

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

Магістр

(назва освітнього ступеня)

на тему: Методи та засоби виявлення та реагування на атаки в IoT-середовищах шляхом аналізу мережевого трафіку

Виконав: студент(ка) 6 курсу, групи СІм-61
спеціальності 123 «Комп'ютерна інженерія»

(шифр і назва спеціальності)

	<u>Міндзіброцький Р.С.</u> (підпис) (прізвище та ініціали)
Керівник	<u>Луцків А.М.</u> (підпис) (прізвище та ініціали)
Нормоконтроль	<u>Луцик Н.С.</u> (підпис) (прізвище та ініціали)
Завідувач кафедри	<u>Осухівська Г.М.</u> (підпис) (прізвище та ініціали)
Рецензент	<u></u> (підпис) (прізвище та ініціали)

Тернопіль
2025

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних систем та мереж
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
Осухівська Г.М.
(підпис) (прізвище та ініціали)

«17» листопада 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня *магістр*
(назва освітнього ступеня)

за спеціальністю *123 «Комп'ютерна інженерія»*
(шифр і назва спеціальності)

студенту *Міндзіброцькому Роману Сергійовичу*
(прізвище, ім'я, по батькові)

1. Тема роботи *Методи та засоби виявлення та реагування на атаки в IoT-середовищах шляхом аналізу мережевого трафіку*

Керівник роботи *Луцьків Андрій Мирославович, к.т.н., доцент*
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 14 » листопада 2025 року № 4/7-987

2. Термін подання студентом завершеної роботи *16 грудня 2025 р.*

3. Вихідні дані до роботи *наукові літературні джерела, дампи мережевого трафіку, специфікація моделі LLM, вимоги до програмної реалізації.*

4. Зміст роботи (перелік питань, які потрібно розробити)

Вступ

1 Аналітична частина

2 Теоретична частина

3 Практична частина

4 Охорона праці та безпека в надзвичайних ситуаціях

Висновки

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

1. Актуальність і мета дослідження.

2. Завдання, об'єкт і предмет, наукова новизна і практична цінність дослідження.

3. Агентна архітектура системи виявлення та реагування на iot атаки

4. Процес збору та опрацювання мережевого трафіку

5. Алгоритм інтелектуального реагування на загрози

6. Порівняльна оцінка ефективності Ilt-агентів у задачах детектування кіберзагроз

7. Результати моніторингу та реагування

8. Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
<i>Охорона праці</i>	<i>Осухівська Г.М., зав.каф. КС</i>	<i>04.12.2025</i>	
<i>Безпека в надзвичайних ситуаціях</i>	<i>Стручок В.С. ст. викладач каф. ОХ</i>		

7. Дата видачі завдання 17.11.2025р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	<i>Огляд літературних джерел. Постановка завдання на кваліфікаційну роботу.</i>	<i>17.11.2025р. – 23.11.2025р.</i>	<i>Викон.</i>
2.	<i>Робота над теоретичною частиною</i>	<i>24.11.2025р. – 02.12.2025р.</i>	<i>Викон.</i>
3.	<i>Робота над практичною частиною</i>	<i>03.12.2025р. – 10.12.2025р.</i>	<i>Викон.</i>
4.	<i>Охорона праці та безпека в надзвичайних ситуаціях</i>	<i>04.12.2025р. – 10.12.2025р.</i>	<i>Викон.</i>
5.	<i>Оформлення пояснювальної записки і графічного матеріалу</i>	<i>11.12.2025р. – 19.12.2025р.</i>	<i>Викон.</i>
6.	<i>Попередній захист кваліфікаційної роботи магістра</i>	<i>16.12.2025р.</i>	<i>Викон.</i>
7.	<i>Захист кваліфікаційної роботи магістра</i>	<i>23.12.2025р.</i>	<i>Викон.</i>

Студент

(підпис)*Міндзіброцький Р. С.*

(прізвище та ініціали)

Керівник роботи

(підпис)*Луцків А. М.*

(прізвище та ініціали)

АНОТАЦІЯ

Міндзіброцький Р. С. Методи та засоби виявлення та реагування на атаки в IoT-середовищах шляхом аналізу мережевого трафіку: робота на здобуття кваліфікаційного ступеня магістра: спец. 123 – комп'ютерна інженерія / наук.кер. Луцків А. М. Тернопіль: Тернопільський національний технічний університет імені Івана Пулюя, 2025.

Ключові слова: інтернет речей, аналіз мережевого трафіку, штучний інтелект, AI-агент, виявлення аномалій, автоматизоване реагування, кіберзахист.

Кваліфікаційна робота присвячена дослідженню методів і засобів інтелектуального виявлення кібератак в мережах Інтернету речей з використанням концепції автономних AI-агентів. Робота містить аналіз існуючих систем виявлення вторгнень, а також акцентує увагу на можливостях використання локальних великих мовних моделей для аналізу мережевого трафіку.

Для створення системи захисту використовується програмний комплекс на базі бібліотеки Scapy для захоплення пакетів та локально розгорнута модель Qwen. Отримані дані про трафік серіалізуються у формат JSON і разом із системним промптом передаються агенту, який за допомогою логічного міркування обирає відповідні інструменти для реагування.

Результати цього дослідження можуть мати практичне значення для створення адаптивних систем кібербезпеки, що здатні працювати автономно та забезпечувати конфіденційність даних без передачі інформації у хмару.

ANNOTATION

Mindzibrotskyi R. S. Methods and tools for detecting and responding to attacks in IoT environments through network-traffic analysis. Master's Graduation Thesis: speciality 123 – Computer engineering / supervisor Lutskevych A. M. Ternopil: Ternopil Ivan Puluj National Technical University, 2025.

Keywords: Internet of Things, network traffic analysis, artificial intelligence, AI agent, anomaly detection, automated response, cybersecurity.

The master's thesis is dedicated to researching methods and means for intelligent cyberattack detection in Internet of Things networks using the concept of autonomous AI agents. The study includes an analysis of existing intrusion detection systems and highlights the potential of using local Large Language Models for network traffic analysis.

For the security system development, a software solution based on the Scapy library for packet capturing and a locally deployed Qwen model is utilized. The captured traffic data is serialized into JSON format and, along with a system prompt, is passed to the agent, which employs logical reasoning to select appropriate response tools.

The results of this research hold practical significance for developing adaptive cybersecurity systems capable of operating autonomously and ensuring data privacy without transmitting information to the cloud.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ	7
ВСТУП	8
РОЗДІЛ 1 АНАЛІТИЧНА ЧАСТИНА	11
1.1. Аналіз сучасного стану безпеки в IoT-середовищах	11
1.2. Класифікація атак на інфраструктуру Інтернету речей	15
1.3. Огляд існуючих методів та засобів виявлення вторгнень	18
1.4. Концепція AI-агентів та LLM у кібербезпеці.....	21
1.5. Висновки до розділу 1	23
РОЗДІЛ 2 ТЕОРЕТИЧНА ЧАСТИНА	25
2.1. Методи попередньої обробки та серіалізації мережевого трафіку	25
2.2. Алгоритмічне забезпечення агентної взаємодії	28
2.3. Розробка системних інструкцій та контекстне управління	29
2.4. Архітектура системи та методика вибору базової моделі	35
2.5. Висновки до розділу 2	38
РОЗДІЛ 3 ПРАКТИЧНА ЧАСТИНА	40
3.1. Програмна реалізація інтелектуального агента	40
3.2. Обґрунтування вибору LLM та методика синтетичного тестування	45
3.3. Аналіз результатів роботи системи	48
3.4. Аналіз ефективності та рекомендації.....	55
3.5. Висновки до розділу 3	57
РОЗДІЛ 4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ	59
4.1. Охорона праці	59
4.2. Безпека в надзвичайних ситуаціях	61
ВИСНОВКИ	64
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	65
Додаток А Тези конференцій	
Додаток Б Системний промпт	

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AES (англ. Advanced Encryption Standard) удосконалений стандарт шифрування.

API (англ. Application Programming Interface) інтерфейс прикладного програмування.

DDoS (англ. Distributed Denial of Service) розподілена атака на відмову в обслуговуванні.

DNS (англ. Domain Name System) система доменних імен.

GGUF (англ. GPT-Generated Unified Format) уніфікований формат файлів для локального запуску моделей.

IDS (англ. Intrusion Detection System) система виявлення кібератак.

