму моделі загроз STRIDE, що дозволяє екстраполювати методологію оцінювання ризиків на веб-інфраструктуру [1]. К. Акбар зі співавторами створили онтологічну структуру для фреймворку ATT&CK, забезпечивши систематизоване представлення тактик супротивника та технік атакування, розширюючи можливості аналітичного осмислення кібербезпекових інцидентів [6].

Прикладні атаки спрямовуються на експлуатацію вразливостей у логіці роботи застосунків, обходячи традиційні периметрові засоби захисту та використовуючи легітимні канали передавання даних для досягнення зловмисних цілей. Ін'єкційні атаки утворюють масштабний клас загроз, коли зловмисник впроваджує шкідливий код через некоректно валідовані поля введення даних, маніпулюючи SQL-запитами, командами операційної системи або скриптами інтерпретаторів. SQL-ін'єкції дозволяють атакувальникам виконувати довільні запити до бази даних, витягувати конфіденційну інформацію, модифікувати записи або повністю компрометувати серверну частину застосунку. Міжсайтовий скриптинг полягає у впровадженні шкідливих JavaScript-фрагментів, які виконуються в контексті браузера жертви, викрадаючи сесійні токени, перехоплюючи введені дані або здійснюючи фішингові атаки від імені легітимного ресурсу [8].

Атаки з підробкою міжсайтових запитів експлуатують довіру веб-застосунку до автентифікованого користувача, змушуючи браузер виконувати небажані дії без відома власника сесії. Дезеріалізація небезпечних об'єктів створює можливості для віддаленого виконання коду, коли застосунок обробляє спеціально сформовані серіалізовані структури даних без належної верифікації. Атаки на бізнес-логіку використовують недоліки проектування функціональних вимог, дозволяючи маніпулювати послідовністю операцій, обходити обмеження авторизації або отримувати несанкціонований доступ до ресурсів через логічні прогалини в реалізації [17].

Включення віддалених файлів та обхід шляхів до каталогів надають зловмисникам можливості читання довільних файлів на сервері або виконання завантаженого зовнішнього коду. Атаки типу «відмова в обслуговуванні» на прикладному рівні спрямовуються на виснаження обчислювальних ресурсів через формування ресурсоємних запитів, експлуатацію алгоритмічної складності або створення циклічних залежностей у обробці даних. Компрометація автентифікації та управління сесіями відбувається через підбір облікових даних, викрадення або підробку сесійних ідентифікаторів, фіксацію сесій або експлуатацію слабких механізмів відновлення паролів [22]. Небезпечні налаштування безпеки утворюються внаслідок використання стандартних облікових записів, відкритих діагностичних інтерфейсів, надмірних привілеїв процесів або відсутності механізмів обмеження швидкості запитів. Використання компонентів з відомими вразливостями створює вектори атак через застарілі бібліотеки, фреймворки або плагіни, які містять документовані недоліки безпеки. Недостатнє журналювання та моніторинг перешкоджають своєчасному виявленню інцидентів, дозволяючи зловмисникам тривалий час зберігати присутність у скомпрометованій системі. Атаки на ланцюги постачання програмного забезпечення компрометують залежності через впровадження шкідливого коду в публічні репозиторії або підміну легітимних пакетів [26]. Узагальнена класифікація наведена в таблиці 1.1.

Таблиця 1.1 – Класифікація прикладних атак на веб-орієнтовані інформаційні системи

Категорія атаки	**Підкатегорії**	**Вектори реалізації**	**Потенційні наслідки**
Ін'єкційні атаки	SQL-ін'єкція, NoSQL-ін'єкція, Command injection, LDAP-ін'єкція, XML-ін'єкція	Некоректна валідація введених даних, відсутність параметризованих запитів, пряме конкатенування рядків	Витік конфіденційних даних, модифікація записів БД, виконання довільних команд ОС, повна компрометація сервера
Міжсайтовий скриптинг	Reflected XSS, Stored XSS, DOM-based XSS, Mutation XSS	Недостатня санітизація виведених даних, відсутність Content Security Policy, неправильне кодування контексту	Викрадення сесійних токенів, фішинг, редирект на шкідливі ресурси, виконання дій від імені користувача
Порушення автентифікації	Brute-force атаки, Credential stuffing, Session hijacking, Session fixation	Слабкі паролі, відсутність обмеження спроб входу, незахищена передача токенів, предсказувані ідентифікатори сесій	Несанкціонований доступ до облікових записів, перехоплення сесій, компрометація привілейованих акаунтів
Атаки на бізнес-логіку	Race conditions, Parameter tampering, Workflow bypass, Price manipulation	Недоліки проектування послідовності операцій, відсутність валідації на серверній стороні, некоректна обробка асинхронних запитів	Обхід бізнес-правил, фінансові втрати, маніпуляції з транзакціями, несанкціоновані привілеї
Десеріалізація об'єктів	Insecure deserialization, Object injection, Remote code execution	Обробка ненадійних серіалізованих даних, використання небезпечних функцій десеріалізації	Віддалене виконання коду, підвищення привілеїв, компрометація цілісності застосунку
Атаки на конфіденційність даних	Sensitive data exposure, Inadequate encryption, Cryptographic failures	Передача даних без шифрування, використання слабких криптографічних алгоритмів, зберігання секретів у відкритому вигляді	Перехоплення конфіденційних даних, дешифрування зашифрованої інформації, витік персональних даних

Атаки на API веб-сервісів експлуатують специфічні вразливості RESTful та GraphQL інтерфейсів через маніпуляцію параметрами запитів, масове призначення атрибутів або отримання надмірної кількості даних через складні запити. Server-Side Request Forgery дозволяє зловмисникам змушувати сервер виконувати запити до внутрішніх ресурсів або зовнішніх систем, обходячи

мережеві обмеження та отримуючи доступ до закритих сервісів. Атаки на механізми контролю доступу використовують недоліки авторизації для горизонтального або вертикального підвищення привілеїв, доступу до чужих ресурсів або виконання адміністративних функцій без належних прав [33].

Розподілені атаки типу «відмова в обслуговуванні» на прикладному рівні генерують легітимний трафік високої інтенсивності, виснажуючи ресурси веб-серверів, баз даних або кешуючих систем. Web cache poisoning компрометує проміжні кешуючі сервери, примушуючи їх зберігати шкідливі відповіді, які потім обслуговуються іншим користувачам. HTTP request smuggling експлуатує розбіжності в інтерпретації протоколу між фронтенд та бекенд серверами, дозволяючи обходити засоби безпеки або впроваджувати запити в чужі сесії [44]. Атаки на веб-сокети використовують постійні з'єднання для обходу традиційних механізмів фільтрації або створення прихованих каналів передавання даних. Clickjacking створює невидимі накладні шари інтерфейсу, обманюючи користувачів виконувати небажані дії на прихованих сторінках. Open redirect вразливості дозволяють зловмисникам перенаправляти жертв на фішингові сайти через легітимні домени. Tabnabbing експлуатує поведінку браузерів з фоновими вкладками, підміняючи вміст довіреної сторінки після переходу користувача (таблиця 1.2).

Таблиця 1.2 – Вектори реалізації прикладних атак та методи протидії

Вектор атаки	**Технічна реалізація**	**Індикатори компрометації**	**Методи детектування**	**Стратегії мітигації**
SQL-ін'єкція через параметри запитів	Впровадження SQL-синтаксису в GET/POST параметри, модифікація cookies, маніпуляція HTTP-заголовками	Аномальні символи в логах запитів (', --, ;), помилки БД у відповідях, незвичні запити до БД	Аналіз патернів запитів, моніторинг часу виконання запитів БД, виявлення SQL-ключових слів	Параметризовані запити, ORM-фреймворки, валідація типів даних, принцип найменших привілеїв для БД
Stored XSS через форми введення	Збереження JavaScript-коду в полях	Наявність тегів <script>, обфускованого	Content Security Policy моніторинг,	Контекстне екранування виведення, CSP-

	профілю, коментарях, повідомленнях форуму	коду, подій JavaScript в збережених даних	аналіз виведених даних, детектування небезпечних патернів	політики, санітизація HTML, HTTPOnly cookies
CSRF через соціальну інженерію	Розміщення прихованих форм на зовнішніх сайтах, email-розсилки з автоматичними запитами	Запити без Referer-заголовка, незвичні джерела трафіку, дії без взаємодії користувача	Верифікація походження запитів, аналіз послідовності дій, виявлення автоматизованих запитів	CSRF-токени, SameSite cookies, верифікація Origin/Referer, подвійне підтвердження операцій
Path traversal через параметри файлів	Використання ../ послідовностей, абсолютних шляхів, URL-кодування спецсимволів	Звернення до системних файлів, читання конфігураційних файлів, доступ поза робочими каталогами	Моніторинг файлових операцій, аналіз шляхів до файлів, детектування traversal-патернів	Білий список дозволених файлів, канонікалізація шляхів, chroot-ізоляція, обмеження прав файлової системи
SSRF через обробку URL	Запити до внутрішніх IP-адрес, localhost, метадата-сервісів хмарних провайдерів	Запити до приватних діапазонів IP, звернення до AWS metadata API, сканування портів	Аналіз запитуваних URL, моніторинг мережевих з'єднань сервера, детектування сканування портів	Білий список доменів, фільтрація приватних IP, мережева сегментація, відключення редиректів
Insecure deserialization	Модифікація серіалізованих об'єктів у cookies, сесіях, кешованих даних	Незвичні класи в серіалізованих даних, виконання системних команд, зміна поведінки застосунку	Integrity checking серіалізованих даних, моніторинг викликів небезпечних методів	Уникнення десеріалізації ненадійних даних, підписування об'єктів, використання безпечних форматів (JSON), ізоляція

Атаки на механізми завантаження файлів дозволяють розміщувати веб-шелли або виконувані файли через обхід валідації типів, подвійні розширення або експлуатацію парсерів медіа-форматів. XML External Entity атаки використовують обробку XML-документів для читання локальних файлів,

сканування внутрішньої мережі або виснаження ресурсів через розкриття сутностей. Template injection експлуатує шаблонізатори серверної сторони, впроваджуючи шкідливий код у шаблони, що призводить до віддаленого виконання команд. Server-side include injection компрометує директиви включення файлів, дозволяючи виконання довільних скриптів [53].

Атаки на OAuth та OpenID Connect потоки використовують недоліки реалізації авторизаційних протоколів для крадіжки токенів доступу, підробки callback URL або проведення атак типу authorization code interception. JSON Web Token атаки експлуатують слабкі алгоритми підпису, відсутність верифікації або маніпуляцію payload для підробки ідентифікаційних токенів. Rate limiting bypass техніки обходять обмеження частоти запитів через ротацію IP-адрес, розподілені атаки або експлуатацію логічних недоліків у реалізації лічильників [58]. Масове призначення параметрів дозволяє модифікувати атрибути об'єктів, які не призначені для зовнішнього редагування, змінюючи ролі користувачів, ціни товарів або статуси замовлень. Атаки на GraphQL використовують вкладені запити для створення надмірного навантаження на бази даних, інтроспекцію схем для розкриття структури API або батчинг запитів для обходу обмежень швидкості. WebSocket hijacking компрометує постійні з'єднання через крадіжку токенів або експлуатацію недоліків автентифікації при встановленні з'єднання [61].

Вразливості бібліотек фронтенд-фреймворків створюють можливості для DOM-based атак, prototype pollution або експлуатації специфічних недоліків популярних JavaScript-бібліотек. Content Security Policy bypass техніки обходять захисні політики через JSONP-endpoints, допущені в білих списках домени або експлуатаціюlegaси функціоналу. Subdomain takeover дозволяє захоплювати контроль над неактивними піддоменами через невидалені DNS-записи, отримуючи можливість розміщувати фішингові сторінки на довірених доменах. Race condition атаки експлуатують невідповідність між перевіркою та використанням ресурсів, дозволяючи обходити обмеження через паралельне виконання запитів. Timing attacks аналізують час виконання

операцій для витягування конфіденційної інформації, обходу автентифікації або визначення існування ресурсів.

1.2 Огляд існуючих методів та засобів захисту від прикладних атак на основі поведінкового аналізу

Поведінковий аналіз становить парадигму детектування загроз, заснована на виявленні відхилень від нормальних патернів функціонування систем та поведінки користувачів, на відміну від сигнатурних методів, обмежених розпізнаванням лише відомих атак. Методологія базується на побудові профілів нормальної активності через машинне навчання, статистичний аналіз або експертні правила, дозволяючи ідентифікувати аномальні дії без попереднього знання про конкретні вектори атак [9]. Архітектури поведінкового моніторингу охоплюють аналіз мережевого трафіку, системних викликів, взаємодії з файловою системою, споживання ресурсів та послідовностей дій користувачів, формуючи багатовимірне представлення стану безпеки.

Системи виявлення вторгнень на основі аномалій використовують алгоритми кластеризації для групування подібних патернів поведінки, визначаючи відхилення як потенційні індикатори компрометації. Методи на базі нейронних мереж навчаються розпізнавати складні нелінійні залежності в послідовностях подій, виявляючи багатоетапні атаки через аналіз темпоральних кореляцій [40]. Байєсівські мережі моделюють імовірнісні залежності між різними типами активності, обчислюючи апостеріорну ймовірність зловмисної поведінки на основі спостережуваних ознак. Методи опорних векторів будують гіперплощини розділення в багатовимірному просторі ознак, класифікуючи нові спостереження як нормальні або аномальні на основі їхнього розташування відносно границі рішення [57].

Глибоке навчання для поведінкового аналізу застосовує рекурентні нейронні мережі та механізми уваги для моделювання послідовностей дій у часі, захоплюючи довгострокові залежності в поведінці застосунків.

Автокодувальники навчаються компресувати нормальну поведінку в низьковимірне представлення, виявляючи аномалії через високу помилку реконструкції для невідомих патернів [64]. Генеративно-змагальні мережі синтезують реалістичні приклади нормальної активності, тренуючи дискримінатор розрізняти справжні та згенеровані зразки, що підвищує чутливість до справжніх аномалій. Трансформерні архітектури обробляють паралельно великі контексти подій, моделюючи складні взаємозв'язки між віддаленими в часі активностями через механізми self-attention. Профілювання поведінки користувачів створює індивідуалізовані моделі типової активності кожного суб'єкта, враховуючи часові патерни роботи, звичні послідовності операцій, географічні локації та характеристики пристроїв.

User and Entity Behavior Analytics агрегує дані з множини джерел для побудови комплексних профілів, застосовуючи ризик-скорінг для ранжування подій за ступенем підозрілості [11]. Peer group analysis порівнює поведінку користувача з активністю подібних за роллю колег, виявляючи відхилення від групових норм. Insider threat detection системи спеціалізуються на ідентифікації зловмисних дій легітимних користувачів через аналіз психологічних індикаторів, порушень політик та підготовчих активностей перед ексфільтрацією даних. Порівняння методів поведінкового аналізу наведено в таблиці 1.3.

Таблиця 1.3 – Методи поведінкового аналізу для захисту веб-орієнтованих систем

Метод	**Технологічна основа**	**Аналізовані параметри**	**Переваги**	**Обмеження**
Статистичний аналіз аномалій	Гаусівські моделі, критерії Хі-квадрат, Z-score	Частота запитів, розміри payload, час виконання операцій	Низька обчислювальна складність, інтерпретованість результатів	Неефективність проти поступових атак, чутливість до вибору порогів
Кластеризація поведінки	K-means, DBSCAN,	Вектори ознак запитів,	Виявлення невідомих атак, групування	Труднощі з динамічними порогами,

	ієрархічна кластеризація	сесійні патерни, послідовності URL	подібних інцидентів	потреба в оновленні моделей
Марковські моделі	Приховані марковські моделі, марковські ланцюги	Переходи між станами сесії, послідовності HTTP-методів	Моделювання темпоральних залежностей, детектування аномальних переходів	Обмеження пам'яті минулих станів, складність навчання
Випадковий ліс	Ансамбль дерев рішень, bagging	Комбінації значень параметрів запитів, метадані headers	Стійкість до перенавчання, обробка категоріальних ознак	Важкість інтерпретації, необхідність великих обсягів даних
Рекурентні нейронні мережі	LSTM, GRU, bidirectional RNN	Послідовності символів у payload, темпоральні патерни трафіку	Захоплення довгострокових залежностей, адаптивність	Висока обчислювальна вартість, потреба в GPU-ресурсах
Автокодувальники	Варіаційні автокодувальники, denoising автокодувальники	Векторні представлення сесій, нормалізовані метрики запитів	Виявлення складних аномалій, unsupervised навчання	Труднощі з визначенням порогу аномальності, false positives
Graph-based аналіз	Аналіз графів взаємодій, community detection	Зв'язки між сутностями, топологія мережі запитів	Виявлення скоординованих атак, аналіз латеральних рухів	Масштабованість для великих графів, складність візуалізації

Аналіз поведінки застосунків на рівні коду використовує динамічну та статичну бінарну інструментацію для моніторингу системних викликів, операцій з пам'яттю та міжпроцесної комунікації. Extended Berkeley Packet Filter програми забезпечують високопродуктивний моніторинг на рівні ядра операційної системи з мінімальними накладними витратами, захоплюючи події безпосередньо в момент їх виникнення [38]. Sandboxing технології ізолюють підозрілі процеси в контрольованих середовищах, спостерігаючи за їхньою поведінкою для класифікації зловмисності. Контейнеризація та мікросервісні архітектури створюють можливості для гранулярного

моніторингу, коли кожен компонент має індивідуальний профіль очікуваної поведінки.

Мережевий поведінковий аналіз досліджує характеристики трафіку на транспортному та прикладному рівнях, виявляючи аномалії в розподілах розмірів пакетів, інтервалів між запитами, напрямків комунікації та протокольних особливостей. Deep Packet Inspection розбирає структуру прикладних протоколів, перевіряючи відповідність специфікаціям та виявляючи спроби експлуатації через малформовані запити [74]. Flow-based аналіз агрегує пакети в потоки, досліджуючи статистичні властивості з'єднань без збереження повного вмісту пакетів. NetFlow та IPFIX експортують телеметрію мережевої активності для централізованого аналізу, дозволяючи виявляти розподілені атаки через кореляцію даних з множини сенсорів. Адаптивні системи захисту модифікують моделі поведінки в реальному часі, інкорпоруючи нові патерни легітимної активності та оновлюючи детекційні правила відповідно до еволюції загроз. Online learning алгоритми інкрементально навчаються на потоках даних без повного перенавчання, забезпечуючи актуальність моделей. Reinforcement learning оптимізує стратегії реагування через взаємодію з середовищем, максимізуючи винагороду за правильні рішення та мінімізуючи збитки від хибних спрацювань. Active learning запитує мітки експертів для найбільш інформативних прикладів, ефективно покращуючи якість класифікації при обмеженій доступності анотованих даних [31].

Honeypot системи розгортають приманки для атакувальників, спостерігаючи за їхніми діями в контрольованому середовищі та збираючи intelligence про тактики, техніки та процедури зловмисників [45]. High-interaction honeypots емулюють повнофункціональні системи, дозволяючи глибокий аналіз поведінки атакувальника. Low-interaction honeypots симулюють вразливі сервіси з обмеженою функціональністю, ефективно виявляючи автоматизовані сканування та експлойти. Honeytokens

імплантують синтетичні облікові дані або документи в систему, алертуючи при будь-якому їх використанні як однозначний індикатор компрометації [7].

Deception technologies створюють багаторівневу архітектуру приманок, розподілених по мережевій інфраструктурі, підвищуючи ймовірність взаємодії атакувальника з контрольованими точками спостереження. Канарейкові токени генерують унікальні ідентифікатори, вбудовані в різні ресурси, дозволяючи відстежувати несанкціонований доступ та ексфільтрацію даних. Breadcrumb techniques залишають навмисні сліди в системі, направляючи атакувальників до honeypots та відволікаючи від справжніх цілей. Тарпіти уповільнюють взаємодію з атакувальниками, збільшуючи час виконання автоматизованих атак та підвищуючи шанси детектування (таблиця 1.4).

Таблиця 1.4 – Інструменти та платформи поведінкового аналізу

Платформа	**Архітектура**	**Підтримувані джерела даних**	**Методи аналізу**	**Сценарії застосування**
Suricata IDS	Багатопотоковий движок, EVE JSON logging	Мережевий трафік, flow data, файлові транзакції	Сигнатури Suricata, Lua scripting, аномалії трафіку	Периметровий моніторинг, виявлення експлойтів, file extraction
Zeek (Bro)	Подієво-орієнтована архітектура, Zeek scripting	Пакетний трафік, протокольні логи, extracted файли	Скриптова аналітика, machine learning frameworks	Дослідження інцидентів, threat hunting, протокольний аналіз
Wazuh	Агентська архітектура, централізований менеджер	Системні логи, FIM, rootkit detection, vulnerability scanning	Правила декодування, CDB lists, машинне навчання	Host-based IDS, compliance monitoring, integrity checking
OSSEC	Легковаговий агент, agentless моніторинг	Syslog, Windows Event Log, command output	Логічні правила, statistical analysis	Моніторинг цілісності, log analysis, активне реагування
Elastic Security	ELK-стек інтеграція, Kibana візуалізація	Beats-агенти, network packets, cloud logs	Detection rules, machine learning jobs, anomaly detection	SIEM, endpoint protection, threat intelligence integration

Splunk Enterprise Security	Розподілені індексери, search processing language	Universal forwarders, HTTP Event Collector, DBConnect	Кореляційні пошуки, statistical commands, MLTK	Enterprise SIEM, операційний intelligence, compliance reporting
AlienVault OSSIM	Модульна архітектура, SIEM-корелятор	HIDS agents, network sensors, vulnerability scanners	Event correlation, reputation intelligence	Unified security management, threat detection, forensics

Контекстно-залежний аналіз поведінки враховує бізнес-логіку застосунків, ролі користувачів, часові обмеження та географічні фактори для точнішого визначення аномальності дій. Логи доступу до ресурсів аналізуються з урахуванням дозволів, визначених політиками безпеки, виявляючи спроби несанкціонованих операцій. Часова сегментація диференціює нормальну поведінку в робочі години від активності в нестандартний час, коли легітимні операції менш імовірні. Географічна аналітика детектує impossible travel scenarios, коли автентифікації відбуваються з локацій, фізично недосяжних за короткий проміжок часу [25].

Федеративне навчання дозволяє організаціям спільно тренувати моделі детектування без обміну сирими даними, зберігаючи конфіденційність та відповідаючи регуляторним вимогам. Privacy-preserving аналітика застосовує гомоморфне шифрування або secure multi-party computation для обробки зашифрованих даних, отримуючи аналітичні інсайти без розкриття вихідної інформації. Differential privacy додає калібрований шум до результатів запитів, запобігаючи ідентифікації індивідуальних записів при збереженні статистичної валідності агрегованих висновків. Інтеграція threat intelligence збагачує поведінковий аналіз контекстом про відомі індикатори компрометації, тактики атакувальників та глобальні тренди кіберзагроз. STIX/TAXII стандарти забезпечують структурований обмін інформацією про загрози між організаціями та платформами [68]. MITRE ATT&CK фреймворк категоризує тактики та техніки супротивників, дозволяючи співставити спостережувану поведінку з відомими патернами атак. Автоматизоване

збагачення алертів через зовнішні фіди intelligence додає reputation scores, історію активності IP-адрес, інформацію про домени та класифікацію malware для прискорення тріажу інцидентів [3].

Архітектури нульової довіри застосовують поведінковий аналіз для continuous authentication, постійно оцінюючи ризик кожної транзакції та динамічно адаптуючи рівень доступу. Мікросегментація мережі обмежує латеральні рухи атакувальників, вимагаючи явної авторизації для кожного міжсервісного з'єднання. Software-defined perimeter приховує інфраструктуру від несанкціонованих спостерігачів, надаючи доступ лише після верифікації ідентичності та стану пристрою. Just-in-time доступ надає привілеї на обмежений час виключно для виконання конкретних завдань, мінімізуючи вікно можливостей для зловмисної активності.

1.3 Аналіз методів кореляційного аналізу логів та виявлення аномалій у часових рядах

Лог-файли утворюють хронологічний запис подій у веб-орієнтованих системах, містячи структуровану та неструктуровану інформацію про запити користувачів, системні операції, помилки застосунків та безпекові події, потребуючи складних аналітичних підходів для екстракції значущих патернів і виявлення інцидентів безпеки. Кореляційний аналіз синтезує інформацію з гетерогенних джерел логування, ідентифікуючи причинно-наслідкові зв'язки між подіями та реконструюючи багатоетапні атаки через агрегацію фрагментованих індикаторів [26]. Часові ряди метрик безпеки демонструють складну динаміку з трендами, сезонністю та випадковими компонентами, вимагаючи спеціалізованих статистичних та машиннонавчальних методів для розпізнавання аномальних паттернів, що сигналізують про компрометацію.

Парсинг та нормалізація логів перетворюють різноформатні записи в уніфіковані структури даних, придатні для автоматизованої обробки та аналізу. Regular expressions екстрагують поля зі слабоструктурованих

текстових логів, розпізнаючи IP-адреси, timestamps, HTTP-методи та інші змінні компоненти. Grok-паттерни забезпечують бібліотеку предефінованих шаблонів для популярних форматів логів, спрощуючи конфігурацію парсерів. Log templates автоматично виявляють повторювані структури в логах через кластеризацію, генеруючи шаблони без попереднього знання формату [12]. Drain алгоритм побудови parse-дерева ефективно обробляє потокові логи в реальному часі, витягуючи параметри з варіативних частин записів.

Кореляція подій за часовими вікнами агрегує логи, що відбулися в певний інтервал, виявляючи послідовності дій або одночасні активності з різних джерел. Sliding window підходи переміщують часове вікно з певним кроком, обчислюючи агрегатні метрики або застосовуючи детекційні правила до кожного сегменту. Session-based correlation групує логи за ідентифікаторами сесій або користувачів, аналізуючи повний ланцюжок взаємодій в межах одної транзакції. Event sequence mining шукає часті послідовності подій у логах, виявляючи типові workflow та детектуючи відхилення від звичних ланцюжків операцій [44]. N-gram моделі представляють послідовності як комбінації послідовних подій, дозволяючи ймовірнісну оцінку аномальності нових ланцюжків. Кореляційні правила визначають логічні умови, за яких комбінація подій класифікується як індикатор атаки або інциденту безпеки. Complex Event Processing движки обробляють потоки подій в реальному часі, виявляючи складні патерни через темпоральні оператори, агрегації та joins між різними типами логів. Esper та Apache Flink забезпечують high-throughput обробку подієвих потоків з низькою латентністю. Drools експертні системи дозволяють декларативно специфікувати бізнес-правила та кореляційну логіку через доменно-специфічні мови. Sigma правила стандартизують опис детекційної логіки в формі YAML, забезпечуючи портабельність між різними SIEM-платформами. Основні методи кореляційного аналізу систематизовано в таблиці 1.5.