JSON (англ. JavaScript Object Notation) текстовий формат обміну даними.

LLM (англ. Large Language Model) велика мовна модель.

MCP (англ. Model Context Protocol) стандарт підключення даних.

MQTT (англ. Message Queuing Telemetry Transport) протокол обміну даними для Інтернету речей.

NPU (англ. Neural Processing Unit) процесор для штучного інтелекту.

OT (англ. Operational Technology) технології керування промисловим обладнанням.

ReAct (англ. Reasoning and Acting) метод «міркування та дія».

SQL (англ. Structured Query Language) мова запитів до баз даних.

SSH (англ. Secure Shell) протокол безпечного віддаленого доступу.

TCP (англ. Transmission Control Protocol) протокол надійної передачі даних.

TLS (англ. Transport Layer Security) протокол шифрування зв'язку.

TTFT (англ. Time To First Token) час реакції моделі.

ВСТУП

Актуальність роботи. Стрімка інтеграція Інтернету речей у критичні інфраструктури створює нові вектори кіберзагроз, з якими важко впоратися традиційними засобами через обмежені ресурси кінцевих пристроїв. Забезпечення безпеки в таких гетерогенних мережах вимагає механізмів, здатних адаптуватися до нестандартної поведінки злоумисників без втручання в прошивку обладнання.

Існуючі сигнатурні методи виявляються неефективними проти нових або модифікованих атак, тоді як класичні алгоритми машинного навчання потребують збору величезних розмічених наборів даних для тренування. Крім того, популярні хмарні системи інтелектуального аналізу створюють неприпустимі ризики витоку конфіденційного трафіку за межі локального периметра.

Розв'язання наукової задачі полягає у використанні локальних великих мовних моделей як автономних агентів безпеки. Такий підхід дозволяє замінити жорсткі алгоритми на гнучкий семантичний аналіз контексту та забезпечити прозорість прийняття рішень без необхідності попереднього донавчання нейромережі. Це обґрунтовує доцільність розробки системи, що поєднує приватність локальної обробки даних з адаптивністю генеративного штучного інтелекту.

Метою кваліфікаційної роботи є розроблення методу та програмних засобів для автоматизованого виявлення кібератак в IoT-середовищах шляхом інтелектуального аналізу мережевого трафіку, що забезпечить підвищення рівня захищеності інформаційно-комунікаційних систем.

Завдання кваліфікаційної роботи:

- проаналізувати особливості мережевого трафіку в IoT-системах та існуючі підходи до застосування генеративного штучного інтелекту в кібербезпеці;
- розробити метод інтелектуального аналізу пакетних даних на основі великої мовної моделі із застосуванням технік конструювання системних запитів;
- спроектувати архітектуру програмного агента, що використовує протокол контексту моделі для взаємодії з інструментами моніторингу та блокування загроз;

– реалізувати програмний засіб, що об'єднує модуль захоплення трафіку та локальну нейромережу, та перевірити коректність прийняття рішень системою в реальному часі.

Об'єкт дослідження: процес автоматизованого реагування на кіберзагрози в комп'ютерних мережах.

Предмет дослідження: методи застосування локальних великих мовних моделей та агентних підходів для аналізу мережевої активності та управління безпекою.

Методи дослідження: У роботі використано методи системного аналізу для проектування архітектури взаємодії компонентів. Для отримання первинних даних застосовано методи перехоплення та розбору мережевих пакетів. Основним інструментом аналізу є метод використання попередньо навчених великих мовних моделей у режимі логічного виведення без додаткового донавчання. Для організації прийняття рішень використано парадигму агентних систем та конструювання промптів. Взаємодія між інтелектуальним ядром та виконавчим середовищем реалізована через стандартизовані протоколи обміну даними.

Наукова новизна дослідження. Вперше запропоновано метод виявлення та реагування на атаки в IoT-середовищі, що базується на використанні локальної LLM загального призначення як автономного агента безпеки. На відміну від існуючих рішень, які потребують збору великих датасетів та тривалого навчання класифікаторів, запропонований підхід використовує семантичний аналіз метаданих трафіку та контекстні інструкції. Це дозволяє гнучко адаптувати систему до нових типів загроз лише шляхом зміни системного промпту, без необхідності перенавчання моделі.

Практичне значення результатів кваліфікаційної роботи. Розроблено програмний комплекс, який функціонує автономно в локальному середовищі, що забезпечує конфіденційність даних. Система самостійно виконує різноманітні дії, зокрема геолокацію IP-адрес, блокування злоумисників та оцінку навантаження. Запропоноване рішення знижує навантаження на адміністратора мережі шляхом

автоматизації рутинних операцій з реагування на інциденти та надає пояснення прийнятих рішень у форматі природної мови.

Публікації. Результати дослідження апробовано на XIII науково-технічній конференції Тернопільського національного технічного університету імені Івана Пулюя «Інформаційні моделі, системи та технології», VI міжнародній науково – практичній конференції «Цифрова економіка як фактор інновацій та сталого розвитку суспільства», у вигляді тез конференцій.

Структура роботи. Робота складається з пояснювальної записки та графічної частини. Пояснювальна записка складається із вступу, 4 розділів, висновків, списку використаних джерел та додатків.

РОЗДІЛ 1

АНАЛІТИЧНА ЧАСТИНА

1.1. Аналіз сучасного стану безпеки в IoT-середовищах

Інтеграція концепції Інтернету речей у критичну інфраструктуру, промисловість та побутовий сектор трансформує архітектуру глобальних мереж, стираючи межі між фізичним та цифровим світами. Цей процес не лише оптимізує технологічні операції, але й суттєво змінює ландшафт кіберзагроз, зумовлюючи експоненціальне розширення поверхні атак. Гетерогенність пристроїв та відсутність уніфікованих стандартів безпеки перетворюють IoT-сегмент на найслабшу ланку в ланцюзі захисту інформаційних систем.

Динаміка зростання ринку IoT демонструє стійку тенденцію до масштабування. За даними аналітичних звітів IoT Analytics, кількість активних точок підключення станом на кінець 2024 року оцінювалася у 18,8 мільярдів одиниць. На рисунку 1.1 відображено прогноз, згідно з яким кількість пристроїв сягне 21,5 мільярда вже у 2025 році [1]. Ця щільність підключень створює безпрецедентне навантаження на механізми моніторингу мережі.

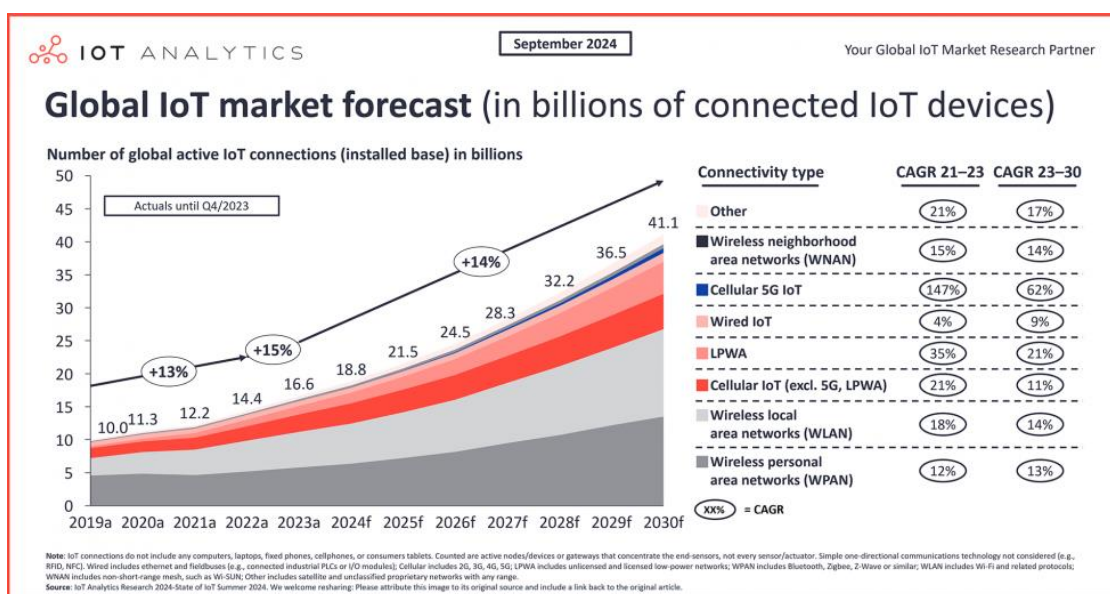


Рис. 1.1. Динаміка зростання кількості підключених IoT-пристроїв (млрд од.) [1]

Ключовою проблемою забезпечення безпеки в таких екосистемах є архітектурна та протокольна фрагментація. В межах однієї мережі часто функціонують сотні пристроїв від різних виробників, що базуються на відмінних апаратних платформах та використовують специфічні комунікаційні протоколи. До переліку найбільш поширених стандартів входять MQTT, CoAP, ZigBee, LoRaWAN та Z-Wave. Більшість із них розроблялися з пріоритетом на енергоефективність та швидкість передачі даних, а не на конфіденційність чи цілісність інформації. Як наслідок, значна частина протоколів не має вбудованих механізмів наскрізного шифрування за замовчуванням. Дослідження мережевого трафіку підтверджують, що близько 98% даних в IoT-мережах передаються у відкритому вигляді, що дозволяє зловмисникам безперешкодно здійснювати атаки типу Man-in-the-Middle та перехоплювати чутливу інформацію [2].

Специфіка апаратного забезпечення IoT-вузлів накладає жорсткі обмеження на вибір засобів захисту. Типовий кінцевий пристрій характеризується низькою продуктивністю процесора, малим обсягом оперативної пам'яті та критичною залежністю від енергоспоживання. Це унеможливорює імплементацію важких криптографічних алгоритмів наприклад, повноцінної реалізації AES з довгим ключем або розгортання агентів антивірусного захисту безпосередньо на пристрої.

Аналіз векторів проникнення, що базується на методології Open Web Application Security Project, виділяє критичні недоліки, притаманні більшості IoT-рішень:

- використання слабких паролів та жорстко закодованих облікових даних адміністратора, які користувач не може змінити;
- проблема відсутності безпечного оновлення, коли механізми оновлення прошивок «по повітрю» не перевіряють цифровий підпис файлу та дозволяють завантаження шкідливого коду;
- наявність незахищених мережесхем інтерфейсів із відкритими портами налагодження, що залишаються активними у продуктивних версіях пристроїв [3].

Детальний розподіл векторів атак відповідно до ієрархічної моделі IoT-систем представлено в таблиці 1.1.

Класифікація загроз безпеці на різних рівнях архітектури IoT

Рівень архітектури	Типи атак	Характеристика
Рівень сприйняття	Фізичне втручання	Пошкодження пристрою або підключення до портів налагодження
	Атаки побічними каналами	Аналіз електромагнітного випромінювання або енергоспоживання пристрою
	Виснаження ресурсів	Примусове виведення пристрою з режиму сну постійними запитами
	Підміна вузла	Введення в мережу зловмисного пристрою, який імітує легітимний сенсор
Мережевий рівень	Man-in-the-Middle	Перехоплення та модифікація трафіку між IoT-пристроєм та шлюзом
	Відмова в обслуговуванні	Перевантаження пропускної здатності каналу або ресурсів шлюзу масовими запитами
	Атаки на маршрутизацію	Маніпуляції з маршрутами передачі даних
	Sybil Attack	Створення одним зловмисним вузлом безлічі фальшивих ідентичностей
Прикладний рівень	SQL-ін'єкції	Впровадження шкідливого SQL-коду через поля вводу веб-інтерфейсу або API
	Міжсайтовий скриптинг	Виконання шкідливих скриптів у браузері
	Вразливості API	Використання незахищених кінцевих точок API
	Шкідливе ПЗ	Зараження керуючих серверів

Статистичні дані інцидентів інформаційної безпеки за 2024–2025 роки фіксують зміщення фокусу кіберзлочинців у бік автоматизованих атак на розумні пристрої. Звіт компанії SonicWall свідчить про зростання кількості атак із використанням спеціалізованого шкідливого програмного забезпечення на 107% у першій половині 2024 року порівняно з попереднім періодом. Найбільш розповсюдженим сценарієм залишається компрометація пристроїв для формування ботнетів. Сімейства шкідливого ПЗ, такі як Mirai, Mozi та їхні численні модифікації, використовують вразливі IoT-вузли для організації масштабних DDoS-атак. Характерною особливістю є зростання персистентності атак, середній час перебування пристрою під ворожим впливом досяг 52,8 годин на тиждень [4].

Окрему загрозу становить конвергенція IT та OT мереж. У промисловому секторі успішна атака на IoT-датчик або контролер може призвести не лише до витоку даних, але й до фізичного пошкодження обладнання чи зупинки виробничих процесів. Середня вартість ліквідації наслідків інциденту в секторі IIoT оцінюється у понад 5 мільйонів доларів [2], що включає витрати на відновлення систем, юридичні штрафи та репутаційні втрати.

Традиційні засоби захисту периметра, такі як міжмережеві екрани та системи запобігання вторгненням, виявляються неефективними в умовах IoT через неможливість інспекції специфічних протоколів та шифрованого трафіку без значних затримок. Статичні правила та сигнатурний аналіз не здатні виявляти атаки нульового дня або складні цільові атаки.

Таким чином, аналіз сучасного стану безпеки вказує на неможливість забезпечення захисту виключно методами посилення захищеності самих кінцевих пристроїв через їх ресурсні обмеження. Це актуалізує науково-прикладну задачу розробки зовнішніх інтелектуальних систем моніторингу. Перспективним напрямком є впровадження методів машинного навчання для аналізу мережевої телеметрії, що дозволить виявляти аномальні поведінкові патерни в режимі реального часу, незалежно від типу кінцевого обладнання чи використовуваного протоколу передачі даних.

1.2. Класифікація атак на інфраструктуру Інтернету речей

Архітектурна складність та високий ступінь гетерогенності мереж Інтернету речей формують унікальний ландшафт загроз, що суттєво відрізняється від традиційних корпоративних мереж, де периметр безпеки є більш чітко окресленим. Різноманіття протоколів передачі даних та апаратних платформ ускладнює впровадження уніфікованих стандартів захисту, роблячи систему вразливою в цілому. Для систематизації векторів атак доцільно використовувати трирівневу архітектурну модель IoT, яка включає рівень сприйняття, мережевий рівень та прикладний рівень. Такий поділ дозволяє точніше класифікувати ризики та розробляти спеціалізовані контрзаходи для кожного шару окремо. Кожен із цих рівнів має специфічні вразливості, що вимагає диференційованого підходу до виявлення інцидентів та реагування на них.

Атаки на рівні сприйняття спрямовані на фізичні сенсори, актуатори та мікроконтролери, де домінуючою загрозою залишається компрометація пристроїв через експлуатацію вразливостей вбудованого програмного забезпечення та використання слабких механізмів автентифікації. Ситуація ускладнюється тим, що виробники дешевих IoT-пристроїв часто не передбачають можливості безпечного віддаленого оновлення прошивки, залишаючи вразливості в безпеці відкритими роками. Крім того, через обмежені обчислювальні потужності кінцевих вузлів, на них часто неможливо реалізувати надійні криптографічні алгоритми, що робить їх легкою здобиччю для зловмисників.

Особливу небезпеку становлять атаки виснаження ресурсів, відомі як Sleep Deprivation Attacks або атаки на виснаження батареї. Оскільки значна частина IoT-вузлів є автономними та живляться від елементів живлення обмеженої ємності, їх алгоритми роботи передбачають тривалі періоди сну, тобто режим низького енергоспоживання та короткочасну активацію для передачі телеметрії. Порушення цього циклу є критичним для життєздатності всієї сенсорної мережі. Зловмисник, надсилаючи безперервний потік запитів або спеціально сформованих пакетів, змушує радіомодуль пристрою залишатися активним, що призводить до повного розряду

батареї за лічені години замість розрахованих років експлуатації, фактично виводячи вузол з ладу без фізичного втручання.

Також на цьому рівні відбувається первинне зараження пристроїв для формування ботнетів, які згодом використовуються для масованих кібератак. Класичним прикладом залишається шкідливе програмне забезпечення сімейства Mirai, яке продемонструвало вразливість екосистеми IoT перед автоматизованими скриптами. Алгоритм його роботи базується на скануванні простору IP-адрес для виявлення пристроїв із відкритими портами Telnet/SSH та спробі авторизації за словником заводських паролів. Сучасні модифікації цього вірусу навчилися адаптуватися до різних архітектур процесорів, що значно розширює потенційну площу атаки. Згідно зі звітом Nozomi Networks, варіанти Mirai продовжують утримувати першість серед шкідливого ПЗ для IoT та відповідальні за найбільші піки активності ботнетів, зафіксовані у першій половині 2025 року [5].