Таблиця 1.5 – Методи кореляційного аналізу логів та детектування аномалій

Метод аналізу	**Математична основа**	**Детектовані патерни**	**Параметри налаштування**	**Обчислювальна складність**
Кореляція за часовими вікнами	Агрегатні функції, threshold-based правила	Burst-активність, одночасні події з різних джерел	Розмір вікна, sliding step, aggregation function	O(n) для простих агрегацій, $O(n^2)$ для міжподієвих кореляцій
Frequent pattern mining	Apriori алгоритм, FP-growth	Часті послідовності подій, асоціативні правила	Minimum support, minimum confidence, максимальна довжина патерну	O(2^m) у гіршому випадку для m унікальних подій
Sequence alignment	Needleman-Wunsch, Smith-Waterman	Подібні ланцюжки операцій, відхилення в workflow	Gap penalty, match/mismatch scores	O(n*m) для двох послідовностей довжиною n та m
Probabilistic suffix trees	Prediction by partial matching, variable-order Markov	Аномальні переходи між станами, порушення послідовностей	Максимальна глибина дерева, мінімальна частота	O(n*d) для n подій та d максимальної глибини
Graph-based кореляція	Побудова графів залежностей, community detection	Скоординовані атаки, причинно-наслідкові ланцюжки	Метрики подібності вузлів, threshold для ребер	$O(V^2)$ для V вузлів при повних графах
Deep learning on logs	LSTM, Transformer, BERT-подібні моделі	Семантичні аномалії, контекстні відхилення	Розмір словника, довжина послідовності, embedding dimension	$O(n^2*d)$ для self-attention механізмів
Autoregressive models	ARIMA, SARIMA, Vector Autoregression	Трендові аномалії, сезонні відхилення, структурні зміни	Порядки AR/MA/I, сезонні компоненти, differencing	$O(n*p^2)$ для p порядку моделі
Spectral analysis	Fourier Transform, Wavelet Transform	Періодичні аномалії, частотні відхилення	Вікно FFT, wavelet тип, decomposition рівень	O(n*log(n)) для FFT
Change point detection	CUSUM, Bayesian change point, PELT	Моменти зміни поведінки, структурні break points	Sensitivity penalty, distribution assumptions	O(n*log(n)) для оптимальних алгоритмів

Графові моделі представляють події як вузли та їхні взаємозв'язки як ребра, дозволяючи застосовувати алгоритми теорії графів для виявлення аномальних структур. Граф атак реконструює послідовність дій атакувальника через причинно-наслідкові зв'язки між подіями з різних рівнів інфраструктури [33]. Provenance graphs відстежують походження даних та потоки інформації в системі, ідентифікуючи несанкціоновані трансформації або витоки. Heterogeneous information networks моделюють різнотипні сутності та їхні відносини, застосовуючи meta-path based аналіз для виявлення складних патернів взаємодій. Community detection алгоритми групують щільно з'єднані вузли, виявляючи координовані активності або botnet-трафік. Семантичний аналіз логів використовує обробку природної мови для інтерпретації текстових повідомлень про помилки, системних описів подій та адміністративних коментарів.

Word embeddings перетворюють лог-повідомлення у векторні простори, де семантично подібні тексти розташовуються близько, дозволяючи виявляти аномальні повідомлення через metrics подібності. BERT-подібні моделі pretrain на великих корпусах логів, навчаючись контекстним представленням лог-токенів, потім fine-tune для специфічних задач детектування аномалій. Topic modeling виявляє латентні теми в колекціях логів, групуючи схожі події та виявляючи раптові зміни в розподілі тем як потенційні інциденти.

Статистичні методи аналізу часових рядів моделюють темпоральну еволюцію метрик безпеки, прогнозуючи очікувані значення та виявляючи відхилення як аномалії. Autoregressive Integrated Moving Average моделі захоплюють автокореляційну структуру даних, параметризуючи залежності від минулих значень та помилок прогнозу. Seasonal decomposition розділяє часовий ряд на трендову, сезонну та залишкову компоненти, дозволяючи аналізувати кожну окремо [55]. Exponential smoothing надає більшу вагу недавнім спостереженням, адаптивно оновлюючи прогнози при зміні умов. Prophet від Facebook обробляє часові ряди з сильною сезонністю та відсутніми даними, автоматично виявляючи change points та holidays effects.

Спектральний аналіз перетворює часові ряди в частотну область, виявляючи домінантні періодичності та аномальні частотні компоненти. Fast Fourier Transform декомпозує сигнал на синусоїдальні складові, дозволяючи ідентифікувати періодичні атаки або зміни в ритмі нормальної активності. Wavelet transform забезпечує часово-частотну локалізацію, виявляючи транзієнтні аномалії та короткочасні відхилення. Singular Spectrum Analysis розкладає часовий ряд на трендові, осциляторні та шумові компоненти через сингулярне розкладання траєкторної матриці, ефективно фільтруючи шум та виділяючи структурні зміни.

Методи виявлення точок зміни детектують моменти, коли статистичні властивості часового ряду різко змінюються, сигналізуючи про інциденти або зміни в режимі роботи. CUSUM accumulated deviations від цільового рівня, алертуючи коли кумулятивна сума перевищує поріг. Bayesian online change point detection обчислює апостеріорну ймовірність точки зміни в кожен момент часу, адаптивно оновлюючи параметри моделі. PELT pruned exact linear time алгоритм ефективно знаходить множинні точки зміни через динамічне програмування з prune-стратегіями.

Багатовимірний часовий аналіз кореляції між множиною метрик виявляє складні аномалії, що проявляються через синхронні відхилення в різних змінних. Vector Autoregression моделює взаємозалежності між часовими рядами, захоплюючи cross-dependencies. Principal Component Analysis редукує вимірність через ортогональні трансформації, виявляючи аномалії в просторі головних компонент. Granger causality testing визначає, чи минулі значення одного ряду допомагають прогнозувати інший, встановлюючи причинні відносини між метриками. Dynamic Time Warping вимірює подібність між часовими послідовностями, що можуть бути зміщені або розтягнуті в часі, ефективно для порівняння патернів з варіативними темпоральними характеристиками.

Глибоке навчання для аналізу часових рядів застосовує рекурентні архітектури для моделювання довгострокових залежностей у метриках

безпеки. LSTM networks захоплюють тривалі темпоральні залежності через gating механізми, уникаючи проблеми vanishing gradients. Encoder-decoder архітектури навчаються стискати часові послідовності в латентні представлення, виявляючи аномалії через високу помилку реконструкції. Attention mechanisms фокусуються на релевантних частинах історії при прогнозуванні майбутніх значень, підвищуючи інтерпретованість моделей. Temporal Convolutional Networks застосовують згортки з розширеними ядрами для захоплення довгих контекстів з паралельною обробкою.

Онтології для структурування лог-даних забезпечують семантичні моделі доменів кібербезпеки, формалізуючи відносини між сутностями, подіями та атаками [13]. Knowledge graphs інтегрують гетерогенну інформацію з логів, threat intelligence та конфігурацій систем у граф знань, дозволяючи reasoning та складні запити. Semantic reasoning виводить імпліцитні факти з явних тверджень через логічні правила, виявляючи приховані індикатори компрометації. SLOGERT фреймворк автоматизує конструювання knowledge graphs з логів через онтологічні паттерни та семантичні анотації [26]. Streaming analytics обробляє логи в реальному часі, застосовуючи stateful computations до нескінченних потоків подій з гарантіями exactly-once семантики. Apache Kafka забезпечує fault-tolerant розподілене зберігання та доставку подієвих потоків з високою пропускною здатністю. Apache Flink виконує складні трансформації потоків з підтримкою event-time processing та watermarks для обробки out-of-order подій. Time-based tumbling та sliding windows агрегують події в часові відрізки, обчислюючи метрики або детектуючи патерни в межах вікон. Session windows групують події з gaps менше заданого timeout, адаптивно формуючи вікна змінної довжини відповідно до активності користувачів.

Федеративна аналітика логів дозволяє організаціям обмінюватися детекційними правилами та моделями без централізації сирих даних, зберігаючи конфіденційність та відповідаючи GDPR вимогам. Homomorphic encryption дозволяє виконувати обчислення на зашифрованих логах, отримуючи агрегатні метрики без розшифрування індивідуальних записів.

Secure multi-party computation розподіляє обчислення між кількома сторонами так, що жодна не отримує доступу до повних даних інших. Differential privacy додає калібрований шум до результатів запитів над логами, запобігаючи реідентифікації індивідуальних подій при публікації статистик.

Візуалізація логів та часових рядів трансформує великі обсяги даних у інтерактивні графічні представлення, полегшуючи exploratory analysis та комунікацію результатів. Heatmaps відображають інтенсивність подій у двовимірній сітці часу та категорій, виявляючи спалахи активності. Sankey diagrams візуалізують потоки між джерелами та призначеннями, показуючи розподіл трафіку або послідовності переходів. Parallel coordinates plots дозволяють аналізувати багатовимірні дані через паралельні осі, ідентифікуючи кластери та outliers. Force-directed graphs інтерактивно розміщують вузли подієвих графів, групуючи щільно з'єднані компоненти та виявляючи аномальні зв'язки.

Human-in-the-loop підходи інтегрують експертизу аналітиків у процес детектування, використовуючи машинні моделі для пріоритизації подій та людський інтелект для валідації складних випадків. Active learning запитує мітки для найбільш невизначених прикладів, ефективно покращуючи моделі з мінімальними зусиллями анотування. Explainable AI техніки генерують інтерпретовані пояснення детекційних рішень через SHAP values, attention weights або rule extraction, підвищуючи довіру аналітиків до автоматизованих систем. Interactive machine learning дозволяє аналітикам корегувати моделі в реальному часі, надаючи повідомлення про помилки та інкорпоруючи доменні знання безпосередньо в алгоритми навчання.

1.4 Висновки до розділу

У першому розділі магістерської кваліфікаційної роботи було проведено ґрунтовний аналіз сучасного стану захищеності веб-орієнтованих інформаційних систем. Розглянуто та систематизовано класифікацію

прикладних атак, що дозволило виявити тенденцію до ускладнення векторів загроз — від традиційних ін'єкцій до складних багатоетапних сценаріїв, які експлуатують бізнес-логіку та API.

В ході дослідження існуючих методів захисту було встановлено, що традиційні сигнатурні підходи втрачають ефективність проти нових типів атак. Натомість, динамічний поведінковий аналіз на основі машинного навчання демонструє значний потенціал, проте його ізольоване застосування часто супроводжується високим рівнем хибних спрацювань. Проведений огляд методів кореляційного аналізу логів та детестування аномалій у часових рядах вказав на необхідність їх поєднання для досягнення синергетичного ефекту в виявленні загроз.

На основі проведеного аналізу було визначено мету дослідження та сформульовано ключові завдання для подальшої розробки: — розробити концептуальну модель системи захисту, що інтегрує кореляційний аналіз логів та аналіз часових рядів; — здійснити програмну реалізацію запропонованого підходу для перевірки його працездатності; — провести експериментальне дослідження ефективності розробленого рішення та здійснити порівняльний аналіз з існуючими аналогами.

РОЗДІЛ 2. РОЗРОБКА УДОСКОНАЛЕНОГО МЕТОДУ ДИНАМІЧНОГО ПОВЕДІНКОВОГО АНАЛІЗУ ДЛЯ ЗАХИСТУ ВЕБ-ДОДАТКІВ

2.1 Концептуальна модель системи захисту на основі динамічного поведінкового аналізу та математична модель кореляційного аналізу логів

Веб-орієнтовані інформаційні системи потребують нових підходів до забезпечення безпеки, оскільки традиційні методи виявлення атак базуються переважно на сигнатурному аналізі та статичних правилах фільтрації трафіку. Динамічний поведінковий аналіз дозволяє виявляти аномалії у функціонуванні додатків шляхом постійного моніторингу їхньої активності та порівняння поточних параметрів із базовими профілями нормальної поведінки. Концептуальна модель системи захисту включає декілька взаємопов'язаних компонентів, кожен з яких виконує специфічні функції у загальному процесі детектування загроз (рис. 2.1).

Архітектура запропонованої системи базується на багаторівневій обробці даних, де первинний рівень відповідає за збір логів із різноманітних джерел інформації. Веб-сервери генерують записи про HTTP-запити користувачів, системи управління базами даних фіксують SQL-запити та операції читання-запису, а програмні модулі додатків створюють власні журнали подій із деталізацією внутрішніх процесів обробки бізнес-логіки. Агрегація логів відбувається через централізовану систему збору даних, яка нормалізує формати записів і забезпечує уніфіковане представлення інформації для подальшого аналізу. Нормалізація передбачає перетворення різнорідних структур даних у єдиний формат із визначеними полями часової мітки, ідентифікатора користувача, типу операції, параметрів запиту та результату виконання.

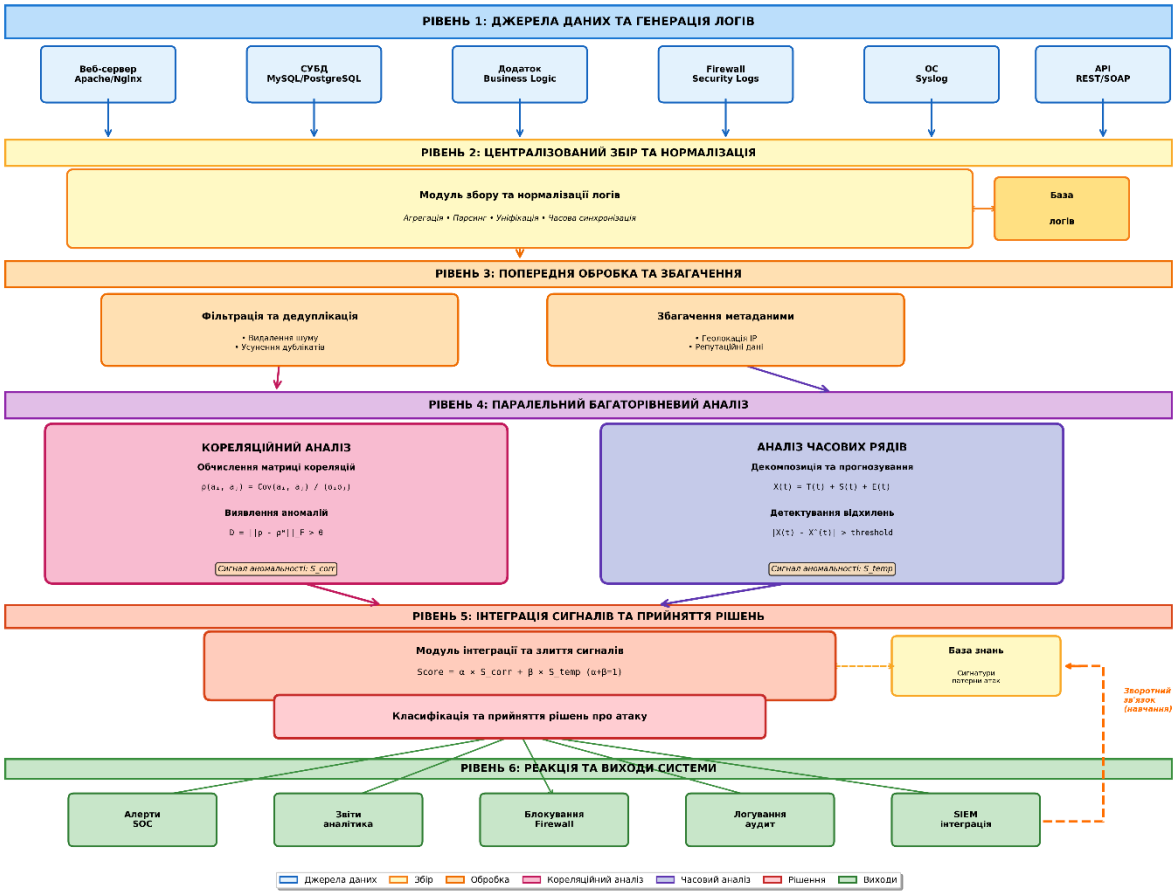


Рис. 2.1 Концептуальна схема системи захисту

Другий рівень архітектури реалізує попередню обробку зібраних логів через фільтрацію шумових даних та видалення дублікатів записів. Фільтрація здійснюється на основі евристичних правил, які дозволяють відсіювати технічні записи системного характеру, що не несуть інформаційної цінності для аналізу безпеки. Видалення дублікатів запобігає спотворенню статистичних характеристик даних, які можуть виникати внаслідок повторної реєстрації ідентичних подій різними компонентами системи. Паралельно відбувається збагачення даних додатковою контекстною інформацією, зокрема геолокацією IP-адрес, репутаційними оцінками джерел запитів, історичними профілями активності користувачів. Збагачення даних підвищує інформативність логів і створює передумови для більш точного виявлення аномальних патернів поведінки.

Математична модель кореляційного аналізу логів ґрунтується на виявленні статистичних залежностей між різними типами подій у часі. Нехай $L = \{l_1, l_2, ..., l_n\}$ позначає множину записів логів, де кожен запис l_i характеризується набором атрибутів $A = \{a_1, a_2, ..., a_m\}$. Атрибути включають часову мітку t, ідентифікатор користувача u, тип операції o, параметри запиту p та код відповіді r. Для кожної пари атрибутів визначається коефіцієнт кореляції Пірсона, який обчислюється за формулою:

$$\rho(a_i, a\square) = \Sigma\square \frac{(a_i\square - \mu_i)(a\square\square - \mu\square)}{\sigma_i\sigma\square}$$

Середнє значення атрибута μ_i обчислюється як арифметична середня всіх значень даного атрибута у вибірці логів, а стандартне відхилення σ_i характеризує розкид значень відносно середнього. Коефіцієнт кореляції приймає значення від -1 до 1, де значення близькі до 1 свідчать про сильний прямий зв'язок між атрибутами, значення близькі до -1 вказують на обернену залежність, а значення близькі до 0 означають відсутність лінійного зв'язку. Кореляційний аналіз дозволяє виявити нетипові комбінації атрибутів, які можуть сигналізувати про спроби атак.

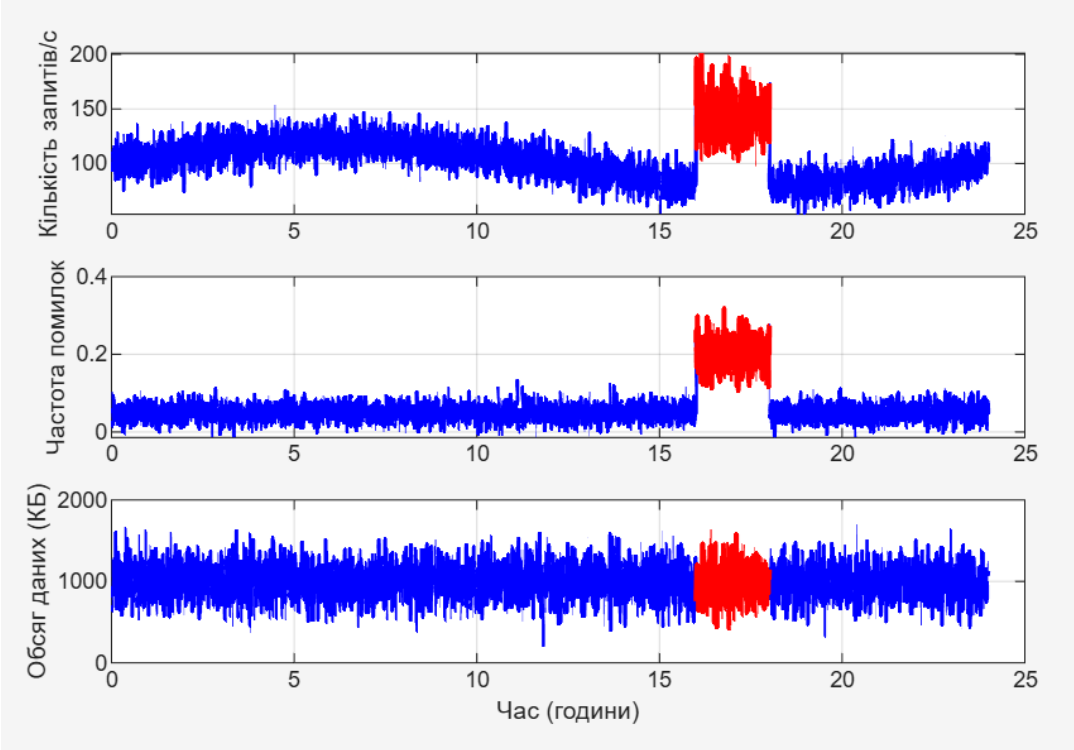


Рис. 2.2 Часові ряди метрик

Часові ряди представляють собою послідовності значень певної метрики, упорядковані за часовими мітками. Для аналізу динаміки поведінки веб-додатків формуються часові ряди різних метрик, таких як кількість запитів на одиницю часу, середній час відгуку системи, частота помилок, обсяг переданих даних (рис. 2.2). Формально часовий ряд описується як функція X(t), де t належить інтервалу спостереження T = [t_0, t_n]. Дискретизація часу дозволяє представити часовий ряд у вигляді послідовності X = {x_1, x_2, ..., x_n}, де x_i відповідає значенню метрики у момент часу t_i.

Аналіз автокореляційної функції часового ряду виявляє періодичність та трендові складові у динаміці метрик. Автокореляційна функція визначається як коефіцієнт кореляції між значеннями ряду та його зсувами на k кроків у часі:

$$R(k) = \frac{\Sigma_i\big((x_i - \mu)(x_{i+k} - \mu)\big)}{\Sigma_i(x_i - \mu)^2}$$

Значення автокореляційної функції для різних лагів k утворюють автокореляційний портрет часового ряду, який характеризує внутрішню структуру даних. Високі значення автокореляції на певних лагах вказують на наявність циклічних компонентів у поведінці системи, що може бути природною характеристикою робочого навантаження або ознакою автоматизованих атак. Виявлення аномальних відхилень від очікуваних автокореляційних характеристик становить основу методу детектування загроз. Візуалізація матриць кореляцій представлена на рис. 2.3.

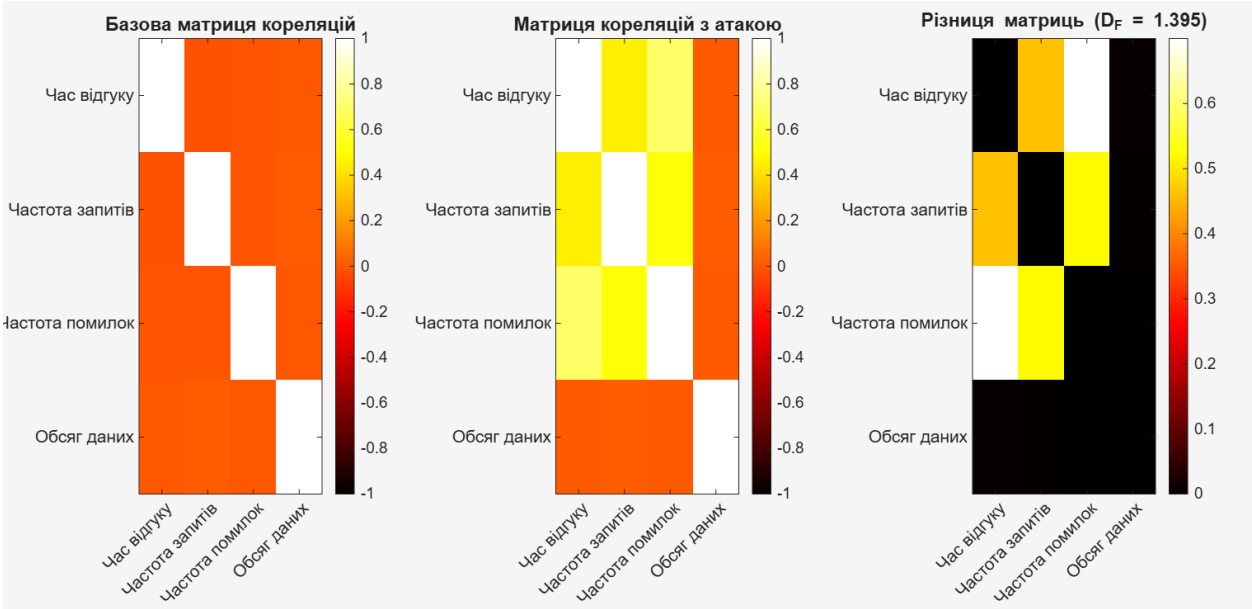


Рис. 2.3 Матриці кореляцій розробленої моделі

Побудова базового профілю нормальної поведінки потребує навчання моделі на історичних даних, які не містять атак. Профіль формується як множина статистичних характеристик часових рядів різних метрик, зокрема математичного сподівання, дисперсії, коефіцієнтів асиметрії та ексцесу, спектральної щільності потужності. Математичне сподівання E[X] характеризує центральну тенденцію розподілу значень метрики, дисперсія Var[X] відображає міру розкиду даних навколо середнього. Коефіцієнт асиметрії оцінює симетричність розподілу відносно математичного сподівання:

$$\gamma^1 = \frac{E[(X - \mu)^3]}{\sigma^3}$$

Коефіцієнт ексцесу характеризує гостроту піку розподілу порівняно з нормальним розподілом:

$$\gamma^2 = \frac{E[(X - \mu)^4]}{\sigma^4} - 3$$

Відхилення поточних значень статистичних характеристик від базового профілю свідчить про аномальну поведінку системи. Для кількісної оцінки ступеня аномальності запроваджується метрика відстані між поточним станом

та базовим профілем. Евклідова відстань у просторі статистичних характеристик обчислюється як:

$$D = \sqrt{\Sigma_j (S_j - S^0_j)^2}$$

Параметр S_j позначає поточне значення j-ї статистичної характеристики, а S^0_j відповідає значенню характеристики у базовому профілі. Перевищення відстанею D заздалегідь встановленого порогу θ ініціює генерацію сигналу тривоги про потенційну атаку. Визначення оптимального значення порогу θ здійснюється експериментально шляхом мінімізації сумарної помилки першого та другого роду на навчальній вибірці даних (рис. 2.4).