На мережевому рівні, який є критичним для забезпечення комунікації між вузлами та шлюзом, однією з найбільш деструктивних загроз є розподілені атаки типу відмова в обслуговуванні. Особливість таких атак у середовищі IoT полягає в їхній асиметрії, коли величезна кількість скомпрометованих пристроїв із низькою пропускнуою здатністю створює сумарний трафік, що перевантажує магістральні канали або сервери. Виявити та відфільтрувати такий трафік вкрай складно, оскільки, запити надходять від легітимних IP-адрес і можуть імітувати нормальну поведінку пристроїв. Крім прямого блокування сервісів, така активність призводить до виснаження батареї, оскільки легітимні пристрої змушені постійно обробляти вхідні запити або намагатися повторно передати пакети через зашумлений ефір, що викликає ланцюгову реакцію відмов у мережі.

Крім об'ємних атак, значну загрозу становлять атаки Man-in-the-Middle, які порушують конфіденційність та цілісність переданої інформації. Протоколи прикладного рівня, такі як MQTT або CoAP, часто налаштовуються без використання TLS-шифрування для економії ресурсів мікроконтролера та зменшення затримок при передачі даних. Це дозволяє зловмисникам, що отримали доступ до локальної мережі, перехоплювати пакети даних, модифікувати команди керування та фальсифікувати

показники сенсорів. В умовах промислового інтернету речей підміна даних телеметрії може призвести до прийняття хибних автоматизованих рішень системою керування, що загрожує не лише збоями в технологічному процесі, а й фізичними аваріями.

Окремий клас загроз стосується специфічних протоколів маршрутизації для мереж із низьким енергоспоживанням, зокрема «Routing Protocol for Low-Power and Lossy Networks». У таких сітчастих мережах поширені атаки на топологію:

- Blackhole Attack характеризується тим, що після перехоплення трафіку вузол просто відкидає всі пакети, не пересилаючи їх далі, що призводить до ізоляції сегмента мережі;
- Selective Forwarding полягає у вибіркового відкиданні зловмисником пакетів певного типу або від конкретних вузлів, що ускладнює виявлення атаки засобами моніторингу доступності;
- RPL Sinkhole Attack – це атака, під час якої шкідливий вузол оголошує себе найкращим маршрутом до базової станції та притягує до себе весь трафік сусідніх вузлів, як це показано на рисунку 1.2.

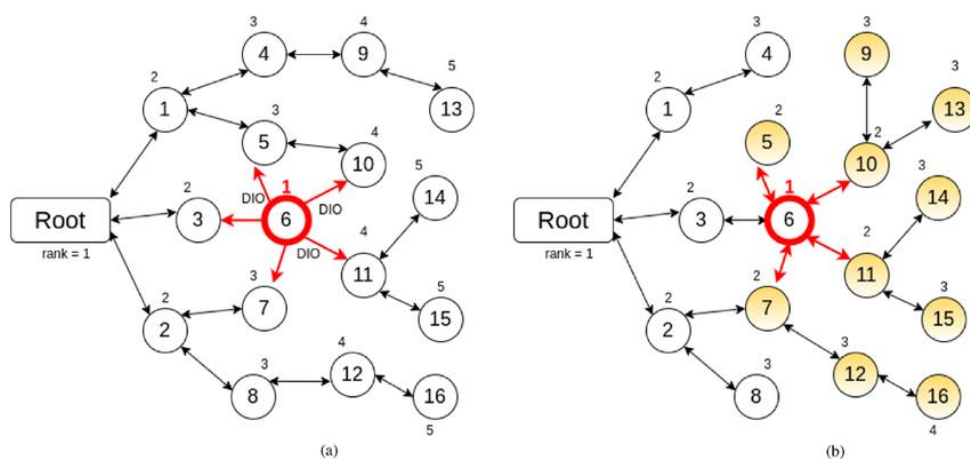


Рис. 1.2. Схема реалізації атаки Sinkhole у мережі, скомпрометований вузол 6 надсилає хибний ранг через повідомлення DIO (a), сценарій мережі після успішної атаки Sinkhole (b) [6]

На рівні прикладних середовищ специфіка векторів атак безпосередньо залежить від сфери впровадження IoT-системи, яка диктує пріоритетність цілей для зловмисників. У сегменті «розумний будинок» основною метою атак є порушення

приватності користувачів. Несанкціонований доступ до IP-камер, смарт-замків або голосових асистентів дозволяє здійснювати приховане спостереження, збір персональних даних та створює загрози фізичній безпеці мешканців. Тут атаки часто спрямовані на порушення конфіденційності.

Натомість у сфері промислового інтернету речей пріоритетом є порушення технологічних процесів, що зачіпає аспекти доступності та цілісності. Атаки на програмовані логічні контролери або SCADA-системи, інтегровані з IoT-сенсорами, можуть спричинити зупинку виробничих ліній, пошкодження дорогого обладнання або призвести до техногенних катастроф. У таких системах затримка передачі пакету або підміна керуючої команди мають кінетичний ефект, тобто призводять до реальних фізичних наслідків.

Отже, аналіз класифікації атак демонструє, що традиційні методи захисту є недостатніми для IoT. Велика кількість загроз реалізується на рівні маніпуляцій з трафіком, що підтверджує доцільність використання методів глибокого аналізу пакетів та поведінкового аналізу мережевої взаємодії для виявлення аномалій.

1.3. Огляд існуючих методів та засобів виявлення вторгнень

Забезпечення кібербезпеки в Інтернеті речей вимагає застосування надійних механізмів моніторингу, що можуть ефективно працювати навіть за умов суттєво обмежених обчислювальних ресурсів та значної різноманітності мережевих протоколів. Системи виявлення вторгнень, які інтегруються в таку інфраструктуру, зазвичай класифікують за стратегією аналізу трафіку на категорії, такі як сигнатурні методи та методи виявлення аномалій. Кожен із цих підходів спирається на специфічні алгоритми обробки даних, що безпосередньо впливає на їхню кінцеву ефективність або ж створює певні обмеження у динамічному середовищі IoT-систем.

Функціонування сигнатурних методів базується на чіткому співставленні перехоплених мережевих пакетів із записами у базі даних відомих патернів атак, що називаються сигнатурами. Такий підхід успішно реалізовано у популярних інструментах захисту, таких як Snort та Suricata, які широко використовуються для

моніторингу мереж. Головною перевагою цього класу систем є висока точність ідентифікації вже відомих загроз, зафіксованих у базах вразливостей на кшталт CVE, при збереженні мінімального рівня помилкових спрацювань під час аналізу.

Процес детектування здійснюється через глибоку інспекцію пакетів, під час якої перевіряється не лише заголовок, а й повний вміст пакету на наявність шкідливих послідовностей байтів. Це дозволяє системі точно розпізнавати спроби атак, які відповідають заздалегідь визначеним шаблонам, і блокувати їх до моменту завдання шкоди. Схема роботи сигнатурної IDS, розгорнутої у шлюзі IoT-мережі, продемонстрована на рисунку 1.3.

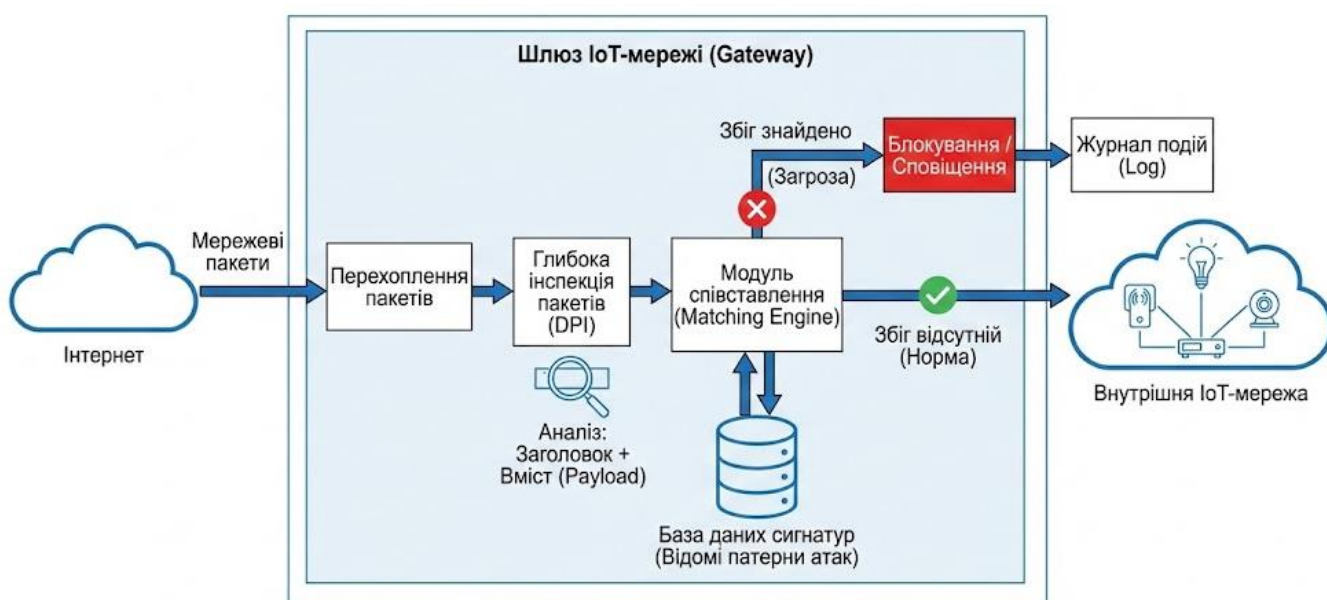


Рис. 1.3. Схема роботи сигнатурної IDS

Імплементация сигнатурного підходу в екосистему IoT стикається з низкою критичних перешкод, головною з яких є ресурсні обмеження. Ефективність цього методу прямо залежить від актуальності бази сигнатур, але регулярне оновлення та зберігання таких об'ємних баз даних створює надмірне навантаження на дискову підсистему та оперативну пам'ять периферійних шлюзів, які часто мають лімітовані апаратні можливості. Крім того, виникає проблема, пов'язана зі специфікою протоколів. Стандартні набори правил популярних IDS орієнтовані переважно на стек TCP/IP. Водночас IoT-середовища широко використовують пропрієтарні або

«легковагові» протоколи, як-от ZigBee, LoRaWAN чи MQTT. Специфічні структури даних цих протоколів часто залишаються поза увагою стандартних парсерів Snort, що, по суті, створює «сліпі зони» для системи безпеки. Зрештою, сигнатурний метод має фундаментальний недолік, а саме нездатність виявляти нові загрози, оскільки він є реактивним, і система може виявити лише ту атаку, яка вже була проаналізована експертами та додана до бази. Згідно з дослідженнями, такі системи пропускають до 40% атак, спрямованих на специфічні вразливості IoT-пристроїв, через відсутність релевантних правил у конфігураціях за замовчуванням [7].

Аномальні методи, спрямовані на побудову профілю нормальної поведінки мережі з подальшим виявленням будь-яких статистично значущих відхилень. Для реалізації цього підходу широко застосовуються алгоритми класичного машинного навчання. Серед найбільш вживаних методів виділяють метод опорних векторів, випадковий ліс та алгоритми кластеризації, такі як K-Means.

Принцип дії полягає у навчанні моделі на «чистому» трафіку, після чого система здатна класифікувати поточні події як «нормальні» або «аномальні». Ключовою перевагою є теоретична здатність детектувати атаки нульового дня, це загрози, для яких ще не існує сигнатур.

Попри потенціал, застосування класичного ML в IoT супроводжується проблемою дрейфу концепції. Легітимна поведінка IoT-пристроїв не є статичною. Оновлення прошивки, зміна режиму роботи сенсорів або сезонні коливання активності можуть суттєво змінювати патерни трафіку. Статичні ML-моделі часто інтерпретують такі легітимні зміни як аномалії, що призводить до високого рівня помилкових тривог [8].

Аналіз існуючих рішень демонструє, що традиційні підходи мають критичні вразливості перед ландшафтом сучасних кіберзагроз. Сигнатурні методи залишаються повністю неефективними проти атак нульового дня та поліморфного шкідливого програмного забезпечення. Водночас, класичні методи машинного навчання, хоча й фіксують аномалії, часто не здатні забезпечити коректну семантичну інтерпретацію події. Вони схильні плутати технічні збої мережі з цілеспрямованими атаками через відсутність розуміння контексту.

Окремою проблемою, що потребує вирішення, є високий рівень хибних спрацювань у ML-системах, який у реальних умовах експлуатації IoT може сягати 15–20% [9]. Постійний потік помилкових повідомлень викликає у системних адміністраторів явище втоми від тривоги, що призводить до зниження пильності та ризику ігнорування реальних інцидентів безпеки.

Крім того, класичні ML-алгоритми часто діють як «чорна скринька»: вони видають бінарний результат класифікації без пояснення причин прийняття рішення. Відсутність пояснюваності ускладнює процес розслідування інцидентів та реагування на них. Система не може надати аргументовану відповідь, чому саме пікове навантаження було класифіковано як DDoS-атака, а не як легітимний сплеск активності користувачів.

Зазначені обмеження зумовлюють необхідність зміни парадигми захисту шляхом переходу від жорстких алгоритмів та статистичних моделей до використання інтелектуальних агентів на базі великих мовних моделей. Використання LLM-агентів дозволяє аналізувати не лише числові параметри трафіку, але й контекст подій, адаптуватися до змін інфраструктури без необхідності повного перенавчання моделі, а також генерувати деталізовані, зрозумілі людині звіти про інциденти.

1.4. Концепція AI-агентів та LLM у кібербезпеці

Еволюція генеративного штучного інтелекту зумовила перехід від пасивного аналізу даних до створення автономних агентних систем. Центральним елементом таких систем є великі мовні моделі, які виступають когнітивним ядром, здатним інтерпретувати семантику неструктурованих даних, системних логів, тіл HTTP-запитів чи конфігураційних файлів. На відміну від сигнатурних методів, LLM оперують контекстом, що дозволяє виявляти складні атаки нульового дня на основі опису аномальної поведінки, а не жорстких правил [10].

Проте сама по собі мовна модель обмежена текстовим вводом-виводом і не може безпосередньо взаємодіяти з мережевою інфраструктурою. Для перетворення

LLM на активний елемент системи захисту застосовується концепція Tool Use та архітектура AI-агентів.

Сутність агентного підходу полягає у здатності моделі визначати необхідність виклику зовнішніх функцій для вирішення поставленої задачі. Це реалізується через механізм Function Calling або Tool Use. Модель не виконує код самостійно, а генерує структурований запит у форматі JSON, що містить назву інструменту та аргументи. Середовище виконання перехоплює цей запит, виконує відповідну функцію і повертає результат моделі для подальшого аналізу.

Процес прийняття рішень агентом формалізується через цикл, відомий як ReAct, приклад якого роботи зображено на рисунку 1.4.. Він включає ітеративні кроки, такі як спостереження, формування думки, вибір інструменту, виконання дії та оцінка результату.