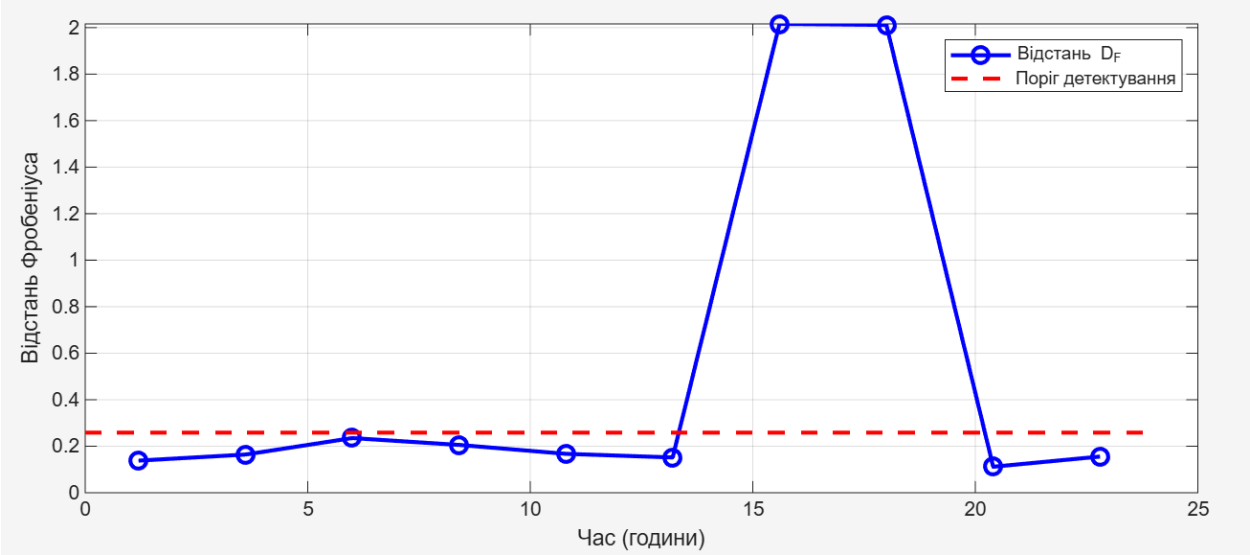


Рис. 2.4 Динаміка відхилень кореляційної структури

Крос-кореляційний аналіз між різними часовими рядами метрик дозволяє виявляти складні атаки, які проявляються через узгоджені зміни декількох параметрів системи одночасно. Функція крос-кореляції для двох часових рядів X та Y визначається як:

$$R_{xy}(k) = \frac{\Sigma_i \left((x_i - \mu_x)(y_i^+ - \mu_y) \right)}{\sqrt{\Sigma_i (x_i - \mu_x)^2 \Sigma_i (y_i - \mu_y)^2}}$$

де $R_{xy}(k)$ - Коефіцієнт крос-кореляції.

x_i - значення першого часового ряду в момент часу i

Лаг k, при якому функція крос-кореляції досягає максимального значення, вказує на часову затримку між змінами метрик X та Y. Аналіз крос-кореляційних зв'язків формує граф залежностей між метриками, де вершинами виступають метрики, а ребра відображають статистично значущі кореляційні зв'язки. Порушення структури графа залежностей порівняно з базовим профілем сигналізує про аномальні процеси у системі (табл. 2.1).

Таблиця 2.1 – Основні компоненти концептуальної моделі системи захисту

Компонент	**Функціональне призначення**	**Вхідні дані**	**Вихідні дані**
Модуль збору логів	Агрегація журналів подій із веб-серверів, СУБД, додатків	Лог-файли різних форматів	Нормалізовані записи у єдиному форматі
Модуль попередньої обробки	Фільтрація, дедуплікація, збагачення даних	Нормалізовані лог-записи	Оброблені лог-записи з додатковою метаінформацією
Модуль кореляційного аналізу	Обчислення коефіцієнтів кореляції між атрибутами	Оброблені лог-записи	Матриця кореляцій атрибутів
Модуль аналізу часових рядів	Формування та аналіз динаміки метрик	Послідовності значень метрик	Статистичні характеристики часових рядів
Модуль виявлення аномалій	Порівняння поточного стану з базовим профілем	Поточні статистичні характеристики	Оцінка аномальності та сигнали тривоги

Спектральний аналіз часових рядів надає інформацію про частотні компоненти у динаміці метрик. Перетворення Фур'є переводить часовий ряд із часової області у частотну область, де кожна частотна компонента характеризується амплітудою та фазою. Періодограма представляє спектральну щільність потужності часового ряду як функцію частоти:

$$P(f) = \frac{|\Sigma(x \cdot \exp(-j2\pi f n \Delta t))|^2}{N}$$

де $P(f)$ - спектральна щільність потужності (Periodogram). Показує, наскільки сильною є активність на певній частоті f.

x - Значення $n-$ го відліку часового ряду

Частота f виміряється у герцах, Δt позначає крок дискретизації часу, N відповідає кількості відліків у часовому ряді, а j означає уявну одиницю. Аналіз періодограми виявляє домінуючі частоти у поведінці системи, що дозволяє ідентифікувати регулярні цикли активності користувачів. Аномальні атаки часто характеризуються появою нових частотних компонент або зникненням природних частот, що відображається у зміні структури спектра потужності.

Ентропійний аналіз логів оцінює міру невизначеності у розподілі значень атрибутів та дозволяє виявляти аномалії через зміни інформаційного змісту подій. Інформаційна ентропія Шеннона для дискретного розподілу ймовірностей обчислюється як:

$H(X) = -\Sigma_i \left(p(x_i) log^2 \left(p(x_i)\right)\right)$ ймовірність $p(x_i)$ визначається як відносна частота появи значення x_i у вибірці даних. Високі значення ентропії свідчать про рівномірний розподіл значень атрибута, тоді як низькі значення вказують на концентрацію даних навколо певних значень. Атаки типу сканування портів або перебору паролів характеризуються специфічними змінами ентропії відповідних атрибутів логів. Моніторинг динаміки ентропії у часі формує додатковий канал виявлення аномальної активності.

Кластеризація записів логів групує подібні події для виявлення патернів нормальної та аномальної поведінки. Алгоритм k-середніх розбиває множину логів на k кластерів шляхом мінімізації внутрішньокластерної дисперсії:

$$J = \Sigma \Sigma_i \in C \; ||l_i - \mu||^{22}$$

Центроїд кластера μ_k обчислюється як середнє значення всіх записів, що належать кластеру C_k. Ітеративний процес оптимізації функції вартості J забезпечує збіжність до локального мінімуму. Нові записи логів класифікуються відповідно до найближчого центроїда, а відстань до центроїда використовується як міра аномальності події. Записи, що значно віддалені від усіх центроїдів, класифікуються як аномальні та підлягають детальному розгляду.

Побудова ланцюгів подій на основі кореляційного аналізу реконструює сценарії багатоетапних атак. Направлений граф подій представляє часову послідовність логів, де вершини відповідають окремим подіям, а ребра відображають каузальні зв'язки між ними. Каузальність визначається через кореляційні залежності атрибутів подій та часову близькість їх виникнення. Ідентифікація підозрілих ланцюгів подій базується на порівнянні їх структури з відомими патернами атак, збереженими у базі знань системи. Машинне навчання застосовується для автоматичної класифікації ланцюгів як легітимних або шкідливих на основі навчальних прикладів.

Адаптивний поріг виявлення аномалій коригується динамічно залежно від поточного стану системи та статистичних характеристик трафіку. Фіксований поріг не враховує природних коливань навантаження на систему протягом доби або тижня, що призводить до підвищення кількості хибних спрацювань. Адаптивний поріг визначається як функція від ковзного середнього та стандартного відхилення метрики:

$$\theta(t) = \mu(t) + k\,\sigma(t)$$

Коефіцієнт k регулює чутливість детектора і встановлюється експериментально для досягнення балансу між частотою пропусків атак та хибних спрацювань. Ковзне середнє μ(t) та стандартне відхилення σ(t) обчислюються на вікні останніх N спостережень, що забезпечує відстеження поточних тенденцій у поведінці системи. Експоненційне згладжування надає більшу вагу недавнім спостереженням порівняно з віддаленими у часі:

$$\mu(t) = \alpha\, x(t) + (1 - \alpha)\mu(t-1)$$

Параметр α визначає швидкість адаптації моделі до змін у даних і приймає значення з інтервалу від 0 до 1. Малі значення α забезпечують стабільність моделі до короткочасних флуктуацій, великі значення підвищують чутливість до змін.

Інтеграція результатів кореляційного аналізу логів та аналізу часових рядів формує комплексну систему детектування загроз з підвищеною точністю виявлення атак. Кореляційний аналіз ідентифікує аномальні комбінації

атрибутів подій, тоді як аналіз часових рядів виявляє відхилення у динаміці метрик від очікуваних значень. Об'єднання сигналів від обох підсистем через логіку голосування або ймовірнісні моделі знижує ймовірність хибних спрацювань і підвищує достовірність детектування справжніх атак.

Модель загроз для веб-додатків включає широкий спектр атак, від ін'єкцій SQL-коду та міжсайтового скриптингу до атак відмови у обслуговуванні та несанкціонованого доступу до даних. Кожен тип атаки характеризується специфічними патернами у логах та часових рядах метрик. SQL-ін'єкції проявляються через аномальні символьні послідовності у параметрах запитів до бази даних, підвищену частоту помилок виконання запитів, незвичні обсяги даних у відповідях. Атаки міжсайтового скриптингу відображаються у наявності HTML- або JavaScript-коду у вхідних параметрах користувачів, спробах виконання сценаріїв у контексті інших користувачів. Валідація моделі здійснюється на незалежній тестовій вибірці даних, яка містить як легітимний трафік, так і приклади атак. Метрики якості детектування включають точність, повноту, F-міру та площу під ROC-кривою. Точність визначає частку правильно ідентифікованих атак серед усіх подій, класифікованих як атаки. Повнота відображає частку виявлених атак серед усіх реальних атак у тестовій вибірці. F-міра представляє гармонійне середнє точності та повноти:

$$F = 2\frac{Precision \times Recall}{Precision + Recall}$$

ROC-крива будується шляхом варіювання порогу детектування та відображає залежність між часткою правильно виявлених атак та часткою хибних спрацювань. Площа під ROC-кривою надає інтегральну оцінку якості класифікатора і приймає значення від 0.5 для випадкового класифікатора до 1.0 для ідеального детектора.

Продуктивність системи захисту критично важлива для забезпечення прийнятної затримки обробки запитів користувачів. Обчислювальна складність кореляційного аналізу залежить квадратично від кількості атрибутів логів, що обмежує можливість обробки великих обсягів даних у

реальному часі. Оптимізація обчислень досягається через попередній відбір найбільш інформативних атрибутів, паралелізацію обробки на багатоядерних процесорах, використання апаратних прискорювачів. Інкрементальні алгоритми обробки дозволяють оновлювати статистичні характеристики без повторного перерахунку на всьому історичному масиві даних.

Масштабованість архітектури забезпечується розподіленою обробкою даних на кластері серверів з балансуванням навантаження між вузлами. Логи розбиваються на фрагменти за часовими інтервалами або джерелами, кожен фрагмент обробляється незалежно на окремому вузлі кластера. Проміжні результати аналізу агрегуються централізованим компонентом, який формує глобальну картину стану безпеки системи. Горизонтальне масштабування шляхом додавання нових вузлів до кластера дозволяє лінійно збільшувати пропускну здатність системи відповідно до зростання обсягів трафіку.

2.2 Розробка методу аналізу часових рядів для виявлення прикладних атак та алгоритмів класифікації типів атак

Прикладні атаки на веб-орієнтовані інформаційні системи характеризуються складними патернами поведінки, які проявляються через зміни у часових характеристиках обробки запитів, розподілі навантаження на компоненти системи та динаміці використання ресурсів. Традиційні методи статичного аналізу не здатні адекватно виявляти атаки, що розгортаються поступово протягом тривалих часових інтервалів або маскуються під легітимну активність користувачів. Розробка методу аналізу часових рядів передбачає формування багатовимірного простору ознак, кожна з яких описує специфічний аспект поведінки системи у часі, та застосування алгоритмів машинного навчання для класифікації спостережуваних патернів як нормальних або шкідливих. Формування вхідних часових рядів здійснюється шляхом агрегації первинних метрик системи з фіксованою частотою дискретизації. Інтервал дискретизації обирається таким чином, щоб

забезпечити баланс між детальністю представлення динаміки процесів та обчислювальною складністю аналізу. Занадто малий інтервал призводить до надмірної деталізації з високим рівнем шуму від випадкових флуктуацій, тоді як великий інтервал може пропустити короткочасні аномальні події. Експериментальні дослідження показують оптимальність інтервалу від одної до десяти секунд для більшості веб-додатків залежно від інтенсивності трафіку.

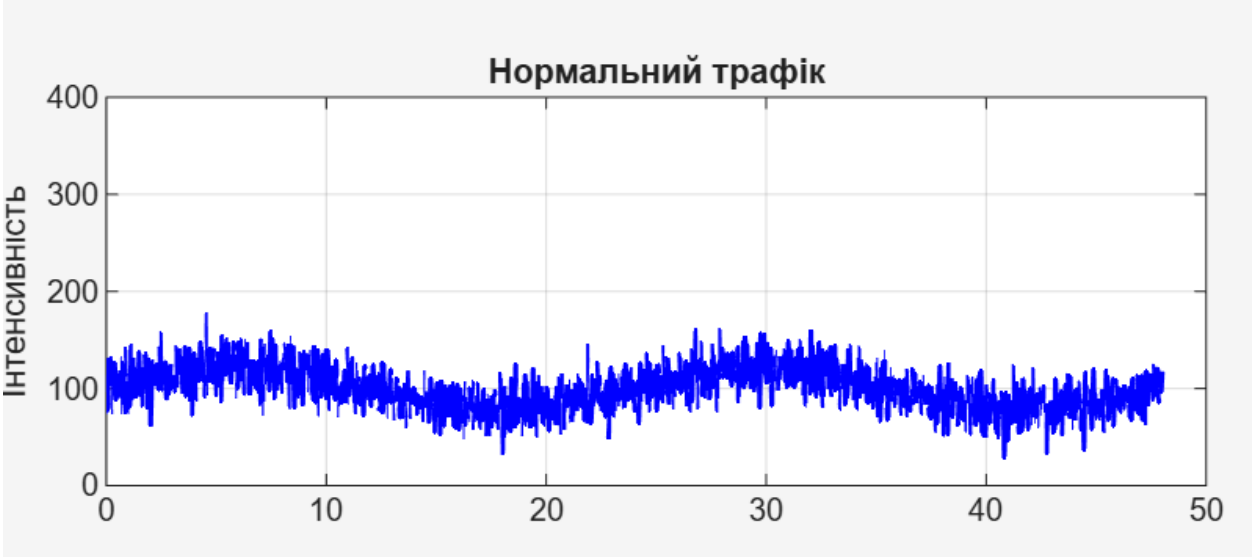


Рис. 2.5 Часовий ряд нормального трафіку

Множина базових метрик включає кількість HTTP-запитів за одиницю часу, середню тривалість обробки запитів, розподіл HTTP-кодів відповіді, обсяг переданих даних у байтах, кількість унікальних IP-адрес джерел запитів, частоту звернень до критичних ресурсів додатку. Кожна метрика формує окремий часовий ряд з власними статистичними характеристиками та автокореляційною структурою. Паралельно збираються метрики рівня додатку, такі як кількість операцій читання та запису до бази даних, тривалість виконання бізнес-логіки, частота виникнення програмних винятків, обсяг використаної оперативної пам'яті процесами додатку.

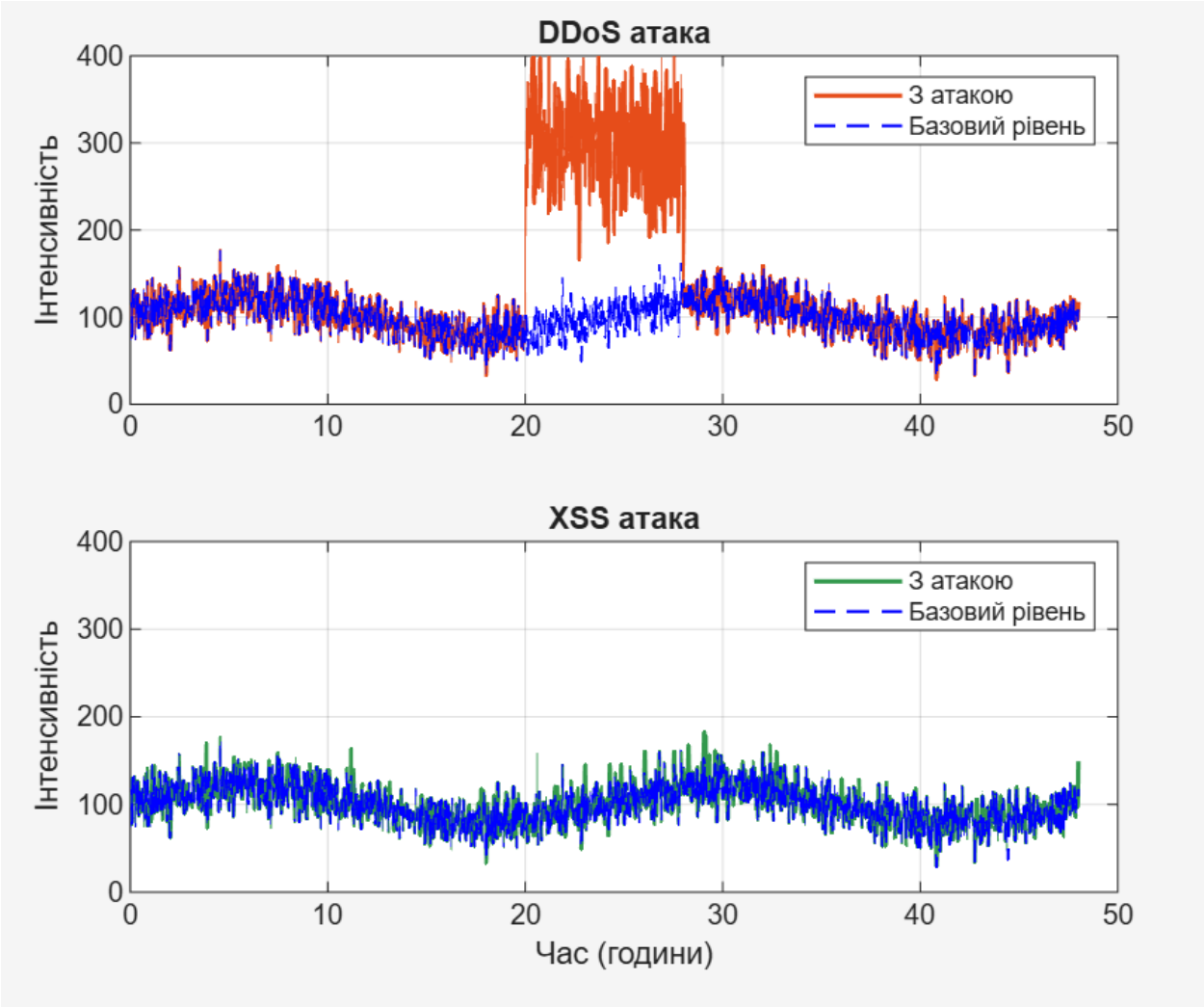


Рис. 2.6 Часовий ряд трафіку під DDoS та XSS атаками

Декомпозиція часових рядів на трендову, сезонну та випадкову компоненти дозволяє ізолювати аномальні відхилення від природних коливань метрик. Адитивна модель декомпозиції представляє спостережуване значення як суму компонентів:

$$X(t) = T(t) + S(t) + E(t)$$

Трендова компонента T(t) відображає довгострокову тенденцію зміни метрики, сезонна компонента S(t) описує періодичні коливання з фіксованим періодом, залишкова компонента E(t) містить випадкові флуктуації та аномальні відхилення. Виділення трендової компоненти здійснюється методом ковзного середнього з вікном, що відповідає періоду сезонності. Сезонна компонента обчислюється як середнє відхилення від тренду для

кожної точки сезонного циклу. Залишкова компонента отримується вирахуванням тренду та сезонності з вихідного ряду.

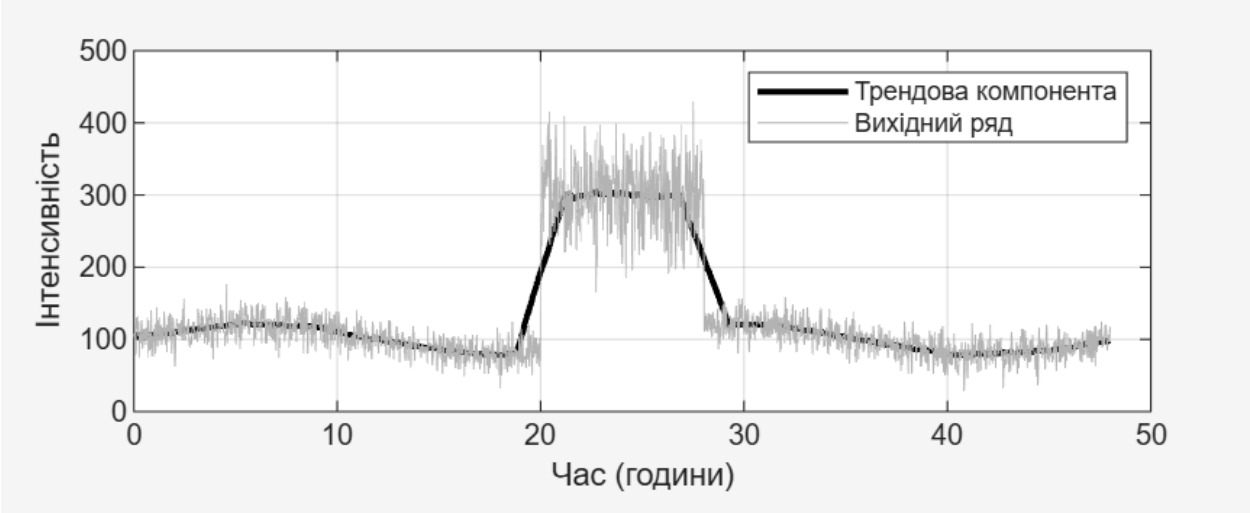


Рис. 2.7 Декомпозиція атак, тренд

Аналіз залишкової компоненти формує основу детектування аномалій, оскільки вона містить відхилення від очікуваної поведінки після виключення систематичних складових. Статистичний контроль процесу застосовує контрольні карти Шухарта для моніторингу залишкової компоненти. Верхня та нижня контрольні межі визначаються як відстань трьох стандартних відхилень від середнього значення залишків:

$$UCL = \mu_E + 3\sigma_E$$

$$LCL = \mu_E - 3\sigma_E$$

Вихід значень залишкової компоненти за межі контрольних границь сигналізує про статистично значущу аномалію у поведінці системи. Правила Western Electric доповнюють базовий критерій додатковими патернами виявлення аномалій, такими як серія послідовних точок по один бік від середньої лінії, монотонний тренд у залишках, циклічні коливання нехарактерного періоду.

Моделі авторегресії та ковзного середнього надають можливість прогнозування майбутніх значень часових рядів на основі історичних спостережень. ARIMA модель об'єднує авторегресійну компоненту порядку p,

інтегрування порядку d для забезпечення стаціонарності та компоненту ковзного середнього порядку q. Математичний запис ARIMA(p,d,q) моделі має вигляд:

$$\varphi(B)(1-B)^d X(t) = \theta(B)\varepsilon(t)$$

Оператор зсуву B діє на часовий ряд відповідно до правила B X(t) = X(t-1), поліном авторегресії φ(B) визначає залежність від минулих значень ряду, поліном ковзного середнього θ(B) моделює вплив попередніх помилок прогнозу, випадкова величина ε(t) представляє білий шум з нульовим середнім. Ідентифікація порядків моделі здійснюється через аналіз автокореляційної та частково автокореляційної функцій, а оцінювання параметрів виконується методом максимальної правдоподібності.

Прогнозні значення метрик порівнюються з фактично спостережуваними для обчислення помилки прогнозу. Аномально великі помилки прогнозу вказують на відхилення поточної поведінки системи від очікуваних патернів, змодельованих на історичних даних. Відносна помилка прогнозу нормалізує абсолютне відхилення на масштаб значень метрики:

$$\varepsilon_{rel(t)} = \frac{\left|X(t) - \hat{X}(t)\right|}{|X(t)|}$$

Накопичення великих помилок прогнозу протягом кількох послідовних часових інтервалів з високою вірогідністю свідчить про початок атаки або несправність системи. Порогове значення відносної помилки встановлюється на рівні, що забезпечує компроміс між швидкістю виявлення атак та частотою хибних тривог. У таблиці 2.2 наведено характерні ознаки різних атак у часових рядах.

Таблиця 2.2 – Характеристики часових рядів для різних типів прикладних атак

Тип атаки	**Метрики з аномаліями**	**Характер відхилень**	**Тривалість прояву**	**Складність детектування**
SQL-ін'єкція	Час відгуку БД, частота помилок SQL	Різкі сплески тривалості запитів, підвищення коду помилок 500	Короткочасна (секунди-хвилини)	Середня
XSS-атака	Розмір параметрів запитів, частота спеціальних символів	Аномальні довжини GET/POST параметрів, збільшення ентропії вхідних даних	Епізодична (окремі запити)	Висока
DDoS	Кількість запитів, унікальні IP-адреси, використання смуги пропускання	Експоненційне зростання трафіку, знижена різноманітність джерел	Тривала (хвилини-години)	Низька
Brute force	Частота невдалих автентифікацій, джерела запитів на вхід	Високочастотні спроби входу з одного джерела, монотонне зростання лічильника	Середня (хвилини-десятки хвилин)	Низька
Підбір даних	Частота доступу до API, різноманітність запитуваних ідентифікаторів	Систематичне сканування простору ідентифікаторів, рівномірний розподіл запитів	Тривала (години-дні)	Висока
Привілейована ескалація	Паттерни доступу до адміністративних функцій, аномальні послідовності операцій	Нетипові комбінації викликів методів, доступ до ресурсів поза звичайним профілем	Короткочасна (секунди-хвилини)	Дуже висока

Багатовимірний аналіз часових рядів розглядає одночасно декілька метрик для виявлення складних атак, які не проявляються через аномалії окремих показників. Векторна авторегресійна модель VAR узагальнює ARIMA підхід на випадок множини взаємопов'язаних часових рядів.

Математична форма VAR моделі порядку p для n-вимірного вектора часових рядів X(t) записується як:

$$X(t) = A^1X(t-1) + A^2X(t-2) + \ldots + A^pX(t-p) + \varepsilon(t)$$

Матриці коефіцієнтів A_i розміром n×n визначають взаємний вплив компонентів вектора часових рядів один на одного з лагом i. Оцінювання параметрів моделі виконується методом найменших квадратів окремо для кожного рівняння системи. Тестування причинності за Грейнджером перевіряє гіпотезу про наявність статистично значущого впливу одного часового ряду на інший через порівняння якості прогнозу з включенням та виключенням відповідних лагових змінних.

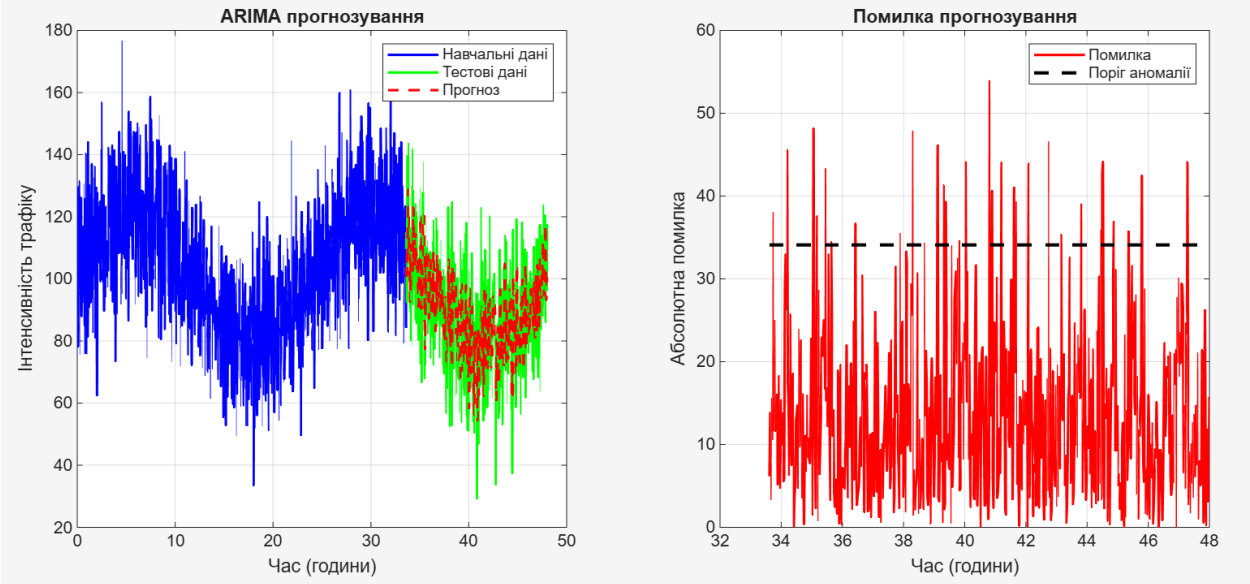


Рис. 2.8 Графіки ARIMA прогнозування

Спектральний аналіз багатовимірних часових рядів виявляє кореляції між частотними компонентами різних метрик. Взаємна спектральна щільність для пари часових рядів X та Y обчислюється через перетворення Фур'є їхньої взаємокореляційної функції. Когерентність між рядами на частоті f визначає міру лінійного зв'язку у частотній області:

$$C(f) = \frac{|S_{XY(f)}|^2}{S_{XX(f)S_{YY(f)}}}$$

Функція когерентності приймає значення від 0 до 1, де значення близькі до 1 свідчать про сильну кореляцію компонентів на відповідній частоті. Аналіз когерентності виявляє координовані зміни різних метрик, характерні для певних типів атак. Розподілені атаки відмови у обслуговуванні проявляються через синхронне зростання трафіку та навантаження на процесор на частотах, що відповідають періодичності хвиль атаки.

Нелінійні методи аналізу часових рядів застосовуються для виявлення складних залежностей, які не описуються лінійними моделями. Реконструкція фазового простору методом затримок перетворює одновимірний часовий ряд у траєкторію у багатовимірному просторі стану. Вектор стану формується з послідовних значень ряду, зсунутих на час затримки τ:

$$Y(t) = [X(t), X(t+\tau), X(t+2\tau), \dots, X(t+(m-1)\tau)]$$

Розмірність вкладення m обирається за методом хибних найближчих сусідів, час затримки τ визначається з першого мінімуму функції взаємної інформації або через поріг декореляції автокореляційної функції. Аналіз геометрії атрактора у фазовому просторі характеризує динамічні властивості системи. Кореляційна розмірність атрактора оцінює фрактальну структуру множини станів:

$$D^2 = \lim(r \to 0)\left[\frac{\log C(r)}{\log r}\right]$$

Кореляційний інтеграл C(r) обчислює частку пар точок траєкторії, відстань між якими не перевищує r. Зміна кореляційної розмірності під час атаки відображає перехід системи до якісно іншого режиму функціонування з відмінною структурою динаміки.

Показники Ляпунова квантифікують чутливість траєкторії до малих збурень початкових умов і характеризують наявність хаотичної динаміки. Додатній максимальний показник Ляпунова вказує на експоненційне розбігання близьких траєкторій і свідчить про детерміністичний хаос у поведінці системи. Алгоритм Розенштейна оцінює максимальний показник Ляпунова через середню швидкість розбігання найближчих сусідів у

реконструйованому фазовому просторі. Аномальні зміни показників Ляпунова сигналізують про порушення звичайної динаміки системи внаслідок зовнішнього втручання.

Рекурентні діаграми візуалізують повторення станів у фазовому просторі та виявляють структурні зміни у динаміці часових рядів. Матриця рекурентності R формується шляхом бінаризації матриці відстаней між усіма парами точок траєкторії з порогом ε. Елемент матриці приймає одиничне значення, якщо відстань між станами менша за поріг:

$$R(i,j) = \Theta\left(\varepsilon - \left||Y(i) - Y(j)|\right|\right)$$

Функція Хевісайда Θ дорівнює одиниці для додатних аргументів та нулю для від'ємних. Візуальний аналіз рекурентної діаграми виявляє діагональні лінії, що відповідають детерміністичним структурам у динаміці, та суцільні блоки, які вказують на ламінарні стани з малою варіабельністю. Міри рекурентного кількісного аналізу, такі як частка рекурентності, детермінізм, ентропія розподілу довжин діагональних ліній, характеризують властивості динаміки кількісно.

Алгоритми класифікації типів атак будуються на основі навчання з учителем, де моделі тренуються на розміченій вибірці часових рядів з відомими мітками класів. Екстракція ознак перетворює вихідні часові ряди у вектори дескрипторів фіксованої розмірності, придатних для подання на вхід класифікаторів. Набір ознак включає статистичні моменти розподілу значень, коефіцієнти авторегресійних моделей, параметри спектральної щільності, нелінійні характеристики динаміки. Автоматична генерація великої кількості ознак через бібліотеки tsfresh або Catch22 з подальшим відбором інформативних дескрипторів методами фільтрації або обгортки підвищує якість класифікації.

Дерева рішень формують інтерпретовані моделі класифікації через послідовність бінарних розбиттів простору ознак. Кожен внутрішній вузол дерева реалізує перевірку умови на значення однієї з ознак, листові вузли

містять мітки класів. Алгоритм побудови дерева рекурсивно розбиває навчальну вибірку, обираючи на кожному кроці ознаку та поріг розбиття, що максимізують чистоту отриманих підмножин. Критерій Джині вимірює неоднорідність підмножини як ймовірність неправильної класифікації випадково обраного елемента:

$$Gini = 1 - \Sigma_i p_i^2$$

Відносна частота p_i обчислюється для кожного класу у підмножині. Pruning дерева запобігає перенавчанню шляхом видалення гілок, що не покращують якість класифікації на валідаційній вибірці. Інтерпретовність дерев рішень дозволяє експертам аналізувати логіку класифікації та уточнювати набори ознак.

Ансамблі дерев рішень комбінують прогнози множини базових класифікаторів для підвищення точності та стабільності. Random Forest будує набір дерев на випадкових підвибірках навчальних даних з випадковим відбором підмножини ознак у кожному вузлі. Фінальне рішення приймається голосуванням більшості серед прогнозів окремих дерев. Gradient Boosting послідовно додає нові дерева, кожне з яких навчається на залишках помилок попередніх моделей. Адаптивне налаштування ваг навчальних прикладів концентрує увагу нових дерев на складних для класифікації випадках [56].

Методи опорних векторів знаходять оптимальну гіперплощину поділу між класами у просторі ознак з максимізацією відстані до найближчих об'єктів кожного класу. Задача оптимізації формулюється як квадратичне програмування з обмеженнями типу нерівностей. Kernel trick дозволяє будувати нелінійні розділяючі поверхні через неявне відображення вхідних векторів у простір вищої розмірності. Радіальна базисна функція ядра обчислює міру подібності між об'єктами як функцію від евклідової відстані:

$$K(x, y) = \exp\left(-\gamma \left|\left|x - y\right|\right|^2\right)$$

Параметр γ контролює ширину гауссіану та впливає на гладкість розділяючої поверхні. Багатокласова класифікація реалізується стратегіями

один-проти-всіх або один-проти-одного з побудовою множини бінарних класифікаторів.

Нейронні мережі забезпечують автоматичне навчання ієрархії ознак безпосередньо з вихідних часових рядів без ручної інженерії дескрипторів. Згорткові нейронні мережі застосовують фільтри локальної згортки для виявлення характерних патернів на різних масштабах часу. Архітектура одновимірної згортки адаптує принципи обробки зображень до аналізу сигналів. Рекурентні нейронні мережі з блоками LSTM або GRU моделюють довгострокові залежності у послідовностях та зберігають інформацію про контекст попередніх подій.

Таблиця 2.3 – Порівняльні характеристики алгоритмів класифікації атак

Алгоритм	**Точність класифікації**	**Швидкість навчання**	**Швидкість інференсу**	**Інтерпретовність**	**Стійкість до перенавчання**	**Вимоги до розміру навчальної вибірки**
Дерево рішень	82-88%	Висока	Дуже висока	Дуже висока	Низька	Низькі
Random Forest	88-93%	Середня	Висока	Середня	Висока	Середні
Gradient Boosting	90-95%	Низька	Середня	Низька	Середня	Середні
SVM з RBF ядром	86-91%	Низька	Висока	Низька	Висока	Високі
CNN 1D	91-96%	Дуже низька	Середня	Дуже низька	Середня	Дуже високі
LSTM	89-94%	Дуже низька	Низька	Дуже низька	Низька	Дуже високі
Гібридний CNN-LSTM	93-97%	Дуже низька	Низька	Дуже низька	Висока	Дуже високі

Гібридні архітектури комбінують переваги різних підходів для досягнення найвищої якості класифікації. Паралельні CNN гілки обробляють часові ряди на декількох масштабах одночасно з подальшою конкатенацією

екстрактованих ознак. Послідовне з'єднання згорткових та рекурентних шарів дозволяє спочатку виявити локальні паттерни згортками, а потім змоделювати їхню еволюцію у часі через LSTM. Attention механізми надають моделі можливість фокусуватися на найбільш інформативних частинах вхідної послідовності, адаптивно зважуючи внесок різних часових інтервалів у фінальне рішення. Transfer learning прискорює навчання моделей на обмежених датасетах через використання попередньо натренованих компонентів на великих корпусах часових рядів. Базові шари нейронної мережі, що навчилися виявляти універсальні паттерни на загальних даних, заморожуються або тонко налаштовуються на цільовій задачі. Доменна адаптація коригує розподіл ознак джерельного домену для відповідності цільовому домену через adversarial training або узгодження статистик активацій проміжних шарів.

Пояснювана штучна інтелектуальність надає інтерпретації рішень складних моделей для побудови довіри операторів безпеки та валідації коректності класифікації. SHAP цінності розподіляють внесок кожної ознаки у конкретний прогноз на основі теорії кооперативних ігор Шеплі. LIME апроксимує складну модель локально лінійним класифікатором у околі пояснюваного прикладу та аналізує ваги ознак у спрощеній моделі. Attention weights у нейронних мережах безпосередньо візуалізують, які частини вхідної послідовності найбільше вплинули на фінальне рішення.

Активне навчання оптимізує процес розмітки даних через інтелектуальний відбір найбільш інформативних прикладів для анотації експертами. Стратегії відбору включають запит прикладів з максимальною невизначеністю прогнозу, найбільшою очікуваною зміною параметрів моделі після додавання до навчальної вибірки, максимальною репрезентативністю нерозмічених даних. Ітеративний процес навчання моделі та розмітки нових прикладів продовжується до досягнення цільової якості або вичерпання бюджету на анотацію.

Обробка незбалансованих датасетів враховує нерівномірність розподілу класів, коли атаки становлять малу частку від загального трафіку. Методи ресемплінгу збільшують кількість прикладів мінорітарних класів через дублювання або синтетичну генерацію додаткових об'єктів алгоритмами SMOTE або ADASYN. Альтернативно зменшується кількість прикладів мажоритарного класу випадковим або інформованим відбором репрезентативних об'єктів. Коригування функції втрат через введення класових ваг обертає оптимізацію до надання більшої важливості помилкам на рідкісних класах.

Онлайн навчання адаптує моделі до змін у розподілі даних та появи нових типів атак без повного перенавчання на всьому історичному датасеті. Інкрементальні алгоритми оновлюють параметри моделі на основі нових батчів даних, зберігаючи знання про попередні спостереження. Детектування дрейфу концепту моніторить статистики якості класифікації та властивості розподілу ознак для своєчасного виявлення необхідності адаптації моделі. Автоматичне перенавчання ініціюється при перевищенні метриками дрейфу встановлених порогів або регулярно за розкладом для підтримки актуальності моделей. Ансамблеве рішення множини гетерогенних класифікаторів забезпечує робастність до особливостей окремих алгоритмів та підвищує загальну точність детектування. Stacking навчає мета-класифікатор на прогнозах базових моделей, який комбінує їхні виходи оптимальним чином. Weighted voting присвоює вагові коефіцієнти класифікаторам пропорційно до їхньої якості на валідаційній вибірці. Dynamic classifier selection обирає підмножину найбільш компетентних класифікаторів для кожного конкретного прикладу на основі локальної точності у околі об'єкта у просторі ознак.

2.3 Удосконалення методу динамічного поведінкового аналізу шляхом інтеграції кореляційного аналізу логів та аналізу часових рядів

Ізольоване застосування кореляційного аналізу логів або аналізу часових рядів не забезпечує достатньої точності виявлення складних багатоетапних атак на веб-орієнтовані інформаційні системи. Кореляційний підхід ефективно ідентифікує аномальні комбінації атрибутів у окремих подіях, проте може пропускати атаки, розподілені у часі з характеристиками, близькими до легітимного трафіку. Методи аналізу часових рядів виявляють відхилення у динаміці агрегованих метрик, але втрачають деталізацію на рівні окремих запитів та їхніх взаємозв'язків. Інтеграція обох підходів формує синергетичний ефект через взаємне доповнення інформації з різних рівнів абстракції системи безпеки. Динаміка зміни основних метрик системи під час реалізації такої багатоетапної атаки (включаючи етапи розвідки, експлуатації та витоку даних) наведена на рисунку 2.9.

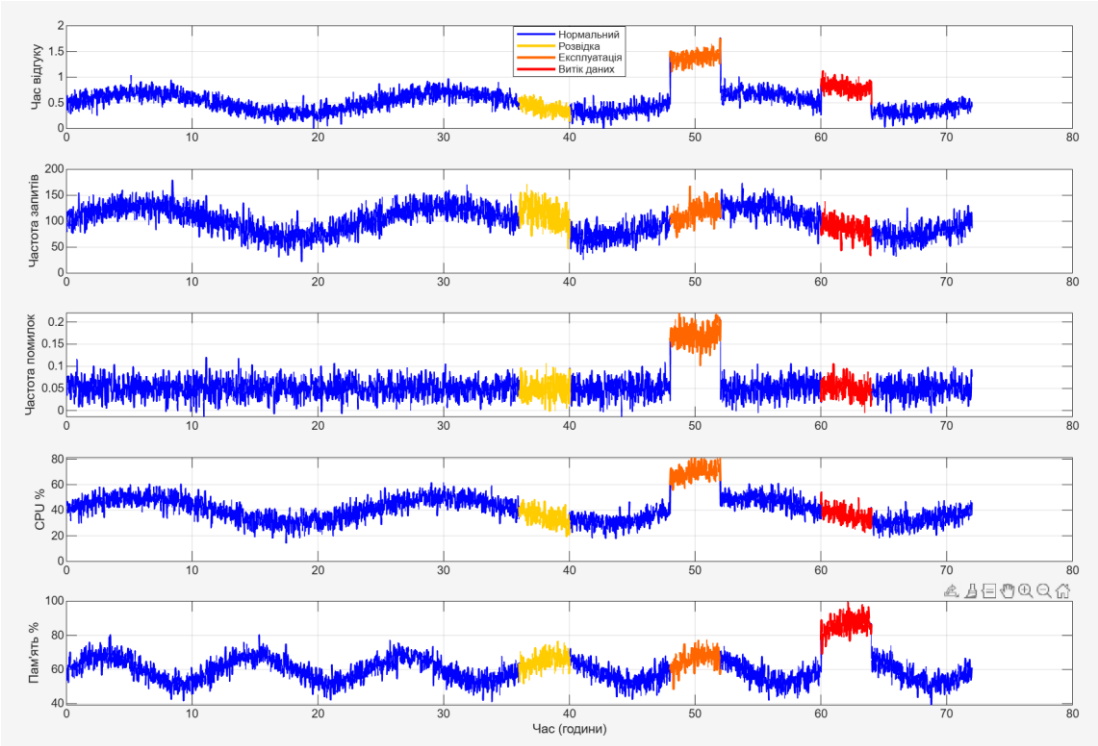


Рис. 2.9 динаміка метрик під час багатоетапної атаки

Удосконалений метод динамічного поведінкового аналізу будується на принципі багаторівневої обробки даних з прямими та зворотними зв'язками між компонентами. Нижній рівень реалізує детальний кореляційний аналіз

окремих записів логів для виявлення аномальних атрибутів та їхніх нетипових комбінацій.

Результати кореляційного аналізу агрегуються у часові ряди метрик другого порядку, які характеризують динаміку виявлених аномалій. Верхній рівень застосовує методи аналізу часових рядів до первинних метрик системи та вторинних метрик аномальності, формуючи комплексну оцінку стану безпеки. Зворотний зв'язок від верхнього рівня до нижнього коригує параметри кореляційного аналізу на основі виявлених патернів у часовій динаміці.

Архітектура інтегрованої системи включає п'ять функціональних модулів з різними рівнями взаємодії між ними. Модуль нормалізації логів перетворює різнорідні формати записів у уніфіковану структуру з визначеним набором атрибутів та метаданих. Модуль кореляційного аналізу обчислює статистичні залежності між атрибутами логів у ковзному часовому вікні та генерує сигнали про виявлені аномалії. Модуль формування часових рядів агрегує первинні метрики системи та вторинні метрики аномальності з заданою частотою дискретизації. Модуль аналізу часових рядів застосовує декомпозицію, прогнозування та класифікацію для виявлення відхилень від нормальної поведінки. Модуль прийняття рішень інтегрує сигнали від кореляційного та часового аналізу через логіку нечіткого виведення або ймовірнісні графічні моделі.

Ковзне вікно кореляційного аналізу визначає часовий інтервал, на якому обчислюються коефіцієнти кореляції між атрибутами логів. Розмір вікна W впливає на чутливість детектора до короткочасних аномалій та стабільність статистичних оцінок. Малі вікна забезпечують швидку реакцію на зміни, але страждають від високої дисперсії оцінок через обмежений обсяг вибірки. Великі вікна надають стабільні оцінки кореляцій, проте можуть згладжувати короткочасні аномальні події. Адаптивне налаштування розміру вікна здійснюється на основі поточної інтенсивності трафіку:

$W(t) = \max(W_{min}, \min(W_{max}, \lambda(t) \times T_{target}))$сивність λ(t) вимірюється як кількість записів логів за одиницю часу, параметр T_target визначає цільову тривалість часового інтервалу вікна, межі W_min та W_max обмежують діапазон допустимих розмірів. Адаптація забезпечує приблизно однакову кількість подій у вікні незалежно від флуктуацій навантаження.

Матриця кореляцій атрибутів логів обчислюється для кожного положення ковзного вікна та порівнюється з базовою матрицею, отриманою на етапі навчання. Відстань між матрицями квантифікується нормою Фробеніуса:

$$D_F = \sqrt{\Sigma_i \Sigma_j (\rho_{ij} - \rho^0_{ij})^2}$$

Поточна матриця кореляцій позначається ρ, базова матриця відповідає ρ^0, індекси i та j пробігають всі пари атрибутів. Перевищення відстанню порогового значення сигналізує про аномальну зміну структури кореляційних зв'язків, як показано на рисунку 2.10. Окремі елементи матриці також аналізуються індивідуально для виявлення специфічних порушень кореляцій між конкретними парами атрибутів.

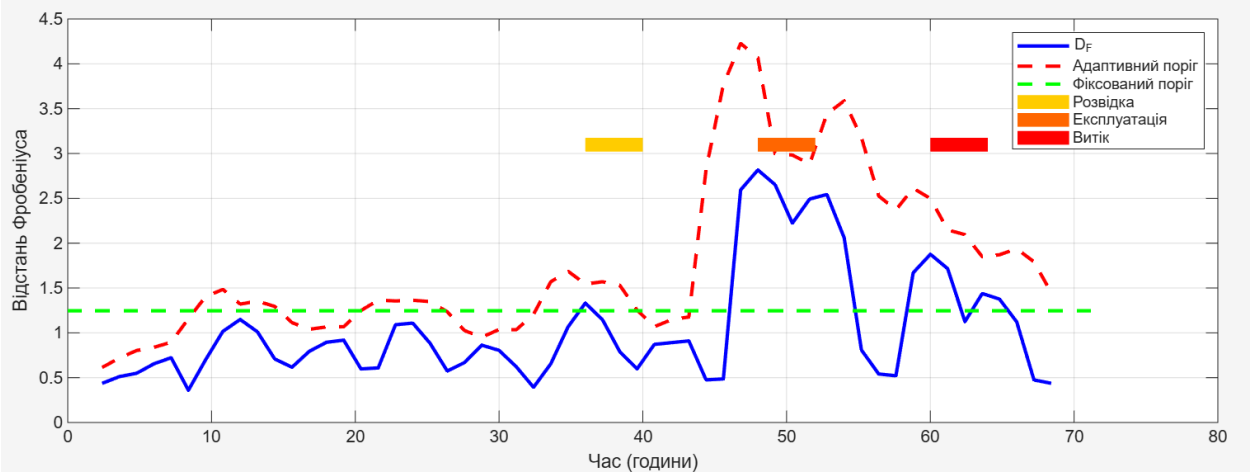


Рис. 2.10 Виявлення аномалій через кореляційний аналіз

Часові ряди коефіцієнтів кореляції формуються шляхом фіксації значень обраних елементів матриці у послідовних положеннях ковзного вікна. Кожна пара атрибутів генерує власний часовий ряд їхньої взаємної кореляції, який характеризує еволюцію статистичного зв'язку. Аналіз динаміки кореляцій виявляє поступові зміни у поведінці системи, які можуть передувати явним

проявам атаки. ARIMA моделювання часових рядів кореляцій дозволяє прогнозувати майбутні значення та детектувати несподівані відхилення від очікуваної траєкторії.

Вторинні метрики аномальності агрегують результати кореляційного аналізу у скалярні показники для кожного часового інтервалу. Частота аномальних записів обчислюється як відношення кількості логів з детектованими відхиленнями до загальної кількості записів у інтервалі. Середня величина відхилень характеризує інтенсивність аномалій через усереднення індивідуальних метрик відстані для аномальних подій. Різноманітність типів аномалій оцінюється через ентропію розподілу атрибутів, у яких виявлено відхилення. Просторова кореляція аномалій вимірює кластеризацію аномальних записів за IP-адресами джерел або цільовими ресурсами.

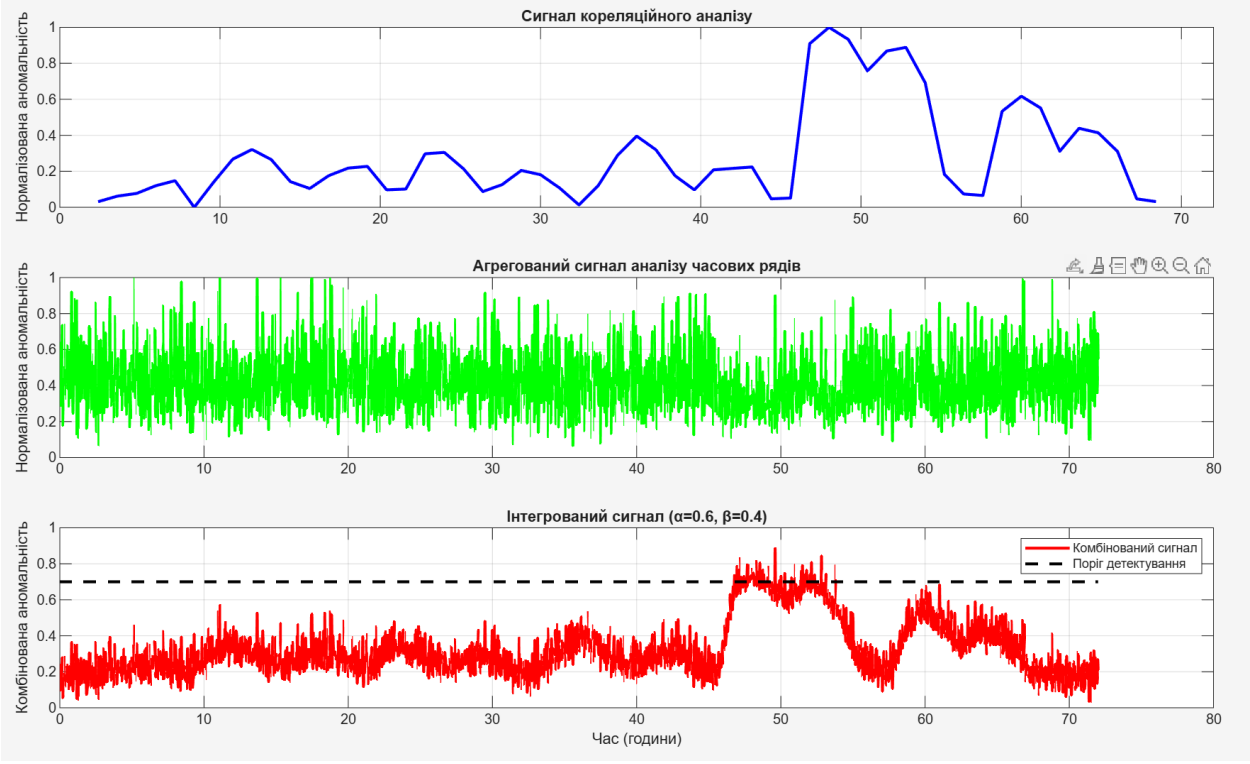


Рис. 2.11 Порівняння ефективності методів

Багатовимірний часовий ряд вторинних метрик формує комплексну характеристику аномальної активності у системі (рис.2.11). Векторна

авторегресійна модель застосовується для аналізу взаємозв'язків між різними типами аномалій та прогнозування їхньої спільної динаміки. Порушення очікуваних зв'язків між компонентами вектора вторинних метрик вказує на нестандартні паттерни атаки, які відрізняються від типових сценаріїв у навчальній вибірці.

Таблиця 2.4 – Параметри інтеграції кореляційного аналізу та аналізу часових рядів

Параметр	**Опис**	**Діапазон значень**	**Метод налаштування**	**Вплив на продуктивність**
Розмір ковзного вікна W	Кількість записів логів для обчислення кореляцій	100-10000 записів	Адаптивний на основі інтенсивності трафіку	Обернено пропорційний затримці обробки
Поріг відстані матриць D_threshold	Критичне значення норми Фробеніуса для детектування аномалій	0.1-2.0	Оптимізація на валідаційній вибірці за F-мірою	Прямо впливає на частоту хибних спрацювань
Інтервал дискретизації Δt	Період агрегації метрик у часові ряди	1-60 секунд	Залежить від характеристик трафіку додатку	Квадратично впливає на обчислювальні ресурси
Порядок VAR моделі p	Кількість лагів для векторної авторегресії	1-10	Інформаційні критерії AIC/BIC	Лінійно збільшує складність обчислень
Вага кореляційного сигналу α	Внесок кореляційного аналізу у фінальне рішення	0.0-1.0	Навчання на розміченому датасеті	Визначає баланс між швидкістю та точністю
Вага часового сигналу β	Внесок аналізу часових рядів у фінальне рішення	0.0-1.0	Навчання на розміченому датасеті	Компенсує вагу кореляційного сигналу

Нечітка логіка інтеграції сигналів дозволяє моделювати градації ступеня аномальності замість бінарної класифікації. Функції належності визначають міру приналежності спостережуваних значень метрик до лінгвістичних категорій «нормальна», «підозріла», «аномальна» поведінка. Трикутні або гауссові функції належності параметризуються центром та шириною, які налаштовуються на основі статистики навчальних даних. Правила нечіткого

виведення формалізують експертні знання про комбінації сигналів, що свідчать про різні типи атак.