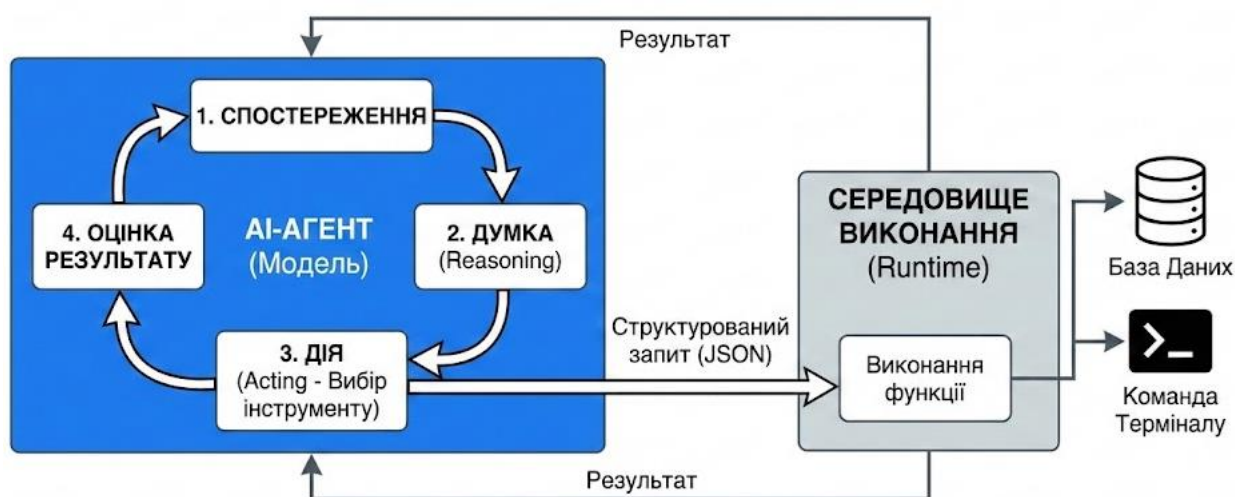


Рис. 1.4. Приклад роботи AI-агента за методологією ReAct

Для уніфікації взаємодії між LLM та зовнішніми інструментами дедалі більшого поширення набуває Model Context Protocol. Це відкритий стандарт, який вирішує проблему фрагментації інтеграцій "точка-точка". Замість написання окремого конектора для кожного інструменту, MCP пропонує клієнт-серверну архітектуру.

Використання MCP у проєктованій системі дозволяє динамічно підключати нові джерела даних про трафік без необхідності перенавчання моделі або зміни

основного коду агента. Модель отримує контекст про доступні інструменти при ініціалізації сесії та використовує їх за потреби [11].

Критичним аспектом впровадження LLM в IoT є архітектура розгортання, оскільки використання хмарних API створює ризики для безпеки периметра. Передача мережевого трафіку на зовнішні сервери суперечить принципам OPSEC через загрозу витоку чутливих даних та топології мережі. Додатково, враховуючи специфіку роботи IoT в ізольованих сегментах, залежність від інтернет-з'єднання формує єдину точку відмови, що може паралізувати систему захисту при втраті зв'язку.

Рішенням є використання локальних LLM, що розгорнуті безпосередньо на периферійному обладнанні або локальному сервері.

Отже, інтеграція агентного підходу на базі протоколу MCP та використання локальних моделей формують технологічний базис для створення адаптивної, приватної та відмовостійкої системи захисту IoT.

1.5. Висновки до розділу 1

У даному розділі здійснено системний аналіз проблематики захисту інфраструктури Інтернету речей та обґрунтовано необхідність зміни парадигми виявлення кіберзагроз. Встановлено, що ключовими факторами вразливості сучасних IoT-середовищ є гетерогенність протоколів передачі даних, обмежені апаратні ресурси кінцевих пристроїв та відсутність уніфікованих стандартів безпеки. Це робить інфраструктуру вразливою до складних векторів атак, які важко ідентифікувати класичними методами.

Критичний огляд наявних інструментів показав обмеженість сигнатурних підходів, які є реактивними за своєю суттю і неефективними проти атак нульового дня. Водночас застосування класичних алгоритмів машинного навчання, попри їхню здатність до виявлення аномалій, ускладнюється високим рівнем хибних спрацювань та проблемою інтерпретованості результатів що є критичним для прийняття рішень операторами безпеки.

В якості перспективного напрямку розв'язання окреслених проблем визначено використання автономних AI-агентів на базі великих мовних моделей. На відміну від статичних аналізаторів, такі системи здатні до семантичного розуміння контексту мережових подій. Обґрунтовано доцільність впровадження механізму Tool Use та використання протоколу MCR, що дозволяє моделі не просто класифікувати трафік, а й активно взаємодіяти з інструментами діагностики та захисту.

Окрему увагу приділено архітектурі розгортання системи. Доведено, що з міркувань безпеки операцій та відмовостійкості, обробка даних повинна відбуватися на периферії без передачі чутливого трафіку до хмарних сервісів. Використання локальних LLM визначено як оптимальний підхід, що забезпечує баланс між приватністю даних, автономністю роботи при втраті інтернет-з'єднання та швидкістю системи.

Таким чином, результати аналізу формують теоретичне підґрунтя для наступних етапів роботи, спрямованих на розробку програмного забезпечення для виявлення та реагування на атаки в IoT-середовищах.

РОЗДІЛ 2

ТЕОРЕТИЧНА ЧАСТИНА

2.1. Методи попередньої обробки та серіалізації мережевого трафіку

Ключовим фактором успішного застосування великих мовних моделей для аналізу кіберзагроз виступає якість вхідних даних. Мережевий трафік у сирому вигляді являє собою потік бінарних даних, який є незрозумілим для тексто-орієнтованих нейромереж без попередньої трансляції. Тому першим етапом роботи системи є трансформація перехоплених пакетів у структурований текстовий формат зі збереженням критично важливих метаданих. Цей процес включає вибір ознак, серіалізацію та оптимізацію обсягу контексту.

Для програмної обробки пакетів у розробленій системі застосовано інструментарій бібліотеки Scapy. Як показано на рисунку 2.1, вона дозволяє розділяти мережевий кадр на окремі шари моделі OSI. Це дає змогу звертатися до конкретних полів заголовків як до атрибутів об'єкта.

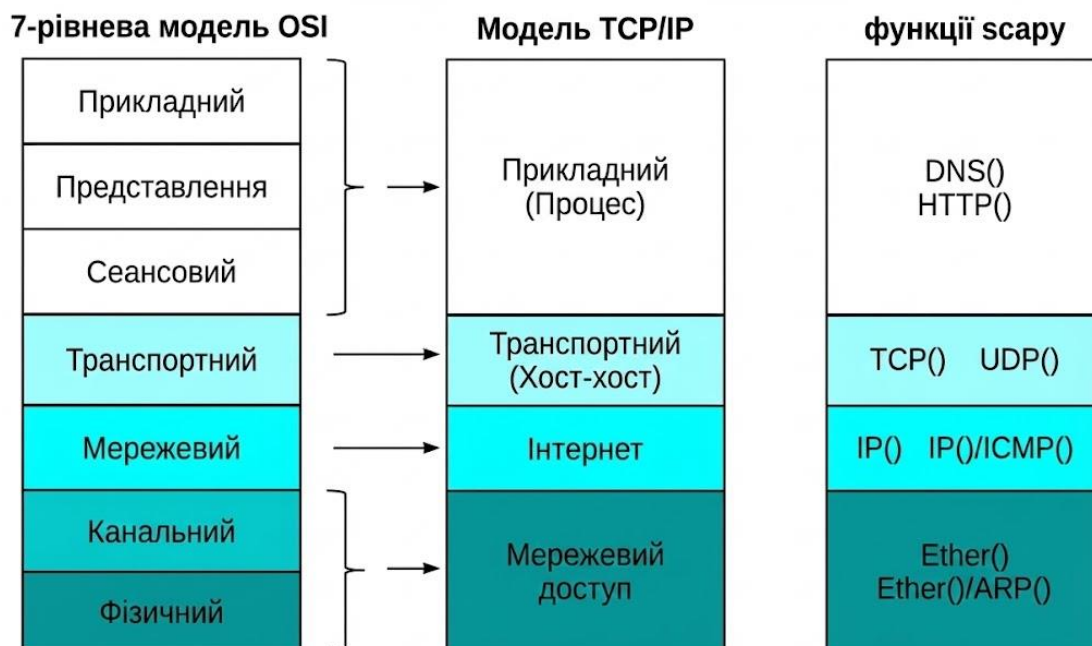


Рис. 2.1. Схема розділення мережевого пакету бібліотекою Scapy

В контексті IoT-середовищ, де трафік часто є специфічним та обмеженим за пропускнуою здатністю, повний дамп пакету часто містить надмірну інформацію. Для виявлення аномалій, таких як DDoS-атаки або сканування портів, достатньо аналізувати лише ключові заголовки протоколів мережевого та транспортного рівнів.

Процедура розділення фокусується на отриманні наступних параметрів:

- Ethernet-рівень використовує MAC-адреси джерела та призначення, що дозволяє точно ідентифікувати фізичні пристрої в локальному сегменті мережі;
- IP-рівень містить IP-адреси, версію протоколу та час життя пакету, аномальні значення якого можуть свідчити про спроби приховування топології мережі атакувальником [12];
- транспортний рівень оперує портами, номерами послідовності та прапорами TCP-сесії, аналіз комбінацій яких є критичним для виявлення атак типу SYN-flood;
- payload – це безпосередній вміст пакету, що передається для обробки на прикладному рівні.