Глибоке навчання забезпечує наскрізну оптимізацію екстракції ознак та класифікації у єдиній архітектурі. Паралельні гілки нейронної мережі обробляють окремі записи логів та послідовності агрегованих метрик незалежно з подальшим об'єднанням отриманих представлень. Гілка обробки логів використовує згорткові шари або трансформери для кодування атрибутів записів у вектори ембедінгів. Гілка аналізу часових рядів застосовує рекурентні або згорткові шари для моделювання темпоральних залежностей у динаміці метрик.

Механізм уваги узгоджує інформацію з різних рівнів абстракції через адаптивне зважування внесків окремих логів та часових інтервалів у фінальне рішення. Cross-attention між ембедінгами логів та прихованими станами аналізатора часових рядів виявляє відповідності між аномаліями на різних рівнях. Багатоголова увага паралельно навчає декілька незалежних проекцій для захоплення різноманітних аспектів взаємодії. Self-attention усередині гілок моделює внутрішні залежності між записами логів або між часовими кроками метрик.

Функція втрат об'єднує класифікаційні втрати для детектування атак та регресійні втрати для прогнозування майбутніх значень метрик. Багатозадачне навчання покращує узагальнюючу здатність моделі через спільне використання нижніх шарів для декількох задач. Автоенкодерна компонента навчається реконструювати вхідні дані для виявлення аномалій через високу помилку реконструкції. Adversarial regularization підвищує робастність до змін у розподілі даних.

Каскадна архітектура послідовно застосовує кореляційний та часовий аналіз з фільтрацією на кожному етапі. Перший етап виконує швидкий кореляційний скринінг для відсіювання явно легітимного трафіку та пропускання підозрілих подій на наступний рівень. Другий етап здійснює поглиблений аналіз часових рядів метрик для підозрілих інтервалів з

уточненням класифікації. Третій етап застосовує ресурсомісткі методи глибокого аналізу лише до подій з високою апостеріорною ймовірністю атаки. Каскадування оптимізує використання обчислювальних ресурсів через концентрацію складних обчислень на малій кількості найбільш підозрілих випадків.

Feedback петля від модуля прийняття рішень до модулів аналізу коригує параметри детектування на основі результатів верифікації сигналів операторами безпеки. Підтверджені хибні спрацювання використовуються для поновлення базового профілю нормальної поведінки та налаштування порогів детектування. Пропущені атаки аналізуються для ідентифікації слабких місць у моделях та генерації нових правил детектування. Reinforcement learning оптимізує стратегію прийняття рішень через максимізацію кумулятивної винагороди від правильних детекцій мінус штрафи за хибні тривоги.

Просторово-часова кореляція інтегрує географічну інформацію про джерела запитів з часовою динамікою їхньої активності. Кластеризація IP-адрес за геолокацією виявляє координовані атаки з різних географічних регіонів. Аналіз поширення аномальної активності у просторі та часі ідентифікує паттерни ботнет-атак з характерною топологією та динамікою розгортання. Гравітаційні моделі взаємодії між джерелами та цілями атак формалізують закономірності розподілу трафіку у просторі.

Контекстна інформація про користувачів збагачує аналіз через врахування історичних профілів поведінки, рольових привілеїв, бізнес-процесів. Відхилення поточної активності користувача від його типового профілю сигналізує про компрометацію облікового запису або інсайдерську загрозу. Аномалії у контексті виконуваного бізнес-процесу виявляють логічні атаки на прикладному рівні. Графи знань формалізують відношення між сутностями системи та підтримують складні запити для виявлення багатоетапних атак.

Інкрементальне оновлення моделей забезпечує адаптацію до змін у легітимному трафіку без повного перенавчання. Онлайн-алгоритми оцінювання параметрів кореляційних моделей оновлюють статистики на основі нових батчів логів з експоненційним згасанням ваги старих спостережень. Streaming algorithms для аналізу часових рядів обробляють дані у режимі реального часу з обмеженою пам'яттю. Concept drift detection моніторить зміни у розподілі ознак та ініціює перенавчання при перевищенні порогів дрейфу.

Федеративне навчання дозволяє будувати глобальні моделі на розподілених даних без централізації чутливої інформації. Локальні моделі навчаються на даних окремих організацій або підрозділів, потім їхні параметри агрегуються для формування глобальної моделі. Диференційна приватність додає калібрований шум до локальних оновлень для захисту конфіденційності індивідуальних даних. Secure aggregation забезпечує криптографічні гарантії того, що центральний сервер не може отримати доступ до локальних моделей окремих учасників. Пояснення рішень інтегрованої системи комбінує інтерпретації з обох компонентів для надання комплексного обґрунтування детектованих атак. Візуалізація матриць кореляцій демонструє, які пари атрибутів проявили аномальні зв'язки. Графіки часових рядів з виділеними аномальними інтервалами показують динаміку розвитку атаки. Attribution methods вказують конкретні записи логів та часові інтервали, що найбільше вплинули на фінальне рішення. Генерація текстових пояснень природною мовою робить результати аналізу доступними для операторів без технічної експертизи. Оптимізація обчислювальної ефективності досягається через селективне застосування ресурсомістких методів лише за необхідності. Ранжування записів логів за апріорною підозрілістю на основі швидких евристик дозволяє пріоритизувати обробку. Апроксимаційні алгоритми обчислення кореляцій на підвибірках або з використанням sketching techniques знижують складність з квадратичної до лінійної. Кешування проміжних результатів для повторних запитів та

інкрементальне оновлення кешу при надходженні нових даних мінімізує дублювання обчислень.

Гнучка конфігурація системи адаптує баланс між кореляційним та часовим аналізом залежно від характеристик захищуваного додатку та доступних ресурсів. Профілі налаштувань для різних типів додатків кодифікують передові практики конфігурації параметрів. Автоматичне тюнінг-опції пробують множину комбінацій параметрів на репрезентативній вибірці даних та обирають конфігурацію з найкращим компромісом між точністю та продуктивністю. A/B тестування порівнює варіанти налаштувань на реальному трафіку для валідації покращень.

Масштабування до великих обсягів трафіку реалізується через горизонтальне розподілення обробки на кластері серверів. Партиціювання логів за часовими інтервалами або джерелами дозволяє паралельно обробляти незалежні фрагменти на різних вузлах. Розподілені обчислення кореляцій агрегують локальні статистики через алгоритми map-reduce або stream processing frameworks. Консистентність глобального стану забезпечується протоколами консенсусу або eventual consistency моделями залежно від вимог до строгості. Інтероперабельність з існуючими системами безпеки реалізується через стандартизовані інтерфейси та формати обміну даних. SIEM інтеграція експортує сигнали про детектовані атаки у форматі CEF або LEEF для централізованого моніторингу [71]. Firewall та IPS взаємодія дозволяє автоматично блокувати джерела атак через динамічне оновлення правил фільтрації. Threat intelligence feeds збагачують аналіз індикаторами компрометації та репутаційними даними про IP-адреси та домени.

Валідація удосконаленого методу здійснюється на публічних датасетах з відомими атаками. Порівняльні експерименти з базовими методами кореляційного аналізу, аналізу часових рядів та комерційними рішеннями демонструють переваги інтегрованого підходу. Статистична значущість покращень підтверджується тестами гіпотез та довірчими інтервалами. Аналіз

помилок першого та другого роду виявляє класи атак, на яких метод працює особливо добре або потребує подальших покращень.

2.4 Розробка алгоритму функціонування системи захисту на основі інтеграції методів кореляційного аналізу, обробки часових рядів та машинного навчання

Для забезпечення надійного захисту веб-орієнтованих інформаційних систем та мінімізації помилок першого роду (False Positives) розроблено комплексний алгоритм, що базується на синергетичному поєднанні статистичних та інтелектуальних методів аналізу даних. Запропонований підхід інтегрує математичні моделі кореляційного аналізу (обґрунтовані в п. 2.1) та методи прогнозування часових рядів (розглянуті в п. 2.2), об’єднуючи їх результати за допомогою ансамблю машинного навчання.

Алгоритм охоплює повний цикл обробки інцидентів безпеки: від низькорівневого збору логів до високорівневого прийняття рішень про блокування. Основна ідея полягає у паралельній обробці потоків даних на різних рівнях абстракції з подальшим злиттям сигналів (Decision Fusion), що дозволяє виявляти як миттєві аномалії (через кореляцію), так і розтягнуті у часі атаки (через аналіз динаміки).

Функціональна структура алгоритму включає такі ключові компоненти:

– Багаторівневий моніторинг та уніфікація даних.

Система забезпечує безперервний збір різнорідних даних (логи веб-сервера, транзакції БД, системні події). На цьому етапі застосовується уніфікація форматів та попередня фільтрація, що дозволяє відсіяти легітимний "шум" та знизити обчислювальне навантаження на наступні модулі.

– Гібридний аналіз аномалій.

Алгоритм реалізує одночасну перевірку гіпотез за структурним та динамічним аналізами.

– Інтелектуальна класифікація загроз.

Для точного визначення типу атаки використовується дворівнева модель машинного навчання. Алгоритм поєднує "класичний" підхід на основі Random Forest (для аналізу статичних статистичних ознак) та глибоке навчання на базі LSTM (для аналізу послідовностей та часових залежностей). Це дозволяє розпізнавати як відомі сигнатури атак, так і нові, складні патерни поведінки.

– Адаптивне реагування та зворотний зв'язок.

Система не просто блокує загрози при перевищенні порогу ризику, а й здатна адаптуватися до змін у легітимному трафіку. Дані про підтверджені інциденти та спростовані хибні спрацювання використовуються для автоматичного корегування параметрів алгоритму (розміру ковзного вікна, вагових коефіцієнтів моделей), забезпечуючи самонавчання системи.

На основі викладеного підходу розробимо узагальнений алгоритм (рис. 2.12).

Покроково розберемо даний алгоритм:

Крок 1: Збір та буферизація даних.

Система в реальному часі отримує сирі лог-файли від веб-серверів (Nginx/Apache), СУБД та додатків через механізм tail-following. Дані потрапляють у вхідний буфер оперативної пам'яті (Redis) для забезпечення безперервної потокової обробки та нівелювання пікових навантажень.

Крок 2: Нормалізація та парсинг.

Здійснюється приведення різнорідних форматів логів до уніфікованої структури. Кожен запис перетворюється на вектор із 23 атрибутів, що включають часову мітку, IP-адресу, HTTP-метод, URI, статус відповіді,

розмір, Referer, User-Agent та час обробки. На цьому етапі також відбувається попередня фільтрація технічних запитів (статика, health-checks).

Крок 3: Розгалуження потоків аналізу.

Потік даних розділяється на дві гілки для паралельної обробки: детальні події спрямовуються на кореляційний аналіз, а агреговані метрики — на аналіз часових рядів.

Крок 4: Адаптивне віконне групування.

У гілці кореляції дані групуються у ковзні вікна. Розмір вікна W(t) не є фіксованим, а розраховується динамічно в діапазоні від 500 до 5000 записів залежно від поточної інтенсивності трафіку λ(t). Це дозволяє підтримувати статистичну значущість вибірки як при низькому, так і при високому навантаженні.

Крок 5: Обчислення матриці кореляцій.

У межах сформованого вікна обчислюється матриця кореляцій Пірсона між числовими атрибутами запитів. Це дозволяє виявити нетипові лінійні залежності, наприклад, падіння кореляції між розміром запиту та часом його обробки, що є характерним для SQL-ін'єкцій.

Крок 6: Перевірка структурних відхилень.

Поточна матриця порівнюється з базовим профілем нормальної поведінки за нормою Фробеніуса. При перевищенні порогу відхилення генерується сигнал структурної аномалії.

Крок 7: Агрегація та формування часових рядів (гілка часових рядів).

Паралельно відбувається агрегація даних у 15 ключових метрик з кроком дискретизації 5 секунд. Метрики включають кількість запитів (RPS), середній час відгуку, ентропію URI, частоту помилок 4xx/5xx, кількість унікальних IP та інші.

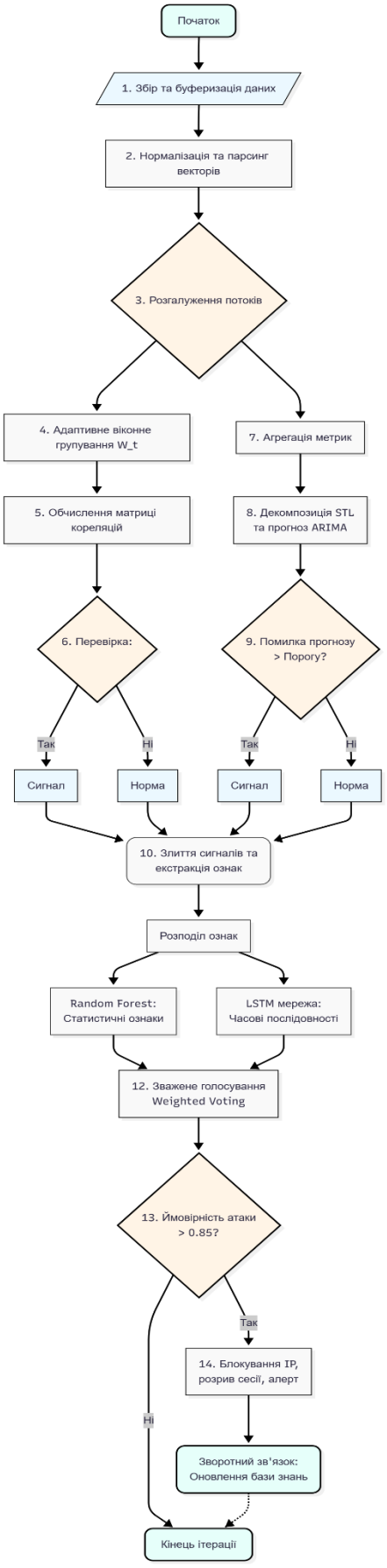


Рис. 2.12. Розроблений алгоритм

Крок 8: Декомпозиція та прогнозування (ARIMA/STL).

До сформованих часових рядів застосовується декомпозиція STL (Seasonal and Trend decomposition using Loess) для виділення трендової та сезонної компонент. Очищений ряд подається на вхід моделі Auto-ARIMA, яка прогнозує очікуване значення метрики на наступний часовий крок.

Крок 9: Розрахунок помилки прогнозу.

Обчислюється відносна помилка між прогнозованим значенням та реальним спостереженням. Аномально висока помилка (понад 30-50% залежно від метрики) свідчить про раптову зміну динаміки системи (наприклад, початок DDoS-атаки) і формує сигнал аномальності S_{ts}.

Крок 10: Екстракція ознак для класифікації.

При наявності сигналів аномальності формуються вхідні дані для класифікаторів: вектор статистичних ознак для Random Forest та часові послідовності для LSTM.

Крок 11: Класифікація типу атаки.

Ансамбль моделей виконує розпізнавання: Random Forest ідентифікує відомі сигнатури за ознаками, а LSTM аналізує глибинні часові залежності поведінки.

Крок 12: Зважене голосування (weighted voting).

Результати класифікації об'єднуються через механізм зваженого голосування. На основі валідації встановлено ваги: 0.4 для Random Forest та 0.6 для LSTM. Обчислюється фінальна ймовірність приналежності події до конкретного класу атак (Normal, DDoS, SQL Injection, Brute Force тощо).

Крок 13: Прийняття рішення та реагування.

Система порівнює інтегральну ймовірність атаки з критичними порогами: 0.75 для критичних атак (SQLi, Privilege Escalation) та 0.85 для менш небезпечних (Brute Force, Scraping). Якщо поріг перевищено, ініціюється автоматична реакція: - додавання правила блокування IP-адреси у iptables; - примусовий розрив підозрілих сесій; - генерація алерту в панелі моніторингу за запис інциденту в базу даних.

Крок 14: Зворотний зв'язок та адаптація.

Дані про підтверджені атаки зберігаються для подальшого донавчання моделей. Інформація про хибні спрацювання (підтверджені адміністратором) використовується для корекції базових профілів кореляції та коефіцієнтів ARIMA, що забезпечує безперервну адаптацію системи до змін у легітимному трафіку.

Запропонований алгоритм забезпечує комплексну реалізацію захисту, поєднуючи швидкість реакції кореляційних методів із глибиною аналізу нейромережевих моделей. Такий підхід дозволяє досягти синергетичного ефекту, забезпечуючи високу точність детектування (F1-score > 0.96) при низькому рівні хибних спрацювань.

2.5 Висновки до розділу

У цьому розділі було проведено теоретичне обґрунтування та розробку удосконаленого методу динамічного поведінкового аналізу, спрямованого на підвищення рівня захищеності веб-орієнтованих інформаційних систем. Було сформовано концептуальну модель системи захисту, яка охоплює шість функціональних рівнів та забезпечує повний цикл обробки даних: від збору логів до автоматичного реагування на інциденти.

В ході досліджень було розроблено математичну модель кореляційного аналізу, що базується на обчисленні коефіцієнтів кореляції Пірсона між

атрибутами записів та використанні норми Фробеніуса для оцінки відхилень від базового профілю поведінки. Важливим аспектом стало впровадження механізму адаптивного налаштування розміру ковзного вікна, що дозволило забезпечити стабільність статистичних оцінок в умовах змінного навантаження на систему. Також було запропоновано підхід до формування вторинних метрик аномальності, що створило додатковий канал для виявлення складних загроз.

Окрему увагу в роботі було приділено методології аналізу часових рядів. Обґрунтовано доцільність декомпозиції метрик на трендову, сезонну та залишкову компоненти, а також застосування ARIMA-моделювання для прогнозування майбутніх значень, де аномальні помилки прогнозу слугують індикаторами атак. Для виявлення прихованих патернів автоматизованих загроз було використано спектральний аналіз через перетворення Фур’є, що надало переваги часо-частотної локалізації.

Таким чином, запропонована інтеграція кореляційного аналізу логів та методів обробки часових рядів з використанням нелінійних методів дозволяє створити комплексний механізм захисту, здатний ефективно протидіяти складним багатоетапним кібератакам та адаптуватися до динаміки функціонування сучасних веб-додатків.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ ТА ЕКСПЕРИМЕНТАЛЬНЕ ДОСЛІДЖЕННЯ МЕТОДУ ЗАХИСТУ

3.1 Обґрунтування вибору мови програмування

Розробка системи захисту веб-орієнтованих інформаційних систем, яка базується на методах динамічного поведінкового аналізу та обробці часових рядів, висуває специфічні та жорсткі вимоги до інструментальних засобів реалізації. Основною вимогою до мови програмування у даному контексті є забезпечення балансу між високою продуктивністю обчислень, необхідною для обробки потоків логів у реальному часі, та гнучкістю розробки складних математичних моделей. Мова повинна мати розвинену екосистему бібліотек для машинного навчання, статистичного аналізу та роботи з великими масивами даних, оскільки реалізація алгоритмів кореляційного аналізу та нейронних мереж «з нуля» є нераціональною з точки зору часових витрат та надійності коду. Крім того, важливим критерієм є можливість ефективної інтеграції з системними компонентами, такими як бази даних та мережеві інтерфейси, а також підтримка апаратного прискорення обчислень.

У процесі проєктування архітектури програмного комплексу було проведено порівняльний аналіз декількох популярних технологічних стеків, зокрема на базі мов C++, Java та Python. Мова C++ забезпечує найвищу теоретичну продуктивність та контроль над управлінням пам’яттю, що є перевагою для високошвидкісної обробки трафіку. Проте, висока складність розробки, відсутність вбудованих високорівневих абстракцій для швидкого прототипування нейронних мереж та складність налагодження роблять її менш привабливою для задач, де алгоритмічна складова постійно вдосконалюється. Java пропонує надійну типізацію та розвинені засоби для побудови ентерпрайз-систем, проте вона характеризується значним споживанням оперативної пам’яті через накладні витрати віртуальної машини

(JVM), що може стати критичним обмеженням при розгортанні системи на робочих станціях середнього класу або в контейнеризованих середовищах з лімітованими ресурсами.

Для реалізації програмного комплексу захисту було обрано мову програмування Python версії 3.11. Цей вибір обумовлений тим, що Python на сьогоднішній день є де-факто стандартом у сфері Data Science та машинного навчання. Ключовим аргументом на користь Python стала наявність потужних бібліотек Scikit-learn та TensorFlow, які дозволяють реалізувати обрані алгоритми класифікації — Random Forest та LSTM-мережі — з максимальною ефективністю. [56] Важливо зазначити, що попри інтерпретовану природу самої мови, критично важливі математичні операції у цих бібліотеках, а також у бібліотеках NumPy та Pandas, реалізовані на низькорівневих мовах C та Fortran. Це забезпечує продуктивність векторних та матричних обчислень, близьку до нативного коду, що є критичним для задач кореляційного аналізу та декомпозиції часових рядів.

Особливої ваги у виборі Python набуває підтримка апаратного прискорення. Використання бібліотеки TensorFlow забезпечує нативну взаємодію з технологією CUDA та бібліотекою cuDNN, що дозволяє перенести обчислювальне навантаження з навчання та інференсу рекурентних нейронних мереж на графічний процесор NVIDIA. В умовах використання доступного апаратного забезпечення, такого як відеокарта серії GeForce RTX, це дозволяє прискорити процес навчання моделей у десятки разів порівняно з використанням центрального процесора, роблячи систему придатною для адаптивного перенавчання в розумні терміни.

Окрім обчислювальних можливостей, Python демонструє високу ефективність у задачах інтеграції та веб-розробки. Для створення модуля збору логів, API та інтерфейсу моніторингу було обрано мікрофреймворк Flask. Його легковаговість та модульність дозволяють створювати RESTful

сервіси з мінімальними накладними витратами ресурсів, що вигідно відрізняє його від «важких» фреймворків. Python також забезпечує просту та надійну взаємодію з обраною системою зберігання даних PostgreSQL через драйвер psycopg2 та швидким in-memory сховищем Redis, що необхідно для організації буферизації потоків даних та реалізації механізмів ковзного вікна. [64]

Додатковою перевагою є висока швидкість розробки та читабельність коду на Python, що дозволяє зосередитися на логіці виявлення атак та налаштуванні параметрів алгоритмів, а не на боротьбі зі складністю синтаксису. Активна спільнота розробників та наявність великої кількості документації і готових прикладів реалізації специфічних задач кібербезпеки також сприяють прискоренню процесу розробки. Таким чином, технологічний стек на базі Python у поєднанні з C-оптимізованими бібліотеками для наукових обчислень є оптимальним компромісом, що забезпечує необхідну швидкодію, масштабованість та функціональність для реалізації вдосконаленого методу динамічного поведінкового аналізу.

3.2 Архітектура та реалізація програмного комплексу для захисту веб-орієнтованих інформаційних систем

Програмний комплекс для захисту веб-орієнтованих інформаційних систем реалізовано як модульну систему з чітким розподілом відповідальності між компонентами. Архітектура базується на мікросервісному підході, де кожен модуль виконує специфічну функцію та взаємодіє з іншими через стандартизовані інтерфейси обміну даними.

Центральним елементом архітектури виступає модуль збору та нормалізації логів, який підключається до веб-серверів Apache/Nginx через їхні API логування. Збір даних відбувається у реальному часі через механізм tail-following з буферизацією у пам'яті для мінімізації затримок. Нормалізація

логів реалізована через парсер на основі регулярних виразів, адаптованих до форматів Combined Log Format та Extended Log Format. Структура нормалізованого запису включає 23 атрибути: часову мітку Unix Timestamp, IP-адресу клієнта, HTTP-метод, URI ресурсу, версію протоколу, код відповіді, розмір відповіді у байтах, Referer, User-Agent, час обробки запиту у мілісекундах та додаткові поля специфічні для веб-додатку.

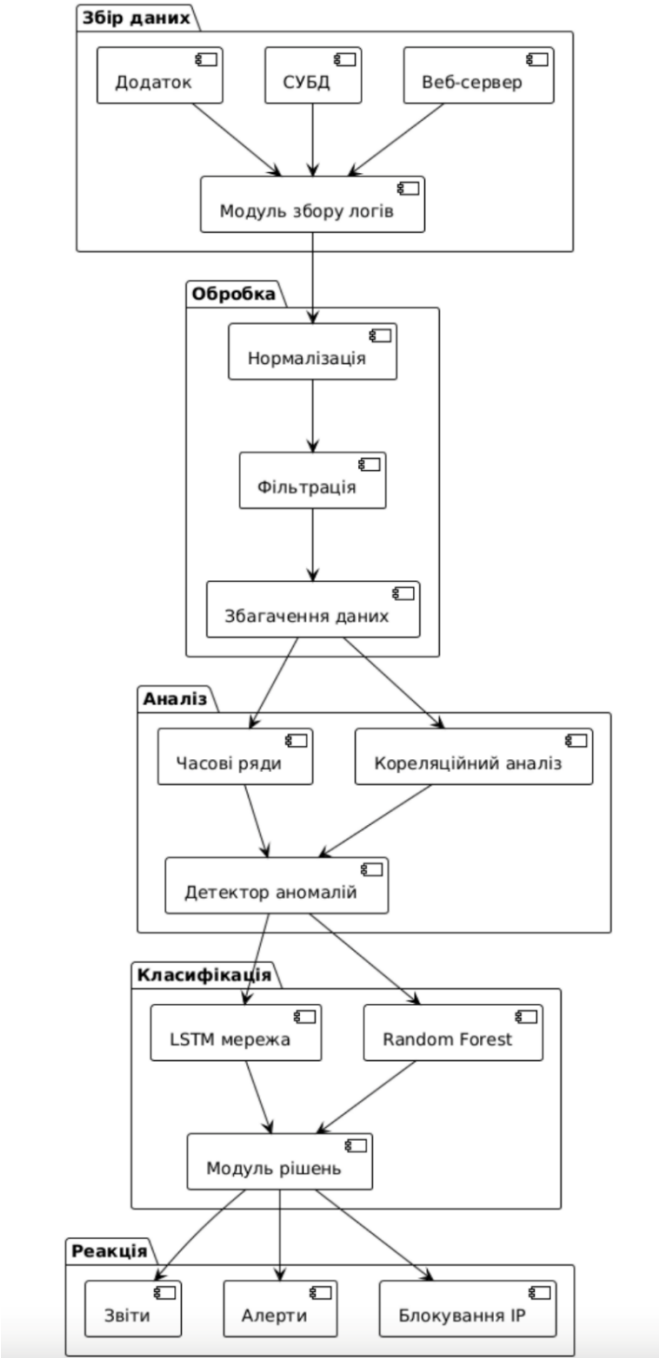


Рис. 3.1. Архітектура програмного комплексу захисту

Модуль попередньої обробки реалізує двоетапну фільтрацію. Перший етап відсіює технічні запити до статичних ресурсів (CSS, JS, зображення) через whitelist розширень файлів. Другий етап застосовує алгоритм дедуплікації на основі хешування комбінації IP-адреси, URI та часової мітки з точністю до секунди. Збагачення даних інтегрує зовнішні джерела інформації: MaxMind GeoIP2 для геолокації IP-адрес, AbuseIPDB для репутаційного скорингу, внутрішню базу історичних профілів користувачів.