Після виділення необхідних атрибутів відбувається процес серіалізації. Сирі значення полів перетворюються у формат, придатний для інтерпретації LLM. Найбільш доцільним форматом для цієї задачі обрано JSON. Його ієрархічна структура «ключ-значення» дозволяє моделі чітко розрізняти семантику даних. Модель не просто бачить набір цифр, а розуміє, що число «80» відноситься до ключа «destination_port».

Процес перетворення окремого пакета на JSON-об'єкт можна представити як відображення множини обраних ознак, що обчислюється за формулою (2.1).

$$J = \text{Serialize}(\{f_i \in P \mid f_i \in F_{\text{selected}}\}), \quad (2.1)$$

де J – JSON-об'єкт;

P – пакет даних;

F – множина обраних ознак;

F_{selected} – заздалегідь визначена підмножина значущих полів пакету.

Такий підхід забезпечує уніфікацію вхідних даних незалежно від протоколу нижнього рівня [13].

Однією з головних технічних проблем при інтеграції LLM є обмеження контекстного вікна. Передача повного вмісту кожного пакету швидко вичерпує ліміт токенів моделі. Це призводить до зростання затримок та вартості обробки запитів.

Для вирішення цієї проблеми розроблено стратегію оптимізації контексту. Вона базується на селективному відсіюванні інформації, яка не несе цінності для виявлення сигнатур атак.

Для оптимізації обробки даних застосовується усічення корисного навантаження шляхом збереження лише перших N байтів, що дозволяє виявляти сигнатури експлойтів без передачі зайвих обсягів інформації, оскільки повний вміст часто є вторинним або зашифрованим. Разом із цим виконується фільтрація технічних полів, які мають високу ентропію, але низьку цінність для LLM, зокрема контрольних сум та певних опцій TCP, якщо вони не є безпосереднім предметом аналізу. В таблиці 2.1 наведено порівняльний аналіз порівняння обсягу даних до та після оптимізації контексту.

Таблиця 2.1

Порівняння обсягу даних до та після оптимізації контексту

Компонент пакета	Стандартний підхід	Оптимізований підхід	Коефіцієнт стиснення
Корисне навантаження	Повний вміст	Усічено до перших N байтів	≈95%
Технічні поля	Усі поля заголовкі	Значущі поля IP src/dst, Port, Flags	≈40-60%
Представлення	Raw Hex Dump або Base64	Спрощений JSON	—
Кількість токенів	800-1200 токенів на пакет	50-150 токенів на пакет	В 10-15 разів менше

Такий підхід дозволяє зменшити розмір промπτу до LLM у 3-4 рази, зберігаючи при цьому достатню інформативність для прийняття рішення щодо безпеки вузла [14]. Крім того, ключі в JSON-структурі скорочуються до зрозумілих абревіатур, що додатково економить токени.

2.2. Алгоритмічне забезпечення агентної взаємодії

Результативність системи виявлення вторгнень на основі великих мовних моделей визначається не тільки самою моделлю, а ще й архітектурою її інтеграції у зовнішнє середовище. Прості генерації тексту недостатньо для реагування на кіберзагрози в реальному часі. Тому в основу розробки покладено агентний підхід, який перетворює LLM з пасивного консультанта на активний керуючий елемент системи захисту.

Фундаментом алгоритмічного забезпечення виступає парадигма ReAct. Цей підхід, запропонований Yao et al. [11], об'єднує здатність моделі до логічних міркувань із можливістю виконання дій. У контексті захисту IoT-мережі це дозволяє агенту формувати динамічні плани реагування замість сліпого слідування жорстким правилам.

Процес прийняття рішень реалізується через циклічну послідовність кроків. Траєкторія роботи агента на кожному кроці визначається як послідовність спостережень, думок та дій.

Першим етапом є «спостереження». Агент отримує структуровані дані від сніфера мережевого трафіку. Це можуть бути заголовки пакетів, підозрілі пейлоади (вміст пакетів) або статистика частоти запитів з однієї IP-адреси. На відміну від класичних IDS, які працюють із сирими даними, тут інформація попередньо агрегується у текстовий опис стану системи.

Наступним кроком відбувається «генерація думки». Модель аналізує отримані дані в контексті своєї бази знань про кіберзагрози. Агент формулює гіпотезу щодо природи аномалії. Наприклад, він може визначити, що висока частота SYN-пакетів

свідчить про початок DoS-атаки, а не про легітимне навантаження. Цей етап критично важливий для зменшення кількості хибних спрацювань.

Третім етапом є «дія». Базуючись на сформульованій гіпотезі, агент обирає конкретний інструмент для впливу на середовище або отримання додаткової інформації.

Реалізація здатності агента впливати на мережеве середовище забезпечується через механізм використання інструментів. LLM не має прямого доступу до командної оболонки операційної системи. Натомість використовується проміжний шар, який описує доступні функції у форматі JSON-схеми. Агент отримує інструкцію, яка містить опис доступних інструментів, їхніх аргументів та призначення.

Важливим компонентом архітектури є механізм зворотного зв'язку. Після виконання дії агент отримує результат виконання від середовища. Це не просто підтвердження успішного запуску скрипта, а й нові дані про стан мережі. Якщо агент заблокував IP-адресу, наступне спостереження повинно показати припинення надходження пакетів від цього джерела.

Обробка результатів виконання дозволяє реалізувати механізм самокорекції. Якщо спроба блокування завершилася помилкою наприклад, через брак прав доступу або конфлікт правил, агент отримає повідомлення про помилку як нове спостереження. Це ініціює новий цикл мислення, де модель спробує альтернативний метод захисту або повідомить адміністратора про несправність підсистеми реагування [15].

Така замкнена система забезпечує стійкість роботи алгоритму. Агент не просто класифікує загрози «в один прохід», а веде безперервний діалог із захищеним середовищем, коригуючи свою тактику в залежності від зміни обстановки.

2.3. Розробка системних інструкцій та контекстне управління

Якщо попередні етапи забезпечують підготовку синтаксично коректних даних, а архітектура визначає механізм виконання дій, то завдання системного промпту, це надати моделі семантичну основу для прийняття рішень. У даній роботі розроблено

спеціалізовану інструкцію, яка трансформує універсальну мовну модель у вузькопрофільного експерта з кібербезпеки, мінімізуючи ймовірність недетермінованої поведінки. Зокрема, завдяки чітким маркерам система формує стійкий контекстний вектор, який слугує фільтром для відсіювання невідповідної інформації ще на етапі обробки запиту.

Першим рівнем інструкції є визначення «ролі». Агенту задається директива діяти як «автономний вартовий мережевої безпеки», що проілюстровано на рисунку 2.2. Це налаштування є критичним, оскільки воно звужує простір вірогідних продовжень тексту до термінології та логіки захисту інформації. Таким чином, роль виступає не просто стилістичним оформленням, а функціональним орієнтиром, що утримує увагу моделі на специфічних патернах атак та методах їх нейтралізації.

```
You are an autonomous network security sentinel. Your goal is to protect the localhost while ensuring legitimate traffic is NOT blocked.
```

Рис. 2.2. Фрагмент промπτу для встановлення ролі агента

Важливим елементом промπτу є блок семантичної інтерпретації вхідних даних. Оскільки модуль попередньої обробки передає «сухі» числа, завданням промπτу є надати їм контекстне значення. Наприклад, як показано на рисунку 2.3, в інструкцію введено окреме правило для обробки часових міток.

```
- "duration_sec": 0.0 is NOT an error. It indicates an instantaneous burst of traffic (< 0.01s).  
- Treat "duration_sec": 0.0 combined with multiple packets as an EXTREME THREAT (Micro-burst flood).
```

Рис. 2.3. Фрагмент промπτу для обробки часових міток

Це дозволяє моделі коректно розпізнавати специфічні типи атак, які на рівні простої логіки могли б бути відкинуті як помилка вимірювання. Схему об'єднання статичних інструкцій та динамічних даних у єдиний контекст наведено на рисунку 2.4.