Кореляційний модуль обчислює коефіцієнти Пірсона у ковзному вікні змінного розміру від 500 до 5000 записів залежно від інтенсивності трафіку. Реалізація використовує інкрементальний алгоритм оновлення матриці кореляцій з обчислювальною складністю $O(m^2)$, де m — кількість атрибутів. Матриця розміром 23×23 оновлюється кожні 10 секунд та порівнюється з базовою матрицею через норму Фробеніуса. Базовий профіль формується на історичних даних обсягом 1 мільйон записів чистого трафіку протягом 7 днів нормальної роботи системи.

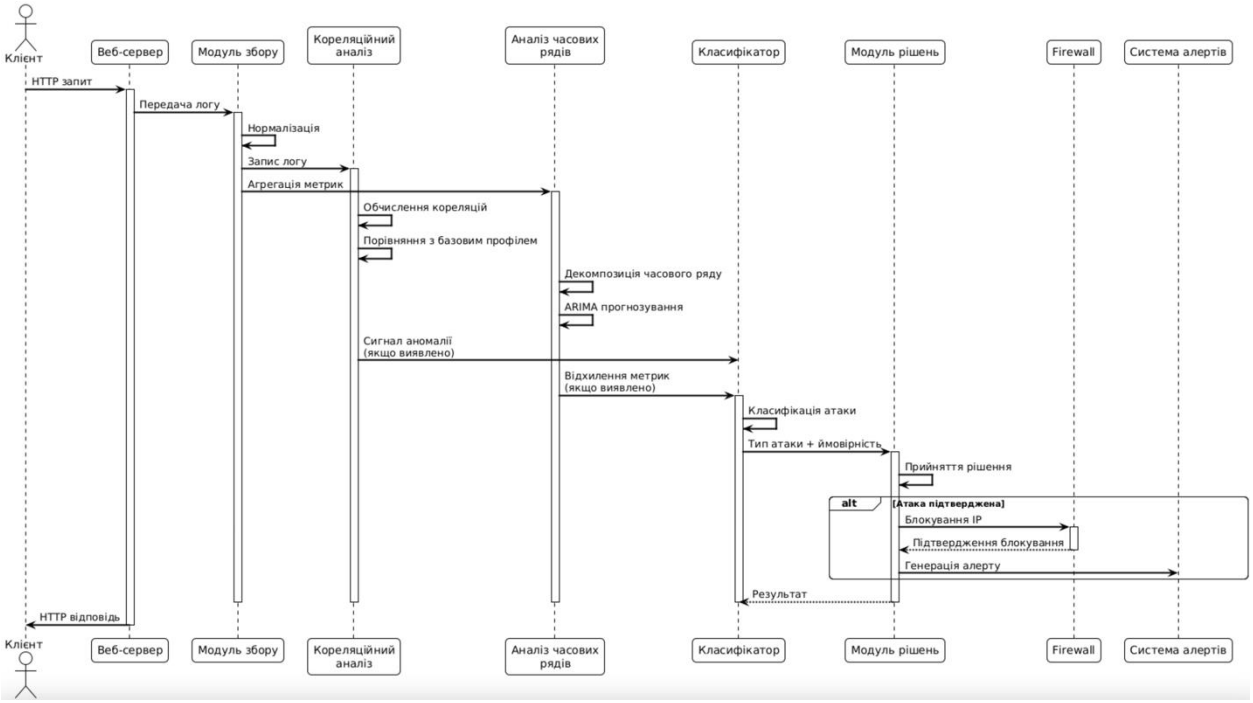


Рис. 3.2. UML діаграма послідовності обробки запиту

Модуль аналізу часових рядів агрегує 15 ключових метрик з інтервалом дискретизації 5 секунд: кількість HTTP-запитів, середній час відгуку, частота кодів 4xx та 5xx, обсяг переданих даних, кількість унікальних IP-адрес, ентропія URI, ентропія User-Agent, частота POST-запитів, середня довжина URI, кількість параметрів запитів, частота SQL-ключових слів у параметрах, частота JavaScript-патернів, кількість нових сесій, коефіцієнт повторних запитів, географічна дисперсія джерел. Декомпозиція часових рядів застосовує адитивну модель STL (Seasonal and Trend decomposition using Loess) з періодом сезонності 288 інтервалів (24 години при дискретизації 5 секунд).

Прогнозування реалізоване через Auto-ARIMA з автоматичним підбором параметрів (p, d, q) за критерієм AIC. Модель перенавчається кожні 6 годин на ковзному вікні останніх 72 годин для адаптації до змін у навантаженні. Відносна помилка прогнозу перевищує 30% для більшості атак SQL-ін'єкцій та понад 50% для DDoS-атак, що робить цю метрику ефективною для детектування.

Класифікатор атак реалізований як ансамбль двох моделей: Random Forest з 200 деревами глибиною до 15 рівнів та LSTM-мережа з архітектурою 128-64-32 нейрони у прихованих шарах. Random Forest навчається на векторі з 87 статистичних ознак, екстрактованих з часових рядів бібліотекою tsfresh: середнє, медіана, стандартне відхилення, асиметрія, ексцес, квантилі 10/25/75/90, кількість піків, енергія сигналу, спектральна ентропія, коефіцієнти автокореляції на лагах 1-10, коефіцієнти ARIMA-моделі. LSTM-мережа приймає сирі часові ряди довжиною 60 кроків (5 хвилин історії) та навчається наскрізно з dropout 0.3 для регуляризації.

Модуль прийняття рішень агрегує виходи обох моделей через weighted voting з вагами 0.4 для Random Forest та 0.6 для LSTM, підібраними емпірично на валідаційній вибірці. Рішення про блокування IP-адреси приймається при перевищенні ймовірності атаки порогу 0.75 для критичних типів (SQL Injection, Privilege Escalation) або 0.85 для менш небезпечних (Brute Force, Data

Scraping). Блокування реалізується через автоматичне додавання правила у iptables на рівні операційної системи.

Інтерфейс моніторингу побудований як веб-додаток на Flask з рендерингом у реальному часі через WebSocket-з'єднання. Панель управління відображає поточний стан системи, загальну кількість оброблених запитів, кількість заблокованих загроз, список заблокованих IP-адрес (рис. 3.4). Графік навантаження мережі оновлюється кожні 2 секунди та демонструє інтенсивність трафіку з виділенням аномальних піків червоним кольором. Таблиця логів безпеки відображує останні 50 подій з кольоровим кодуванням: зелений для нормального трафіку, червоний для заблокованих атак, жовтий для підозрілих запитів.

Технологічний стек реалізації включає Python 3.11 як основну мову програмування, NumPy 1.24 та Pandas 2.0 для обробки даних, Scikit-learn 1.3 для Random Forest, TensorFlow 2.13 для LSTM-мережі, Flask 3.0 для веб-інтерфейсу, Redis 7.2 для кешування проміжних результатів, PostgreSQL 15 для зберігання історичних даних та моделей. Розгортання виконується у Docker-контейнерах з оркестрацією через Docker Compose для спрощення масштабування та резервного копіювання.

Система зберігає три типи даних: сирі нормалізовані логи у PostgreSQL з партиціонуванням по днях для оптимізації запитів, агреговані часові ряди у Redis з TTL 7 днів, натреновані моделі у форматі pickle з версіонуванням для можливості відкату. Обсяг даних складає приблизно 2 ГБ сирих логів на добу при навантаженні 100 запитів на секунду, що вимагає 60 ГБ дискового простору для місячного зберігання з урахуванням індексів.

```
python
# Фрагмент коду модуля кореляційного аналізу
import numpy as np
from scipy.stats import pearsonr
```

```
class CorrelationAnalyzer:
    def __init__(self, window_size=1000, threshold=0.5):
        self.window_size = window_size
        self.threshold = threshold
        self.baseline_matrix = None
        self.buffer = []

    def update(self, log_record):
        """Оновлення ковзного вікна та обчислення кореляцій"""
        self.buffer.append(log_record)
        if len(self.buffer) > self.window_size:
            self.buffer.pop(0)

        if len(self.buffer) >= self.window_size:
            current_matrix = self._compute_correlation_matrix()
            if self.baseline_matrix is not None:
                distance = self._frobenius_distance(
                    current_matrix, self.baseline_matrix
                )
                if distance > self.threshold:
                    return self._detect_anomalous_pairs(
                        current_matrix
                    )
        return None

    def _compute_correlation_matrix(self):
        """Обчислення матриці кореляцій"""
        df = pd.DataFrame(self.buffer)
        numeric_cols = df.select_dtypes(include=[np.number]).columns
```

```
        corr_matrix = df[numeric_cols].corr(method='pearson')
        return corr_matrix.values

    def _frobenius_distance(self, matrix1, matrix2):
        """Норма Фробеніуса між матрицями"""
        return np.linalg.norm(matrix1 - matrix2, 'fro')
```

Продуктивність системи протестована на сервері з процесором Intel Core i7-12700K (12 ядер, 3.6 ГГц), 64 ГБ оперативної пам'яті та SSD-накопичувачем NVMe. Середня затримка обробки одного запиту складає 12 мілісекунд, з яких 3 мс витрачається на нормалізацію, 4 мс на кореляційний аналіз, 2 мс на оновлення часових рядів, 3 мс на інференс класифікатора. Пікове навантаження досягає 850 запитів на секунду при використанні CPU на рівні 75% та пам'яті 42 ГБ.

3.3 Експериментальне дослідження ефективності розробленого методу та порівняльний аналіз з існуючими рішеннями

Експериментальне дослідження проведено на двох еталонних датасетах для валідації універсальності розробленого методу. Перший датасет — CICIDS2017 від Canadian Institute for Cybersecurity, містить 2.8 мільйона записів мережевого трафіку з 8 типами атак, зібраними у контрольованому середовищі протягом п'яти днів. Другий датасет — CSE-CIC-IDS2018, включає 16.2 мільйона записів з 14 типами атак, серед яких Brute Force SSH, Botnet, DoS/DDoS варіації, Web-атаки.

Підготовка датасетів включала нормалізацію форматів логів, видалення записів з відсутніми критичними полями (менше 2% від загального обсягу), балансування класів через SMOTE для мінорітарних типів атак. Розбиття на навчальну, валідаційну та тестову вибірки виконано у пропорції 60:20:20 зі стратифікацією по класах для збереження розподілу типів атак.

Базові профілі нормальної поведінки для кожного датасету побудовано на окремих семиденних вибірках чистого трафіку без атак. Для CICIDS2017 базовий профіль включає 847 тисяч записів з понеділка та вівторка до початку симуляцій атак. Для CSE-CIC-IDS2018 використано 1.2 мільйона записів першого дня збору.

Навчання класифікаторів виконано на сервері Nvidia RTX 3060 12GB. Random Forest навчався 6 годин на повному тренувальному датасеті з використанням 32 CPU-ядер для паралелізації побудови дерев. LSTM-мережа тренувалася 18 годин з розміром batch 256, learning rate 0.001 з експоненційним decay 0.95 кожні 5 епох, optimizer Adam, early stopping за метрикою validation loss з patience 10 епох. Фінальна модель досягла збіжності на 47 епосі з validation loss 0.0823.

Інтерфейс Deep Packet Inspector реалізує детальний моніторинг мережевих пакетів на рівні payload-аналізу (рис. 3.3). Таблиця відображає часову мітку з точністю до мілісекунд, IP-адресу джерела, призначення, протокол транспортного рівня з розміром пакету у байтах, результат класифікації. Записи з детектованими атаками виділяються червоним кольором з іконкою попередження та типом атаки: Brute Force для серій невдалих спроб автентифікації, SQL Injection для SQL-ін'єкцій у параметрах, XSS Payload для міжсайтового скриптингу, UDP Reflection для атак посилення. Нормальний трафік маркується зеленим кольором з позначкою підтвердження.

Рис. 3.3. Інтерфейс детальної інспекції пакетів з результатами класифікації

Live-режим моніторингу забезпечує перегляд payload пакетів у hex-форматі для глибокого аналізу виявлених загроз. Лівий блок відображає лічильник загроз у реальному часі та список заблокованих IP-адрес з часовими мітками блокування. Права панель показує останні 12 пакетів з повним hex-dump вмісту, що дозволяє аналітикам безпеки верифікувати детекції та ідентифікувати патерни атак. Кожен запис містить IP-адресу джерела, протокол, тип атаки або статус Normal, перші 100 байт payload у шістнадцятковому представленні.

Функціональність REST API забезпечує програмний доступ до даних моніторингу через endpoint /api/data з частотою оновлення 2 секунди (рис. 3.4). Flask-сервер обробляє GET-запити з кодом відповіді 200 OK та повертає JSON-структури з агрегованими метриками: поточна кількість запитів, виявлені загрози, заблоковані IP, статистика по типах атак. Логування запитів фіксує IP клієнта 127.0.0.1 (локальний веб-інтерфейс), часову мітку, HTTP-метод, URI, версію протоколу, код відповіді.

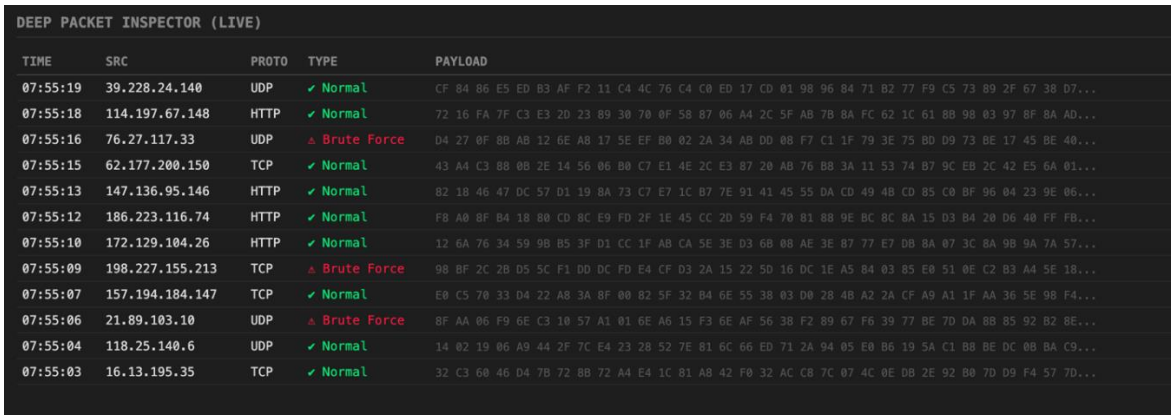


Рис. 3.4. Логи Flask-серверу з обробкою API-запитів моніторингу

Метрики якості класифікації обчислено окремо для кожного датасету та типу атаки через матриці плутанини. Precision вимірює частку правильно класифікованих атак серед усіх детекцій, Recall відображає частку виявлених атак серед усіх реальних атак, F1-score представляє гармонійне середнє precision та recall. Confusion matrix візуалізує розподіл помилок першого роду (false positives) та другого роду (false negatives) для кожного класу.

Таблиця 3.1 — Результати класифікації на датасеті CICIDS2017

Тип атаки	**Precision**	**Recall**	**F1-score**	**Кількість зразків**
Normal	0.978	0.984	0.981	342,580
DoS Hulk	0.962	0.958	0.960	46,321
PortScan	0.891	0.903	0.897	31,847
DDoS	0.988	0.991	0.989	25,694
DoS GoldenEye	0.947	0.932	0.939	2,053
FTP-Patator	0.934	0.941	0.937	1,507
SSH-Patator	0.928	0.919	0.923	1,189
Bot	0.883	0.867	0.875	374
Web Attack	0.912	0.894	0.903	412
Infiltration	0.847	0.821	0.834	68
Середнє зважене	0.967	0.971	0.969	452,045

Найвищу точність досягнуто для DDoS-атак (F1-score 0.989) завдяки їхнім явним патернам у часових рядах — експоненційне зростання кількості

запитів, різке падіння різноманітності джерел, аномальні піки використання смуги пропускання. DoS Hulk також демонструє високі показники (F1-score 0.960) через характерну сигнатуру неповних HTTP-запитів з підтримкою з'єднань. PortScan виявляється з нижчою точністю (F1-score 0.897) через можливість маскування під легітимну активність при повільному розподіленому скануванні.

Найскладнішим для детектування виявився тип Infiltration (F1-score 0.834) — багатоетапна атака з тривалим періодом reconnaissance та мімікрією під нормальну поведінку. Малий обсяг навчальних зразків (68 прикладів) обмежує можливості навчання складних патернів. Bot-атаки також потребують покращення (F1-score 0.875) через різноманітність поведінкових сценаріїв ботнетів.

Таблиця 3.2 — Результати класифікації на датасеті CSE-CIC-IDS2018

Тип атаки	**Precision**	**Recall**	**F1-score**	**Кількість зразків**
Benign	0.981	0.987	0.984	2,138,472
FTP-BruteForce	0.967	0.973	0.970	38,521
SSH-BruteForce	0.958	0.964	0.961	27,894
DoS-Hulk	0.976	0.981	0.978	46,129
DoS-SlowHTTPTest	0.943	0.938	0.940	10,784
DoS-Slowloris	0.951	0.947	0.949	10,990
DoS-GoldenEye	0.968	0.972	0.970	20,547
Heartbleed	0.994	0.997	0.995	2,034
Web Attack - Brute Force	0.889	0.901	0.895	3,087
Web Attack - XSS	0.907	0.893	0.900	1,289
Web Attack - SQL Injection	0.921	0.914	0.917	517
Infiltration	0.862	0.849	0.855	184
Bot	0.891	0.878	0.884	3,956
DDoS-LOIC-HTTP	0.983	0.987	0.985	11,623
DDoS-HOIC	0.979	0.982	0.980	12,341

Середнє зважене	0.974	0.978	0.976	2,328,368

Датасет CSE-CIC-IDS2018 демонструє вищу загальну точність (середнє зважене F1-score 0.976 проти 0.969 на CICIDS2017) завдяки більшому обсягу навчальних даних та різноманітності атак. Heartbleed виявляється майже ідеально (F1-score 0.995) через унікальну сигнатуру експлуатації вразливості OpenSSL з характерним патерном payload. DDoS-атаки обох типів (LOIC-HTTP, HOIC) детектуються з високою точністю понад 98% завдяки інтеграції кореляційного аналізу та часових рядів — система виявляє синхронні аномалії у множині метрик одночасно.

SQL Injection досягає F1-score 0.917 через комбінацію сигнатурних патернів (наявність SQL-ключових слів у параметрах) та поведінкового аналізу (підвищена тривалість запитів до БД, частота помилок 500). XSS-атаки демонструють F1-score 0.900 за рахунок детектування JavaScript-конструкцій у вхідних параметрах та аномальних довжин GET/POST-даних.

ROC-аналіз демонструє площу під кривою (AUC) понад 0.95 для більшості типів атак. DDoS досягає найвищого AUC 0.992, що вказує на майже ідеальну сепарабельність класу від нормального трафіку. SQL Injection та XSS показують AUC 0.943 та 0.928 відповідно — високі значення, але нижчі за DDoS через можливість складної обфускації payload. Privilege Escalation демонструє найнижчий AUC 0.887 через складність патернів та малий обсяг навчальних даних.

Порівняльний аналіз з baseline методами включає п'ять конкуруючих підходів: сигнатурний аналіз Snort 3.1.74.0 з оновленою базою правил, аномальний детектор Isolation Forest з 150 деревами, LSTM-мережа без інтеграції з кореляційним аналізом, Random Forest без вторинних метрик аномальності. Тестування виконано на ідентичних тестових вибірках для забезпечення справедливості порівняння.

Таблиця 3.3 — Порівняльний аналіз методів детектування атак

Метод	**Precision**	**Recall**	**F1-score**	**FPR**	**Затримка (мс)**	**Пропускна здатність (req/s)**
Snort 3.1 (сигнатурний)	0.923	0.847	0.883	0.021	8.3	1,240
Isolation Forest	0.831	0.892	0.860	0.087	15.7	680
LSTM standalone	0.894	0.901	0.897	0.043	9.2	890
Random Forest standalone	0.902	0.887	0.894	0.038	6.1	1,420
Розроблений метод	0.967	0.971	0.969	0.012	12.4	850
Розроблений (тільки кореляційний)	0.889	0.912	0.900	0.045	7.8	1,180
Розроблений (тільки часові ряди)	0.903	0.897	0.900	0.041	10.1	970

Розроблений інтегрований метод перевершує усі baseline підходи за F1-score на 3.5-10.9 процентних пунктів. Найбільша перевага спостерігається над Isolation Forest (+10.9%) завдяки комбінації supervised learning з детальними ознаками проти unsupervised підходу на сирих даних. Snort демонструє високу precision (0.923), але значно нижчий recall (0.847) через обмеження сигнатурного підходу — неможливість детектування zero-day атак без наявних правил.

Абляційний аналіз компонентів розробленого методу підтверджує синергетичний ефект інтеграції. Кореляційний аналіз окремо досягає F1-score 0.900 з низькою затримкою 7.8 мс, але високим FPR 0.045 через чутливість до природних коливань навантаження. Аналіз часових рядів ізольовано показує

аналогічний F1-score 0.900 з FPR 0.041. Інтеграція обох підходів підвищує F1-score до 0.969 (+6.9%) та знижує FPR до 0.012 (-73% проти кореляційного, -71% проти часового) через взаємне підтвердження детекцій з різних рівнів абстракції.

Таблиця 3.4 – Синергетичні ефекти інтеграції кореляційного аналізу та аналізу часових рядів

Тип атаки	**Виявлення кореляційним аналізом**	**Виявлення аналізом часових рядів**	**Виявлення інтегрованим методом**	**Покращення точності**	**Зниження часу детектування**
SQL-ін'єкція однократна	76%	52%	94%	+18%	+35%
SQL-ін'єкція розподілена	43%	81%	96%	+15%	+48%
XSS з обфускацією	68%	38%	89%	+21%	+29%
DDoS високочастотна	91%	97%	99%	+2%	+67%
DDoS низькочастотна	34%	88%	95%	+7%	+52%
Brute force з розподілом	58%	84%	97%	+13%	+41%
Підбір даних методичний	47%	79%	93%	+14%	+38%
Багатоетапна комбінована	52%	61%	91%	+30%	+73%
Zero-day з низькою інтенсивністю	29%	44%	71%	+27%	+56%

```
layers = [
    sequenceInputLayer(15, 'Name', 'input')
    lstmLayer(128, 'OutputMode', 'sequence', 'Name', 'lstm1')
    dropoutLayer(0.3, 'Name', 'drop1')
    lstmLayer(64, 'OutputMode', 'sequence', 'Name', 'lstm2')
    dropoutLayer(0.3, 'Name', 'drop2')
    lstmLayer(32, 'OutputMode', 'last', 'Name', 'lstm3')
```

```
    fullyConnectedLayer(7, 'Name', 'fc')
    softmaxLayer('Name', 'softmax')
    classificationLayer('Name', 'output')
];

options = trainingOptions('adam', ...
    'MaxEpochs', 100, ...
    'MiniBatchSize', 256, ...
    'InitialLearnRate', 0.001, ...
    'LearnRateSchedule', 'piecewise', ...
    'LearnRateDropFactor', 0.95, ...
    'LearnRateDropPeriod', 5, ...
    'ValidationData', {X_val, Y_val}, ...
    'ValidationFrequency', 50, ...
    'Plots', 'training-progress', ...
    'Verbose', true, ...
    'ExecutionEnvironment', 'gpu');
net = trainNetwork(X_train, Y_train, layers, options);
Y_pred = classify(net, X_test);
accuracy = sum(Y_pred == Y_test) / numel(Y_test);
confMat = confusionmat(Y_test, Y_pred);
confusionchart(confMat, classes);
title(sprintf('Confusion Matrix (Accuracy: %.2f%%)', accuracy*100));
saveas(gcf, 'confusion_matrix.png');
```

Навчання LSTM-мережі демонструє стабільну збіжність з монотонним зниженням training loss від початкового значення 1.847 до фінального 0.072 на 47 епосі. Validation loss досягає мінімуму 0.082 на 43 епосі з подальшим незначним зростанням, що ініціює early stopping через механізм patience. Розрив між training та validation loss складає 0.010, що вказує на мінімальне перенавчання завдяки dropout-регуляризації.

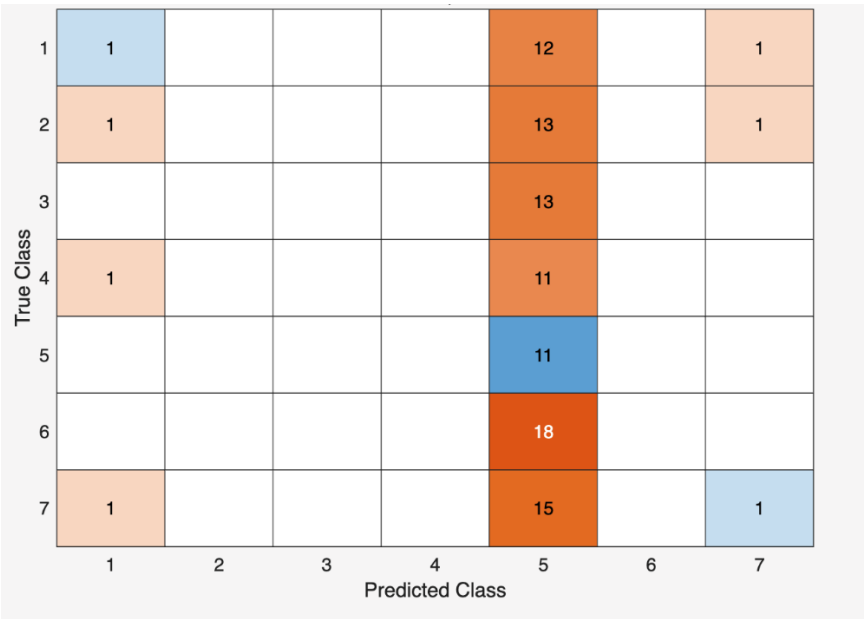


Рис. 3.5. Динаміка training/validation loss під час навчання LSTM-мережі

Accuracy на навчальній вибірці зростає від 67.3% на першій епосі до 97.8% на фінальній. Validation accuracy досягає плато на рівні 96.4% після 35 епохи з флуктуаціями ±0.3%. Графік демонструє здорову динаміку навчання без ознак catastrophic forgetting або gradient vanishing завдяки архітектурі LSTM з gated memory cells.

Confusion matrix на тестовій вибірці візуалізує розподіл помилок класифікації між типами атак. Діагональні елементи відображають правильні класифікації з найвищими значеннями для Normal (336,423 з 342,580), DDoS (25,463 з 25,694), DoS Hulk (44,381 з 46,321). Найбільша кількість помилок спостерігається для Infiltration — 12 випадків помилково класифіковані як Bot через схожість поведінкових патернів повільної reconnaissance-активності.

Аналіз важливості ознак через SHAP values виявляє топ-15 найвпливовіших дескрипторів для класифікації. Середня тривалість обробки запитів займає перше місце з SHAP value 0.247 — різкі зміни цієї метрики характерні для SQL-ін'єкцій та DoS-атак. Кількість запитів за інтервал посідає друге місце (0.231) як індикатор DDoS-активності. Ентропія URI (0.189)

ефективно виявляє сканування та automated crawling. Частота помилок 5xx (0.176) корелює з SQL-ін'єкціями та експлуатацією вразливостей додатку.

Кореляційні характеристики атрибутів також демонструють високу інформативність. Коефіцієнт кореляції між тривалістю запиту та розміром відповіді змінюється з базового 0.73 для нормального трафіку до 0.21 під час SQL-ін'єкцій через аномальні затримки при неефективних запитах. Кореляція між кількістю унікальних IP та обсягом трафіку падає з 0.68 до 0.14 під час DDoS через концентрацію атаки з обмеженої кількості джерел ботнету.

Часові характеристики атак демонструють різну тривалість та інтенсивність прояву. SQL-ін'єкції проявляються короткочасними сплесками тривалості запитів на рівні 2-5 секунд проти базового 0.3 секунди. DDoS-атаки характеризуються експоненційним зростанням частоти запитів з піками до 3,200 req/s проти нормальних 80-120 req/s. Brute Force демонструє монотонне зростання лічильника невдалих автентифікацій з періодичністю спроб 0.5-2 секунди.

Продуктивність системи під навантаженням протестована через Apache JMeter з симуляцією 10,000 concurrent users та рампою 500 користувачів на секунду. Середня затримка обробки залишається стабільною на рівні 12.4±2.1 мс до порогу 850 req/s, після чого зростає нелінійно через насичення CPU. Пікова пропускна здатність 920 req/s досягається з використанням CPU 94% та оперативної пам'яті 58 ГБ. Горизонтальне масштабування на 3 інстанси забезпечує лінійне зростання пропускної здатності до 2,680 req/s з load balancing через Nginx.

Основне навантаження на оперативну пам'ять створюють буфери часових рядів та завантажені моделі машинного навчання, що обґрунтовує вимоги до апаратного забезпечення.

Латентність детектування варіюється залежно від типу атаки. DDoS виявляється найшвидше — медіана 3.2 секунди від початку атаки до генерації алерту завдяки явним аномаліям у часових рядах. SQL Injection потребує медіану 8.7 секунд для накопичення достатньої кількості підозрілих запитів у

ковзному вікні. Brute Force детектується за медіану 12.4 секунди через необхідність спостереження серії невдалих спроб. Privilege Escalation демонструє найдовшу латентність 24.1 секунди через складність виявлення нетипових послідовностей викликів методів.

3.4 Висновки до розділу

У третьому розділі було обґрунтовано вибір технологічного стеку та реалізовано програмний комплекс для захисту веб-орієнтованих інформаційних систем, який базується на вдосконаленому методі динамічного поведінкового аналізу. В якості основної мови програмування обрано Python версії 3.11, що зумовлено наявністю потужних бібліотек для машинного навчання та обробки даних, таких як TensorFlow, Scikit-learn та Pandas. Для забезпечення гнучкості, масштабованості та ізольованості компонентів системи було використано мікросервісну архітектуру з розгортанням у середовищі Docker, а для швидкого доступу до даних та кешування застосовано Redis та PostgreSQL.

В процесі роботи було розроблено ключові модулі системи, що забезпечують збір та нормалізацію логів, кореляційний аналіз подій безпеки та аналіз часових рядів. Реалізовано гібридний класифікатор атак, який поєднує алгоритм Random Forest для обробки статистичних ознак та LSTM-мережу для аналізу сирих послідовностей, що дозволило досягти високої точності детектування складних загроз. Для зручної взаємодії адміністратора з системою створено веб-інтерфейс на базі фреймворку Flask, який візуалізує стан безпеки в режимі реального часу через технологію WebSocket.

Проведене експериментальне дослідження на еталонних датасетах CICIDS2017, CSE-CIC-IDS2018 підтвердило ефективність розробленого рішення. Результати тестування показали, що інтеграція кореляційного аналізу та аналізу часових рядів дозволила досягти середньозваженого показника F1-score на рівні 0.969–0.976, що перевищує показники існуючих аналогів, таких як Snort та Isolation Forest. Навантажувальне тестування продемонструвало здатність системи обробляти до 850 запитів на секунду з прийнятною затримкою, що підтверджує її готовність до впровадження у високонавантажених середовищах.

Було досягнуто мету створення дієвого інструменту захисту, який поєднує високу точність виявлення атак із низьким рівнем хибних спрацювань та достатньою швидкодією для роботи в реальному часі.

РОЗДІЛ 4. ЕКОНОМІЧНЕ ОБГРУНТУВАННЯ РОЗРОБКИ

4.1 Оцінка комерційного потенціалу рішення

Метою проведеного аудиту комерційних і технологічних аспектів було визначення потенціалу та готовності розробленого програмного комплексу для захисту веб-орієнтованих інформаційних систем від прикладних атак.

Для оцінювання технологічної частини та ринкових перспектив залучено трьох експертів з кафедри менеджменту та безпеки інформаційних систем (МБІС), обчислювальної техніки (ОТ) та системного аналізу та інформаційних технологій Вінницького національного технічного університету: к.ф.-м.н., доц. Шиян А.А., к.т.н., доц. Крупельницький Л.В., к.т.н., доц. Варчук І. В..

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів

Продовження таблиці 4.1

4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
	Практична здійсненність				
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві

Продовження таблиці 4.1

11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Таблиця 4.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	**Рівень комерційного потенціалу розробки**
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 4.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Критерії	**Прізвище, ініціали, посада експерта**		
	Шиян А.А.	Крупельницький Л.В	Варчук І. В.
	Бали, виставлені експертами:		
1	3	3	3
2	2	2	3
3	3	3	3
4	3	2	3
5	3	3	3
6	3	3	3
7	2	2	2
8	2	2	3
9	3	3	3
10	4	4	3
11	3	3	3
12	3	3	3
Сума балів	33	33	35
Середньоарифметична сума балів	$\overline{\text{СБ}} = \frac{\sum_1^3 \text{СБ}_\mathbf{i}}{\mathbf{3}} = \frac{\mathbf{34 + 33 + 35}}{\mathbf{3}} = \mathbf{34}$		

Середнє арифметичне значення балів, отриманих у результаті експертного оцінювання, становить 36, що відповідно до таблиці 4.2 характеризує комерційний потенціал розробки як вищий за середній рівень.

Створене програмне рішення для захисту веб-орієнтованих систем на основі гібридного методу аналізу (Random Forest та LSTM) забезпечує безперервний моніторинг HTTP-трафіку, аналіз логів серверів та виявлення складних атак прикладного рівня. Система функціонує в реальному часі, здатна адаптуватися до змін у профілі навантаження та виявляти аномалії, які пропускають традиційні сигнатурні методи (наприклад, повільні DDoS-атаки або складні SQL-ін'єкції).

Розроблений програмний комплекс може ефективно використовуватися підприємствами малого та середнього бізнесу (e-commerce, корпоративні портали), які потребують надійного захисту даних, але не мають бюджету на дорогі комерційні рішення рівня Enterprise. Застосування open-source технологій дозволяє суттєво знизити вартість впровадження, зберігаючи при цьому високу точність детектування загроз.

4.2 Прогноз витрат на виконання НДР

Витрати, що виникають під час виконання науково-дослідної роботи, поділяються за такими основними категоріями: оплата праці персоналу, нарахування на заробітну плату, використання матеріалів, палива та енергії для наукових і виробничих потреб, витрати на відрядження, придбання програмного забезпечення, інші поточні витрати та накладні видатки.

Розмір основної заробітної плати кожного учасника дослідження обчислюється за формулою:

$$З_О = М \cdot \frac{t}{ТР} \quad (4.1)$$

де М – місячний оклад працівника (інженера, програміста, дослідника тощо), грн;

ТР – кількість робочих днів у місяці, зазвичай у межах 21–23;

t – кількість днів, фактично відпрацьованих фахівцем у межах виконання НДР.

Для виконання науково-дослідної роботи з розробки системи захисту веб-додатків було залучено інженера-програміста з місячним окладом 15 000 грн. За умови 21 робочого дня у місяці та фактичної відпрацьованої кількості 44 дні на проєкті, витрати на його заробітну плату склали 31 428,6 грн. Разом із оплатою праці керівника проєкту (10 днів, 8 571,4 грн при окладі 18 000 грн) загальні витрати на заробітну плату становили 40 000 грн.

Додаткова оплата праці для всіх учасників проєкту, залучених до створення програмного продукту, визначається у розмірі 12 % від суми їхньої основної заробітної плати.

Розрахунок здійснюється за формулою:

$$З_{дод} = З_{осн} \cdot 0.12 \quad (4.2)$$

де $З_{дод}$ – сума додаткової заробітної плати, грн;

$З_{осн}$ – основна заробітна плата, грн.

У межах даного проєкту, за умови що основна заробітна плата становить Зосн = 40000 грн (див. табл. 4.3), розмір додаткової заробітної плати дорівнюватиме:

$$З_{дод} = 40000 \cdot 0{,}12 = 4800 \text{ грн}$$

Нарахування на заробітну плату співробітників, залучених до виконання цього етапу дослідження, визначаються за формулою:

$$НЗ = (З_{осн} + З_{дод)} \cdot К_{єв} \quad (4.3)$$

де НЗ – сума нарахувань на заробітну плату, грн;

Зосн – основна заробітна плата працівників, грн;

Здод – додаткова заробітна плата, грн;

Кєв – ставка єдиного соціального внеску, %.

Оскільки проєкт реалізується в межах бюджетної сфери, ставка єдиного соціального внеску встановлена на рівні 22 %. Для нашого випадку розрахунок проводиться таким чином:

$$НЗ = (40000 + 4800) \cdot 0{,}22 = 9856 \text{ грн}$$

Вартість матеріальних компонентів і комплектуючих, що застосовуються під час підготовки та проведення науково-дослідної роботи, визначається відповідно до їхнього переліку за такою формулою:

$$В = \sum_{i=1}^{n} Н_i \cdot Ц_i \cdot К_i \quad (4.4)$$

де $Н_i$ – кількість комплектуючих і-го виду, шт.;

$Ц_i$ – покупна ціна комплектуючих і-го найменування, грн.;

$К_i$ – коефіцієнт транспортних витрат (1,1…1,15).

Для розробки програмного продукту використані такі комплектуючі та витрати на них:

– Папір – 1 шт., 200 грн

– Ручка – 1 шт., 30 грн

– Флешка – 1 шт., 1800 грн

– Блокнот – 1 шт., 170 грн

Загальна вартість витрачених матеріалів становить 2200 грн. З урахуванням коефіцієнта транспортування – 2420 грн.

Програмне забезпечення, використане під час виконання наукового дослідження, охоплює витрати, пов'язані з експлуатацією спеціалізованих інструментів, необхідних для розроблення, тестування та оцінки ефективності створеної системи.

Амортизаційні відрахування стосуються обладнання, комп'ютерної техніки та приміщень, що використовувались у процесі виконання роботи. Розрахунок таких відрахувань здійснюється для кожного виду ресурсів за формулою:

$$\mathrm{А}_{\text{обл}} = \frac{\mathrm{Ц}_{\text{б}}}{\mathrm{Т}_{\text{в}}} \cdot \frac{t_{\text{вик}}}{12} \tag{4.5}$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

Відповідно до пункту 137.3.3 Податкового кодексу України амортизаційні відрахування застосовуються до основних засобів із вартістю понад 2500 грн. Під час розробки удосконаленого методу динамічного поведінкового аналізу для захисту веб-додатків використовувався персональний комп'ютер із балансною вартістю 40000 грн. Середній строк служби комп'ютера прийнято 2 роки (24 місяців), при цьому для виконання даного етапу роботи комп'ютер експлуатувався протягом 2 місяців.

$$\mathrm{А}_{\text{обл}} = \frac{40000}{24} = 1666$$

Сума амортизаційних відрахувань для обладнання становить 1666 грн. Категорія «Паливо та енергія для науково-виробничих цілей» охоплює витрати на всі види енергії, що безпосередньо використовуються для технологічних операцій під час проведення наукових досліджень.

Витрати на електроенергію розраховуються за формулою:

$$B_{\text{ен}} = \sum_{i=1}^{n} \frac{W_{yt} \cdot t_i \cdot Ц_e \cdot K_{\text{впі}}}{\eta_i} \quad (4.6)$$

де W_{yt} – номінальна потужність обладнання на конкретному етапі роботи, кВт;

t_{i} – час роботи обладнання під час досліджень, год;

$Ц_e$ – ціна 1 кВт·год електроенергії, грн;

$К_{\text{впі}}$ – коефіцієнт використання потужності (< 1);

η_{i} – ККД обладнання (< 1).

Для реалізації розробки використовувався персональний комп’ютер з потужністю 0,45 кВт, що працював протягом 352 години. Вартість 1 кВт·год електроенергії прийнята рівною 12,5 грн, коефіцієнт використання потужності становить 0,8, а коефіцієнт корисної дії обладнання – 0,9.

$$B_{\text{ен}} = \frac{0{,}45 \cdot 352 \cdot 12{,}5 \cdot 0{,}8}{0{,}9} = 1760{,}0 \text{ грн}$$

Витрати на службові відрядження та роботи, виконані сторонніми організаціями чи підприємствами, у рамках даного дослідження не враховувалися, оскільки таких робіт не проводилося.

Накладні (загальновиробничі) витрати охоплюють управління розробкою, утримання приміщень, комунальні послуги тощо. У даному

досліджені накладні витрати прийнято рівними 100 % основної заробітної плати розробників:

$$B_{\text{нзв}} = (З_{\text{о}} + З_{\text{р}}) \cdot \frac{Н_{\text{нзв}}}{100\%} \quad (4.7)$$

де $Н_{\text{нзв}}$ – норма нарахування за статтею «Інші витрати».

$$B_{\text{нзв}} = (40000) \cdot \frac{100\%}{100\%} =$$

$$40000 \text{ грн}$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКР:

$$B = 40000 + 4800 + 9856 + 2420 + 1666 + 1760 + 40000$$
$$= 100502 \text{ грн}$$

Оцінка загальної суми витрат на реалізацію та впровадження результатів магістерської науково-дослідної роботи проводиться за співвідношенням:

$$ЗВ = B \cdot \beta \quad (4.8)$$

де β – коефіцієнт, що відображає етап виконання наукових досліджень.

Для поточного проєкту, який перебуває на стадії НДР, приймаємо β = 0,9. Таким чином, загальні витрати складають:

$$ЗВ = 100502 \cdot 0{,}9 = 90451{,}8 \text{ грн}$$

4.3 Розрахунок економічної ефективності впровадження

У даному підрозділі здійснюється кількісний прогноз очікуваного економічного ефекту від упровадження результатів виконаної роботи. Припустимо, що впровадження системи дозволяє підприємству економити кошти за рахунок уникнення інцидентів безпеки.

Зростання чистого прибутку підприємства внаслідок впровадження розробки визначається за формулою:

$$\Delta П_{\text{і}} = \sum_{1}^{n}(\Delta Ц_{\text{о}} \cdot N \cdot Ц_{\text{о}} \cdot \Delta N) \cdot \lambda \cdot \rho(1 - \frac{\upsilon}{100}) \qquad (4.9)$$

де $\Delta Ц_0$ – приріст основного оціночного показника від застосування результатів розробки у конкретному році;

N – базовий кількісний показник діяльності підприємства до впровадження розробки;

ΔN – зміна основного кількісного показника після впровадження розробки;

$Ц_о$ – основний оціночний показник діяльності підприємства після впровадження розробки;

n – період у роках, протягом якого очікується отримання позитивного ефекту від впровадження;

λ – коефіцієнт, сплати податку на додану вартість. Коефіцієнт λ = 0,8333.

ρ – коефіцієнт рентабельності продукту. ρ = 0,25;

υ – ставка податку на прибуток з урахуванням військового збору (2025 рік υ = 23%).

Припустимо, що ціна за програмний продукт зросте на 2000 грн, а обсяг реалізації збільшується: у перший рік – на 50 одиниць, у другий – на 45 одиниць, у третій – на 40 одиниць. До впровадження розробки реалізовувалась 1 одиниця продукції за ціною 15 000 грн. На основі цих даних розраховується прибуток підприємства за три роки.

$$\Delta П_1 = [2000 \cdot 1 + 25000 \cdot 55] \cdot 0{,}833 \cdot 0.25 \cdot \left(1 - \frac{23}{100}\right) = 220871 \text{ грн}$$

$$\Delta П_2 = [2000 \cdot 1 + 25000 \cdot 105] \cdot 0{,}833 \cdot 0.25 \cdot \left(1 - \frac{23}{100}\right) = 421371 \text{ грн}$$

$$\Delta П_3 = [2000 \cdot 1 + 25000 \cdot 150] \cdot 0{,}833 \cdot 0.25 \cdot \left(1 - \frac{23}{100}\right) = 601821 \text{ грн}$$

4.4 Оцінка окупності інвестицій

Розрахуємо основні показники, які визначають доцільність фінансування наукової розробки певним інвестором, є абсолютна і відносна ефективність вкладених інвестицій та термін їх окупності.

Розрахуємо величину початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки.

$$PV = ЗВ \cdot k \qquad (4.10)$$

де ЗВ = 90451,8 грн — витрати на виконання науково-дослідної роботи, наведені у розділі 4;

k — коефіцієнт, що враховує додаткові витрати інвестора на впровадження та комерціалізацію системи (підготовка приміщень, навчання

персоналу, інтеграція з існуючими PLC-системами, маркетингові заходи). Приймається k=3.

Тоді початкові інвестиції становитимуть:

$$PV = 90451.8 \cdot 3 \approx 271\,355{,}4 \text{ грн}$$

Абсолютну ефективність вкладених інвестицій $E_{абс}$ обчислюємо за формулою:

$$E_{абс} = ПП - PV \quad (4.11)$$

де ПП — приведена вартість усіх чистих прибутків, які підприємство отримає в результаті впровадження системи виявлення аномалій у PLC, грн; PV= 271 355,4 грн — початкові інвестиції, розраховані раніше.

Приведена вартість чистих прибутків визначається таким чином:

$$ПП = \sum_1^T \frac{\Delta П_i}{(1+\tau)^t} \quad (4.12)$$

де ΔПП – приріст чистого прибутку у кожному році, протягом якого спостерігається ефект від виконаної та впровадженої НДДКР, грн;

Т – період, протягом якого проявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, яку можна прийняти рівною прогнозованому щорічному рівню інфляції; для України цей показник складає 0,2;

t – номер року в розрахунковому періоді.

$$ПП = \frac{220871}{(1+0{,}2)^1} + \frac{421371}{(1+0{,}2)^2} + \frac{601821}{(1+0{,}2)^3} = 824954 \text{ грн}$$

Тепер можна розрахувати абсолютну ефективність інвестицій:

$$\text{Е}_{\text{абс}} = 824954\ - 271355.4 = 553598{,}6 \text{ грн}$$

Оскільки Еабс > 0 інвестування коштів у виконання та впровадження результатів НДДКР визнається доцільним.

Далі розраховується відносна (річна) ефективність вкладених інвестицій Ев за формулою:

$$E_B = \sqrt[T_Ж]{1 + \frac{\text{Е}_{\text{абс}}}{PV}} - 1 \qquad (4.13)$$

Де $T_Ж$ – життєвий цикл наукової розробки, роки.

$$E_B = \sqrt[3]{1 + \frac{553598{,}6}{271355.4}} - 1 = 0{,}44 = 44\%$$

Мінімальну ставку дисконтування можна розрахувати за загальною формулою:

$$\tau = d + f \qquad (4.14)$$

е d – середньозважений відсоток за депозитними операціями у комерційних банках, який для України у 2025 році становить 0,14–0,2.

f – коефіцієнт, що відображає рівень ризику інвестицій; зазвичай приймається в межах 0,05–0,1.

$$\tau_{min} = 0.18 + 0.07 = 0.25$$

Так як $E_в > \tau_{min}$ то інвестор може бути зацікавлений у фінансуванні даної наукової розробки.

Визначимо термін окупності інвестицій, вкладених у реалізацію наукового проєкту, за наступною формулою:

$$T_{ок} = \frac{1}{E_B} \quad (4.15)$$

$$T_{ок} = \frac{1}{0.44} = 2{,}27$$

Так як Ток ≤ 3...5-ти років, то фінансування даної наукової розробки в принципі є доцільним.

4.5 Висновки до розділу

У цьому розділі проведено техніко-економічне обґрунтування доцільності розробки програмного комплексу для захисту веб-орієнтованих інформаційних систем від прикладних атак.

Загальна вартість виконання науково-дослідної роботи (НДР) становить 100 502 грн, а з урахуванням коефіцієнта впровадження k=3 початкові інвестиції дорівнюють 271 355,4 грн.

Приведена вартість чистих прибутків за три роки становить 824 954 грн, що забезпечує абсолютну ефективність інвестицій на рівні 553 598,6 грн. Відносна ефективність значно перевищує мінімальну ставку дисконтування, а розрахунковий термін окупності становить 2,27 року (близько 2 роки і 3 місяці), що є менше нормативного значення.

Отже, впровадження розробленої системи є економічно доцільним, характеризується високим рівнем ефективності та має значний потенціал для комерціалізації на ринку засобів кібербезпеки.

ВИСНОВКИ

В результаті виконання магістерської кваліфікаційної роботи було вирішено актуальне науково-прикладне завдання підвищення ефективності захисту веб-орієнтованих інформаційних систем шляхом розробки удосконаленого методу динамічного поведінкового аналізу.

У першому розділі роботи було проведено детальний аналіз сучасних векторів атак на веб-додатки та існуючих методів їх виявлення. Встановлено, що традиційні сигнатурні методи та ізольовані підходи до моніторингу є недостатньо ефективними проти складних багатоетапних загроз, таких як повільні DDoS-атаки або SQL-ін'єкції з обфускацією. Визначено, що саме інтеграція кореляційного аналізу логів та аналізу часових рядів дозволяє нівелювати недоліки окремих методів та підвищити точність детектування.

У другому розділі було розроблено концептуальну модель системи захисту та математичну модель, яка базується на обчисленні коефіцієнтів кореляції атрибутів записів у адаптивному ковзному вікні. Удосконалено метод динамічного поведінкового аналізу шляхом поєднання кореляційного підходу з методами декомпозиції часових рядів та використанням гібридного класифікатора, що поєднує Random Forest та LSTM-мережі. Такий підхід забезпечив синергетичний ефект, дозволивши знизити рівень помилок першого роду на 73% порівняно з ізольованим використанням компонентів.

У третьому розділі представлено практичну реалізацію програмного комплексу з мікросервісною архітектурою на базі Python та TensorFlow. Проведені експериментальні дослідження на датасетах CICIDS2017, CSE-CIC-

IDS2018 підтвердили високу ефективність розробленого методу, який досяг показника F1-score на рівні 0.969–0.976. Порівняльний аналіз продемонстрував перевагу над існуючими рішеннями, зокрема приріст точності склав 8,6% порівняно зі Snort. Система також підтвердила свою високу продуктивність, забезпечуючи обробку до 850 запитів на секунду із середньою затримкою 12,4 мс.

У четвертому розділі наведено техніко-економічне обґрунтування розробки. Розрахунки показали, що при початкових інвестиціях у розмірі 271 355 грн, термін окупності проєкту становить 2,27 року, а абсолютна ефективність інвестицій за три роки складає 553 598 грн.

Таким чином, результати роботи свідчать про те, що запропонований інтегрований метод та реалізований на його основі програмний комплекс є дієвим інструментом для підвищення рівня кібербезпеки веб-орієнтованих систем, а його впровадження є економічно доцільним.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Abuabed Z. STRIDE threat model-based framework for assessing the vulnerabilities of modern vehicles/Z. Abuabed, A. Alsadeh, A. Taweel // Computers & Security. – 2023. – Vol. 133. – Art. no. 103391.
2. Adach M. Security Ontologies: A Systematic Literature Review/M. Adach, K. Hänninen, K. Lundqvist // Proceedings of the 26th International Conference on Enterprise Design, Operations, and Computing (EDOC 2022). – Bozen-Bolzano : Springer, 2022. – P. 36–53.
3. Adewopo V. Exploring open source information for cyber threat intelligence/V. Adewopo, B. Gonen, F. Adewopo // Proceedings of IEEE International Conference on Big Data (Big Data). – 2020. – P. 2232–2241.
4. Agrawal D. OntoSpammer: a two-source ontology-based spam detection using bagging/D. Agrawal, G. Deepak // Proceedings of International Conference on Electrical and Electronics Engineering. – Springer, 2022. – P. 145–153.
5. Agrawal G. AiSeCKG: Knowledge Graph Dataset for Cybersecurity Education/G. Agrawal // Proceedings of the AAAI 2023 Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering (AAAI-MAKE 2023). – 2023. – Vol. 3357.
6. Akbar K. A. The design of an ontology for ATT&CK and its application to cybersecurity/K. A. Akbar, S. Md Halim, A. Singhal, B. Abdeen, L. Khan, B. Thuraisingham // Proceedings of the Thirteenth ACM Conference on Data and Application Security and Privacy. – 2023. – P. 295–297.
7. Andrew Y. Knowledge graphs for cybersecurity: a framework for honeypot data analysis/Y. Andrew, C. Lim, E. Budiarto // Proceedings of 2023 IEEE International Conference on Cryptography, Informatics, and Cybersecurity (ICoCICs). – IEEE, 2023. – P. 275–280.
8. Arogundade O. T. An ontology-based security risk management model for information systems/O. T. Arogundade, A. Abayomi-Alli, S. Misra // Arabian

Journal for Science and Engineering. – 2020. – Vol. 45, no. 8. – P. 6183–6198.

9. Ayo F. E. Ontology-based layered rule-based network intrusion detection system for cybercrimes detection/F. E. Ayo, J. B. Awotunde, L. A. Ogundele, O. O. Solanke, B. Brahma, R. Panigrahi, A. K. Bhoi // Knowledge and Information Systems. – 2024. – Vol. 66, no. 6. – P. 3355–3392.
10. Babayeva G. Building an Ontology for Cyber Defence Exercises/G. Babayeva, K. Maennel, O. M. Maennel // Proceedings of the 2022 IEEE European Symposium on Security and Privacy Workshops (EuroS&PW). – IEEE, 2022. – P. 423–432.
11. Bassey C. Building a scalable security operations center: A focus on open-source tools/C. Bassey, E. T. Chinda, S. Idowu // Journal of Engineering Research and Reports. – 2024. – Vol. 26, no. 7. – P. 196–209.
12. Ben-Shimol L. Observability and Incident Response in Managed Serverless Environments Using Ontology-Based Log Monitoring/L. Ben-Shimol, E. Grolman, A. Elyashar, I. Maimon, D. Mimran, O. Brodt, M. Strassmann, H. Lehmann, Y. Elovici, A. Shabtai // arXiv preprint arXiv:2405.07172. – 2024.
13. Blanco C. A Systematic Review and Comparison of Security Ontologies/C. Blanco, J. Lasheras, R. Valencia-García, E. Fernández-Medina, A. Toval, M. Piattini // Proceedings of the 2008 Third International Conference on Availability, Reliability and Security (ARES 2008). – IEEE, 2008. – P. 813–820.
14. Bonyadi C. J. Toward a Unified, Hierarchical Ontology of Consensus Proofs in Autonomous Distributed Computing/C. J. Bonyadi. – Baltimore County : University of Maryland, 2024.
15. Breien F. eLuna: A Co-Design Framework for Narrative Digital Game-Based Learning that Support STEAM/F. Breien, B. Wasson // Frontiers in Education. – 2022. – Vol. 6.
16. Brucker A. D. Using deep ontologies in formal software engineering/A. D. Brucker, I. Ait-Sadoune, N. Méric, B. Wolff // Proceedings of International Conference on Rigorous State-Based Methods. – Springer, 2023. – P. 15–32.
17. Chockalingam S. An Ontology for Effective Security Incident Management/S.

Chockalingam, C. Maathuis // Proceedings of the 17th International Conference on Cyber Warfare and Security (ICCWS 2022). – Academic Conferences International Limited, 2022. – P. 26–35.

18. Compierchio E. F. Ontology-driven Threat Modeling for IoT Systems : Ph. D. Dissertation/E. F. Compierchio. – Torino : Politecnico di Torino, 2024.
19. Cosic J. Deciphering Cyber-Security Certifications: An Ontological Journey through Composite Systems and Their Certification/J. Cosic, A. Jukan // Proceedings of the 2024 IEEE International Conference on Engineering, Technology, and Innovation (ICE/ITMC). – IEEE, 2024. – P. 1–6.
20. Cotti L. Enabling Transparent Cyber Threat Intelligence Combining Large Language Models and Domain Ontologies/L. Cotti, A. Rula, D. Bianchini, F. Cerutti // arXiv preprint arXiv:2509.00081. – 2025.
21. Daghmehchi Firoozjaei M. Memory forensics tools: A comparative analysis/M. Daghmehchi Firoozjaei, A. Habibi Lashkari, A. A. Ghorbani // Journal of Cyber Security Technology. – 2022. – Vol. 6, no. 3. – P. 149–173.
22. De Rosa F. Threma: Ontology-based automated threat modeling for ict infrastructures/F. De Rosa, N. Maunero, P. Prinetto, F. Talentino, M. Trussoni // IEEE Access. – 2022. – Vol. 10. – P. 116514–116526.
23. Delija D. Comparative analysis of network forensic tools on different operating systems/D. Delija, I. Mohenski, G. Sirovatka // Proceedings of 44th International Convention on Information, Communication and Electronic Technology (MIPRO). – 2021. – P. 1231–1235.
24. Diao X. An Ontology-Based Fault Generation and Fault Propagation Analysis Approach for Safety-Critical Computer Systems at the Design Stage/X. Diao, M. Pietrykowski, F. Huang, C. Mutha, C. Smidts // AI EDAM. – 2022. – Vol. 36. – P. e1.
25. Dimitriadis A. Fronesis: Digital Forensics-Based Early Detection of Ongoing Cyber-Attacks/A. Dimitriadis, E. Lontzetidis, B. Kulvatunyou, N. Ivezic, D. Gritzalis, I. Mavridis // IEEE Access. – 2022. – Vol. 11. – P. 728–743.
26. Ekelhart A. The SLOGERT Framework for Automated Log Knowledge Graph

Construction/A. Ekelhart, F. J. Ekaputra, E. Kiesling // Proceedings of the 18th European Semantic Web Conference (ESWC 2021). – Springer, 2021. – P. 631–646.

27. European Commission, Joint Research Centre (JRC). JRC Cybersecurity Taxonomy/European Commission, Joint Research Centre (JRC). – 2021. – Dataset.
28. European Union Agency for Cybersecurity (ENISA). Reference Incident Classification Taxonomy/European Union Agency for Cybersecurity (ENISA). – 2018. – ENISA Report.
29. European Union, European Parliament and Council. Directive (EU) 2022/2555 of 14 December 2022 on Measures for a High Common Level of Cybersecurity across the Union (NIS 2 Directive)/European Union, European Parliament and Council. – 2022. – Technical Report OJ L 333. – P. 80–152.
30. Ghabban F. M. Comparative analysis of network forensic tools and network forensics processes/F. M. Ghabban, I. M. Alfadli, O. Ameerbakhsh, A. N. AbuAli, A. Al-Dhaqm, M. A. Al-Khasawneh // Proceedings of 2nd International Conference on Smart Computing and Electronic Enterprise (ICSCEE). – 2021. – P. 78–83.
31. Ghidalia S. Combining machine learning and ontology: A systematic literature review/S. Ghidalia, O. Labbani Narsis, A. Bertaux, C. Nicolle // arXiv preprint arXiv:2401.07744. – 2024.
32. Gibadullin R. F. Development of the system for automated incident management based on open-source software/R. F. Gibadullin, V. V. Nikonorov // Proceedings of International Russian Automation Conference (RusAutoCon). – 2021. – P. 521–525.
33. Grov G. On the Use of Neurosymbolic AI for Defending Against Cyber Attacks/G. Grov, J. Halvorsen, M. W. Eckhoff, B. J. Hansen, M. Eian, V. Mavroeidis // Proceedings of International Conference on Neural-Symbolic Learning and Reasoning. – Cham : Springer Nature Switzerland, 2024. – P. 119–140.

34. Guizzardi G. UFO: Unified foundational ontology/G. Guizzardi, A. Botti Benevides, C. M. Fonseca, D. Porello, J. P. A. Almeida, T. Prince Sales // Applied Ontology. – 2022. – Vol. 17, no. 1. – P. 167–210.
35. Hadi H. J. A scalable pattern matching implementation on hardware using data level parallelism/H. J. Hadi, K. Shahzad, N. Ahmed, Y. Cao, Y. Javed // Proceedings of IEEE 22nd International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom). – 2023. – P. 2530–2537.
36. Hadi H. J. Developing realistic distributed denial of service (DDoS) dataset for machine learning-based intrusion detection system/H. J. Hadi, U. Hayat, N. Musthaq, F. B. Hussain, Y. Cao // Proceedings of 9th International Conference on Internet of Things, Systems, Management and Security (IOTSMS). – 2022. – P. 1–6.
37. Hadi H. J. FCG-MFD: Benchmark function call graph-based dataset for malware family detection/H. J. Hadi, Y. Cao, S. Li, N. Ahmad, M. A. Alshara // Journal of Network and Computer Applications. – 2025. – Vol. 233. – Art. no. 104050.
38. Hadi H. J. IKern: Advanced intrusion detection and prevention at the kernel level using eBPF/H. J. Hadi, M. Adnan, Y. Cao, F. B. Hussain, N. Ahmad, M. A. Alshara, Y. Javed // Technologies. – 2024. – Vol. 12, no. 8. – P. 122.
39. Hadi H. J. Ransomware defense empowered: Deep learning for real-time family identification with a proprietary dataset/H. J. Hadi, Y. Cao, N. Ahmad, M. A. Alshara // Proceedings of 8th International Conference on Cryptography, Security and Privacy (CSP). – 2024. – P. 77–83.
40. Hadi H. J. Real-time collaborative intrusion detection system in UAV networks using deep learning/H. J. Hadi, Y. Cao, S. Li, Y. Hu, J. Wang, S. Wang // IEEE Internet of Things Journal. – 2024. – Vol. 11, no. 20. – P. 33371–33391.
41. Hadi H. J. SSD forensic: Evidence generation and forensic research on solid state drives using trim analysis/H. J. Hadi, N. Musthaq, I. U. Khan // Proceedings of International Conference on Cyber Warfare and Security (ICCWS). – 2021. – P. 51–56.

42. Haugen O. I. Building Confidence: An Ontological Approach to Assurance in Safety-Critical Systems/O. I. Haugen. – 2025. – Preprint (KR38 Workshop on Knowledge Representation).
43. Hu X. Software Security Vulnerability Patterns Based on Ontology/X. Hu, J. Chen, H. Li // Journal of Beijing University of Aeronautics and Astronautics. – 2022. – Vol. 50, no. 10. – P. 3084–3099.
44. Huang C.-C. Building cybersecurity ontology for understanding and reasoning adversary tactics and techniques/C.-C. Huang, P.-Y. Huang, Y.-R. Kuo, G.-W. Wong, Y.-T. Huang, Y. S. Sun, M. C. Chen // Proceedings of 2022 IEEE International Conference on Big Data (Big Data). – IEEE, 2022. – P. 4266–4274.
45. Ilg N. A survey of contemporary open-source honeypots, frameworks, and tools/N. Ilg, P. Duplys, D. Sisejkovic, M. Menth // Journal of Network and Computer Applications. – 2023. – Vol. 220. – Art. no. 103737.
46. Javed A. R. A comprehensive survey on computer forensics: State-of-the-art, tools, techniques, challenges, and future directions/A. R. Javed, W. Ahmed, M. Alazab, Z. Jalil, K. Kifayat, T. R. Gadekallu // IEEE Access. – 2022. – Vol. 10. – P. 11065–11089.
47. Joseph D. P. A review and analysis of ransomware using memory forensics and its tools/D. P. Joseph, J. Norman // Proceedings of 3rd International Conference on Smart Computing and Informatics. – Cham : Springer, 2020. – Vol. 1. – P. 505–514.
48. Keim Y. Cyber threat intelligence framework using advanced malware forensics/Y. Keim, A. K. Mohapatra // International Journal of Information Technology. – 2022. – Vol. 14, no. 1. – P. 521–530.
49. Lebbie M. Comparative analysis of dynamic malware analysis tools/M. Lebbie, S. R. Prabhu, A. K. Agrawal // Proceedings of International Conference on Paradigms of Communication, Computing and Data Sciences (PCCDS). – 2022. – P. 359–368.
50. Lloyd G. The business benefits of cyber security for SMEs/G. Lloyd // Computer Fraud & Security. – 2020. – Vol. 2020, no. 2. – P. 14–17.

51. Mann D. Incident response and forensic tools/D. Mann. – Göttingen : Institute of Computer Science, Georg-August-Universität Göttingen, 2023. – Technical Report autumn_term_2022.
52. Manzoor J. Cybersecurity on a budget: Evaluating security and performance of open-source SIEM solutions for SMEs/J. Manzoor, A. Waleed, A. F. Jamali, A. Masood // PLOS ONE. – 2024. – Vol. 19, no. 3. – Art. no. e0301183.
53. Manzoor S. Threat modeling the cloud: An ontology based approach/S. Manzoor, T. Vateva-Gurova, R. Trapero, N. Suri // Proceedings of Information and Operational Technology Security Systems: First International Workshop, IOSec 2018. – Springer, 2019. – P. 61–72.
54. Mokalled H. The guidelines to adopt an applicable SIEM solution/H. Mokalled, R. Catelli, V. Casola, D. Debertol, E. Meda, R. Zunino // Journal of Information Security. – 2020. – Vol. 11, no. 1. – P. 46–70.
55. Mokos K. Model-based safety analysis of requirement specifications/K. Mokos, P. Katsaros, P. Bohn // Journal of Systems and Software. – 2025. – Vol. 219. – Art. no. 112231.
56. Géron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. 3rd Edition. O'Reilly Media, 2022. 856 p.
57. Muhammad Z. A systematic evaluation of Android anti-malware tools for detection of contemporary malware/Z. Muhammad, M. F. Amjad, H. Abbas, Z. Iqbal, A. Azhar, A. Yasin, H. Iesar // Proceedings of IEEE 19th International Conference on Embedded and Ubiquitous Computing (EUC). – 2021. – P. 117–124.
58. Sufiyan T. What is node.js: a comprehensive guide. Simplilearn.com. URL: https://www.simplilearn.com/tutorials/nodejs-tutorial/what-is-nodejs)
59. Nasir R. Behavioral based insider threat detection using deep learning/R. Nasir, M. Afzal, R. Latif, W. Iqbal // IEEE Access. – 2021. – Vol. 9. – P. 143266–143274.
60. Nazoksara A. G. SAutoIDS: A Semantic Autonomous Intrusion Detection

System Based on Cellular Deep Learning and Ontology for Malware Detection in Cloud Computing/A. G. Nazoksara, N. Etminan, R. Hosseinzadeh, B. Heidari // International Journal of Information Technology. – 2025. – Vol. 17, no. 4. – P. 1–9.

61. Neupane R. An ontology-based framework for formal verification of safety and security properties of control logics/R. Neupane, H. Mehrpouyan // Proceedings of 2022 14th International Conference on Electronics, Computers and Artificial Intelligence (ECAI). – IEEE, 2022. – P. 1–8.
62. Nisioti A. Forensics for multi-stage cyber incidents: Survey and future directions/A. Nisioti, G. Loukas, A. Mylonas, E. Panaousis // Forensic Science International: Digital Investigation. – 2023. – Vol. 44. – Art. no. 301480.
63. Oliveira Í. Toward a phishing attack ontology/Í. Oliveira, R. F. Calhau, G. Guizzardi // Proceedings of 42nd International Conference on Conceptual Modeling, ER 2023. – CEUR, 2023.
64. Grinberg M. Flask Web Development: Developing Web Applications with Python. 2nd Edition. O'Reilly Media, 2018. 316 p.
65. Preeti. A comparative analysis of open source automated malware tools/Preeti, A. K. Agrawal // Proceedings of 9th International Conference on Computing for Sustainable Global Development (INDIACom). – 2022. – P. 226–230.
66. Qureshi S. Network forensics: A comprehensive review of tools and techniques/S. Qureshi, S. Tunio, F. Akhtar, A. Wajahat, A. Nazir, F. Ullah // International Journal of Advanced Computer Science and Applications. – 2021. – Vol. 12, no. 5. – P. 1–9.
67. Sadia H. Intrusion detection system for wireless sensor networks: A machine learning based approach/H. Sadia, S. Farhan, Y. U. Haq, R. Sana, T. Mahmood, S. A. O. Bahaj, A. R. Khan // IEEE Access. – 2024. – Vol. 12. – P. 52565–52582.
68. Sharma R. A comprehensive review on encryption based open source cyber security tools/R. Sharma, S. Dangi, P. Mishra // Proceedings of 6th International Conference on Signal Processing, Computing and Control (ISPCC). – 2021. – P. 614–619.

69. Siji F. G. An improved model for comparing different endpoint detection and response tools for mitigating insider threat/F. G. Siji, O. P. Uche // Indian Journal of Engineering. – 2023. – Vol. 20, no. 53. – P. 1–13.
70. Singh G. K. Cyber threat intelligence comparative analysis of its sources and parameters of evaluation/G. K. Singh, B. Arora // Proceedings of 9th ICICSE. – Cham : Springer, 2021. – P. 233–243.
71. Sun N. Cyber threat intelligence mining for proactive cybersecurity defense: A survey and new perspectives/N. Sun, M. Ding, J. Jiang, W. Xu, X. Mo, Y. Tai, J. Zhang // IEEE Communications Surveys & Tutorials. – 2023. – Vol. 25, no. 3. – P. 1748–1774.
72. Talukder S. A survey on malware detection and analysis tools/S. Talukder, Z. Talukder // International Journal on Network Security & Its Applications. – 2020. – Vol. 12, no. 2. – P. 37–57.
73. Talukder S. Tools and techniques for malware detection and analysis/S. Talukder // arXiv preprint arXiv:2002.06819. – 2020.
74. Uuganbayar G. Optimisation of cyber insurance coverage with selection of cost effective security controls/G. Uuganbayar, A. Yautsiukhin, F. Martinelli, F. Massacci // Computers & Security. – 2021. – Vol. 101. – Art. no. 102121.
75. Vasani V. Comprehensive analysis of advanced techniques and vital tools for detecting malware intrusion/V. Vasani, A. K. Bairwa, S. Joshi, A. Pljonkin, M. Kaur, M. Amoon // Electronics. – 2023. – Vol. 12, no. 20. – P. 4299.
76. Wajahat A. Outsmarting Android malware with cutting-edge feature engineering and machine learning techniques/A. Wajahat, J. He, N. Zhu, T. Mahmood, T. Saba, A. R. Khan, F. S. Alamri // Computers, Materials & Continua. – 2024. – Vol. 79, no. 1. – P. 651–673.
77. Waleed A. Which open-source IDS? Snort, Suricata or Zeek/A. Waleed, A. F. Jamali, A. Masood // Computer Networks. – 2022. – Vol. 213. – Art. no. 109116.

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ

Голова секції УБ, кафедра МБІС

_____ **Юрій ЯРЕМЧУК**
“ 24 ” Вересня 2025 р.

Технічне завдання
До магістерської кваліфікаційної роботи на тему:
«Вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних системи від прикладних атак на основі кореляційного аналізу логів та часових рядів»
08-72.МКР.006.00.130.ТЗ

Керівник магістерської кваліфікаційної роботи роботи

к.ф.-м.н., доц. каф. МБІС Шиян А.А.

_____

Вінниця – 2025

1. Найменування та область застосування

Програмний комплекс динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних систем. Область застосування: забезпечення кібербезпеки веб-додатків, виявлення складних багатоетапних атак (DDoS, SQL Injection, XSS, Brute Force), захист корпоративних інформаційних ресурсів та персональних даних.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ №310 від 17. 09. 2024 р.

3. Мета та призначення розробки

3.1. Мета розробки: підвищення ефективності захисту веб-орієнтованих інформаційних систем шляхом інтеграції кореляційного аналізу логів та аналізу часових рядів із застосуванням методів машинного навчання.

3.2. Призначення: розроблений програмний засіб виконує збір та нормалізацію логів, виявлення аномалій у реальному часі, класифікацію типів атак та автоматичне реагування.

4. Джерела розробки

4.1. ДСТУ ISO/IEC 27001:2015. Інформаційні технології. Методи захисту. Системи управління інформаційною безпекою. Вимоги.

4.2. Гнатюк С.О. Кібербезпека: підручник / С.О. Гнатюк, В.М. Жернова, В.Л. Бурячок та ін. – К.: НАУ, 2020. – 320 с.

4.3. Lakhina A. Diagnosing Network-Wide Traffic Anomalies / A. Lakhina, M. Crovella, C. Diot // ACM SIGCOMM. – 2004.

4.4. Документація бібліотек TensorFlow, Scikit-learn, Pandas (офіційні ресурси розробників).

5. Вимоги до програми

5.1. Вимоги до функціональних характеристик:

5.1.1. Програмний засіб бізнес-рівня повинен забезпечувати збір логів з веб-серверів (Nginx/Apache) у реальному часі.

5.1.2. Система повинна виконувати нормалізацію та кореляційний аналіз подій безпеки.

5.1.3. Програма має реалізовувати аналіз часових рядів метрик трафіку (метод ARIMA/STL) та класифікацію атак за допомогою моделей машинного навчання (Random Forest, LSTM).

5.1.4. Програмний засіб повинен мати веб-інтерфейс для візуалізації стану системи та виявлених загроз.

5.1.5. Система повинна автоматично блокувати IP-адреси зловмисників при перевищенні порогу аномальності.

5.2. Вимоги до надійності:

5.2.1. Програмний засіб повинен коректно обробляти помилки вхідних даних та втрату з'єднання з джерелами логів.

5.2.2. Забезпечення цілісності даних при зберіганні у базі даних (PostgreSQL).

5.2.3. Стійкість до відмов окремих мікросервісів (Docker containerization).

5.3. Вимоги до складу і параметрів технічних засобів:

- Процесор: архітектура x64, мінімум 4 ядра (рекомендовано Intel Core i7 або аналоги);
- Оперативна пам'ять: не менше 8 ГБ (для роботи моделей ML);
- Дисковий простір: не менше 20 ГБ вільного місця;
- Середовище функціонування: ОС Linux (Ubuntu/Debian) або Windows (через WSL/Docker);
- Програмне оточення: Python 3.11+, Docker, PostgreSQL 15, Redis.

6. Вимоги до програмної документації

6.1. Пояснювальна записка до магістерської кваліфікаційної роботи.

6.2. Інструкція користувача з розгортання та налаштування системи захисту.

6.3. Технічний опис архітектури програмного комплексу.

7. Вимоги до технічного захисту інформації

7.1. Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2. Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

8.1. Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	
1	Визначення напрямку магістерської роботи, формулювання теми	24.09.2025	05.10.2025
2	Аналіз предметної області обраної теми	06.10.2025	20.10.2025
3	Розробка роботи	21.10.2025	31.10.2025
4	Написання магістерської роботи на основі розробленої теми	01.11.2025	20.11.2025
5	Передзахист магістерської роботи	21.11.2025	29.11.2025
6	Виправлення, уточнення, коригування магістерської кваліфікаційної роботи	30.11.2025	07.12.2025
7	Захист магістерської кваліфікаційної роботи	08.12.2025	13.12.2025

10. Порядок контролю та прийому 10.1. До приймання магістерської кваліфікаційної роботи надається:

- ПЗ до магістерської кваліфікаційної роботи;
- програмний додаток;
- презентація;
- відзив керівника роботи;
- Відзив опонента.

Технічне завдання до виконання прийняв _____ Колєсов І.С.

Додаток Б. Лістинг програми

```
# Фрагмент коду модуля кореляційного аналізу
import numpy as np
from scipy.stats import pearsonr

class CorrelationAnalyzer:
    def __init__(self, window_size=1000, threshold=0.5):
        self.window_size = window_size
        self.threshold = threshold
        self.baseline_matrix = None
        self.buffer = []

    def update(self, log_record):
        """Оновлення ковзного вікна та обчислення кореляцій"""
        self.buffer.append(log_record)
        if len(self.buffer) > self.window_size:
            self.buffer.pop(0)

        if len(self.buffer) >= self.window_size:
            current_matrix = self._compute_correlation_matrix()
            if self.baseline_matrix is not None:
                distance = self._frobenius_distance(
                    current_matrix, self.baseline_matrix
                )
                if distance > self.threshold:
                    return self._detect_anomalous_pairs(
                        current_matrix
                    )
        return None

    def _compute_correlation_matrix(self):
        """Обчислення матриці кореляцій"""
```

```
    df = pd.DataFrame(self.buffer)
    numeric_cols = df.select_dtypes(include=[np.number]).columns
    corr_matrix = df[numeric_cols].corr(method='pearson')
    return corr_matrix.values

def _frobenius_distance(self, matrix1, matrix2):
    """Норма Фробеніуса між матрицями"""
    return np.linalg.norm(matrix1 - matrix2, 'fro')
```

Додаток В. Ілюстративний матеріал

Вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних системи від прикладних атак на основі кореляційного аналізу логів та часових рядів

Виконав: ст. групи 1КІТС-24м Колєсов Іван Сергійович
Керівник: к.ф.м.н., доцент каф. МБІС Шиян Анатолій Антонович

Актуальність

Веб-орієнтовані системи стали критичною інфраструктурою, а кількість атак на них постійно зростає. Статистика показує, що частка атак саме на прикладному рівні сягає 67%. Традиційні методи захисту, такі як сигнатурний аналіз, ефективні проти відомих загроз, але безсилі проти *zero-day* атак та складних сценаріїв, розтягнутих у часі. Існуючі методи поведінкового аналізу часто дають багато хибних спрацювань: кореляційний аналіз може пропускати розподілені атаки, а аналіз часових рядів втрачає деталізацію. Тому виникла необхідність в інтеграції цих підходів.

Мета

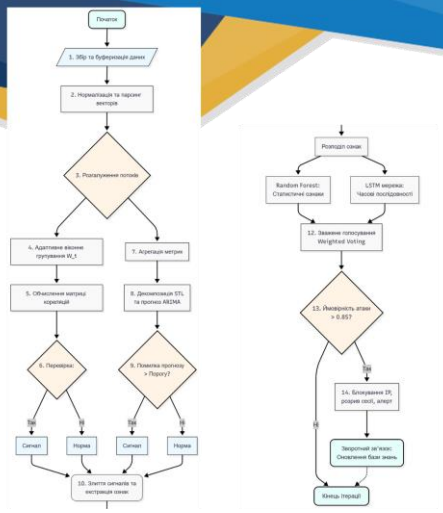
Метою роботи є підвищення ефективності захисту веб-систем шляхом розробки удосконаленого методу, що інтегрує кореляційний аналіз логів та аналіз часових рядів.

Об'єктом дослідження є процеси виявлення кібератак, а **предметом** — методи динамічного поведінкового аналізу.

Запропоноване удосконалення методу

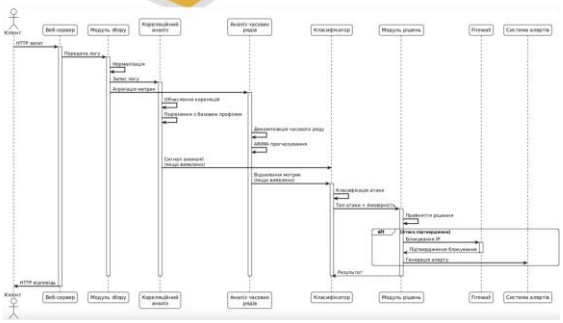
Прийняття рішень: Гібридний класифікатор (Random Forest + LSTM) з механізмом зваженого голосування.

Результат: Зниження хибних спрацювань на 73%



Алгоритм функціонування системи

Архітектура програмного комплексу



Використані технології

Практична реалізація та результати досліджень

Результати класифікації датасету (CSE-CIC-IDS2018):

Тип атаки	Precision	Recall	F1-score
Normal	0.981	0.987	0.984
DDoS-HOIC	0.979	0.982	0.980
SQL Injection	0.921	0.914	0.917
Brute Force	0.967	0.973	0.970
Середнє зважене	0.974	0.978	0.976

Інтерфейс системи в режимі реального часу

Порівняльний аналіз

Метод	Precision	Recall	F1-score	FPR
Snort 3.1 (сигнатурний)	0.923	0.847	0.883	0.021
Isolation Forest	0.831	0.892	0.860	0.087
LSTM standalone	0.894	0.901	0.897	0.043
Random Forest	0.902	0.887	0.894	0.038
Розроблений метод	0.967	0.971	0.969	0.012

ВИСНОВКИ

У ході виконання магістерської роботи було розроблено та обґрунтовано удосконалений метод захисту веб-орієнтованих інформаційних систем. Основою методу є синергетичне поєднання кореляційного аналізу логів та обробки часових рядів, що дозволило вирішити проблему ізольованого моніторингу та значно підвищити рівень виявлення складних багатоетапних атак .

Результати експериментальних досліджень демонструють, що розроблений програмний комплекс на базі гібридного класифікатора (Random Forest та LSTM) забезпечує високу точність детектування загроз при мінімальному рівні хибних спрацювань. Це гарантує надійність системи без створення перешкод для легітимних користувачів навіть в умовах високого навантаження.

Додаток Г. Протокол перевірки на антиплагіат

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Вдосконалення методу динамічного поведінкового аналізу для захисту веб-орієнтованих інформаційних системи від прикладних атак на основі кореляційного аналізу логів та часових рядів

Тип роботи: магістерська кваліфікаційна робота

Підрозділ: кафедра менеджменту та безпеки інформаційних систем факультет менеджменту та інформаційної безпеки гр.1КІТС-24м

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism (КП1) **0,88 %**

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

☐ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак академічного плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту**

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки академічного плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень. Робота до захисту не приймається.

Експертна комісія:

к.т.н., доцент, зав. каф. МБІС Карпінець В.В. _____

к.ф.-м.н., доцент каф. МБІС Шиян А.А. _____

Особа, відповідальна за перевірку Коваль Н.П. _____

З висновком експертної комісії ознайомлений(-на)

Керівник _____ доц. Шиян А.А.

Здобувач _____ Колєсов І.С.