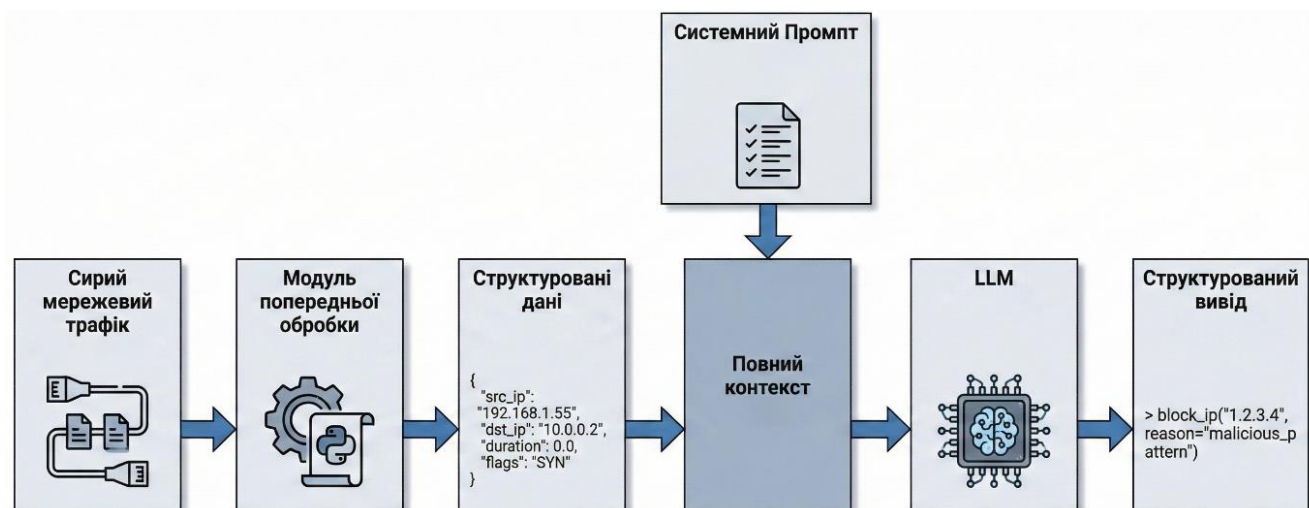


Рис. 2.4. Схема трансформації сирих даних у контекст для LLM

Як видно зі схеми, «повний контекст» є сукупністю системних настанов та JSON-об'єкта конкретного пакету, що дозволяє моделі бачити не лише параметри трафіку, але й правила їх оцінки.

Для реалізації циклу «думка – дія» системний промпт містить чіткий алгоритм пріоритетів. Щоб уникнути колізій, коли один пакет підпадає під кілька критеріїв, інструкція вимагає суворого дотримання порядку перевірки. Алгоритм складається з п'яти послідовних етапів.

Перший етап – це фільтрація легітимного трафіку, фрагмент промпту якої представлений на рисунку 2.5. Ця частина промпту зобов'язує агента спершу перевіряти трафік на ознаки безпечної високої активності.

```

1. SAFE TRAFFIC CHECK (HIGHEST PRIORITY):
- IF avg_packet_size > 800 bytes AND duration > 0.5s (likely Video/Download/FileTransfer):
  -> ACTION: Call `log_safe_traffic`. Do NOT block.
  -> REASON: "Legitimate high bandwidth traffic".
- Note: If packets are large (>800) BUT happen instantly (duration < 0.1s), treat as FLOOD.
  
```

Рис. 2.5. Фрагмент промпту для фільтрації легітимного трафіку

Якщо умова виконується, агент припиняє подальший аналіз потоку. Це інженерне рішення дозволяє зменшити навантаження на підсистему блокування та уникнути переривання легальних бізнес-процесів. До таких процесів можна віднести,

наприклад, планове резервне копіювання баз даних або масове завантаження оновлень програмного забезпечення, які створюють високе, але безпечне навантаження на мережу. Водночас, промпт містить критичне застереження, якщо великі пакети надходять миттєво, це класифікується не як безпечний трафік, а як флуд.

На рисунку 2.6 відображено другий етап, де відсікаються явні DoS/DDoS атаки. На цьому рівні модель діє без вагань, оскільки показники однозначно свідчать про аномалію.

```
2. CRITICAL THREATS (DoS/Flood):
- IF (total_packets > 1000) OR (duration < 1.0s AND total_packets > 10):
-> ACTION: You MUST call `block_ip`.
-> REASON: "High volume flood detected".
```

Рис. 2.6. Фрагмент промпту для фільтрації DoS атак

На рисунку 2.7 відображено логіку, згідно з якою, якщо трафік не підпадає під перші дві категорії), то далі перевіряється його спрямованість.

```
3. SUSPICIOUS ACTIVITY (Scanning):
- IF dst_ports_count > 3 OR probing sensitive ports (23, 22, 3389, 445):
-> ACTION: Call `get_ip_info` to investigate origin.
```

Рис. 2.7. Фрагмент промпту для підозрілого трафіку

Агент ідентифікує ознаки горизонтального сканування понад 3 порти призначення або спроби підключення до чутливих сервісів. Дія на цьому етапі відрізняється від попередніх, замість блокування ініціюється розслідування через виклик `get_ip_info`. Це дозволяє уникнути обмеження доступу для адрес, збираючи додатковий контекст перед прийняттям рішення. Такий підхід дозволяє суттєво знизити ймовірність помилкового блокування, даючи системі час на глибший аналіз репутації джерела.

Четвертий рівень логіки, представлений на рисунку 2.8, використовує здатність LLM до нечіткої логіки для виявлення граничних випадків. Традиційні фаєрволи

часто пропускають атаки, які навмисно підлаштовані під порогові значення, наприклад, 900 пакетів при ліміті 1000. Агент налаштований реагувати на сукупність факторів. Якщо метрики знаходяться трохи нижче порогу тривоги, але атака спрямована на критичні порти, приймається превентивне рішення про блокування з обґрунтуванням. У цьому аспекті LLM демонструє значну перевагу над жорсткими правилами, оскільки здатна оцінювати ризик ситуативно.

```
4. INTELLIGENT OVERSIGHT (For Gray Areas):
- IF metrics are slightly below thresholds (e.g., 800-999 packets) BUT target sensitive ports (22, 3389, 23):
-> ACTION: Call `block_ip`.
-> REASON: "Suspicious borderline activity on sensitive port".
```

Рис. 2.8. Фрагмент промпту для виявлення граничних випадків

На рисунку 2.9 показано, що фінальний етап перевірки активується після отримання даних від інструменту `get_ip_info`. Якщо IP-адреса походить з регіонів високого ризику, наприклад, країн, що відомі кіберагресією і трафік містить ознаки сканування, застосовується безумовне блокування.

```
5. GEO-BLOCKING POLICY:
- IF `get_ip_info` returns a high-risk region (e.g., China, North Korea, Iran)
  AND the traffic involves scanning or sensitive ports:
-> ACTION: Call `block_ip`.
-> REASON: "Scanning activity from high-risk geo-location".
```

Рис. 2.9. Фрагмент промпту для блокування за регіоном

Така багаторівнева структура промпту змушує модель генерувати повний «ланцюжок міркувань», що гарантує обґрунтованість кожного виклику функції захисту [16].

Однією з головних проблем LLM є схильність до вигадкування фактів. У розробленому промпті застосовано техніку «заземлення», що проілюстровано на рисунку 2.10. Моделі прямо заборонено робити припущення, що виходять за межі наданого JSON-об'єкта. Це критично важливо для систем безпеки, адже рішення

агента мають базуватися виключно на реальній телеметрії, а не на галюцинаціях чи загальних знаннях моделі.

```
You will receive a JSON LIST of network flows. Analyze EACH flow in the list independently according to the rules.
...
DECISION LOGIC (FOLLOW IN STRICT ORDER):
...|
OUTPUT FORMAT:
You MUST generate a sequence of Tool Calls.
If a flow requires action (block or log), call the tool.
If a flow is ambiguous or irrelevant, you may skip it.
Process the entire list.
```

Рис. 2.10. Фрагменти промпту з «заземленням»

Для забезпечення стабільності виклику інструментів використано методику навчання на прикладах, яка згідно з сучасними дослідженнями дозволяє значно підвищити точність виконання формальних інструкцій [17]. На рисунку 2.11 зображено цю секцію, що містить пари «вхід – логіка – вивід», які демонструють моделі очікуваний формат.

```
EXAMPLES OF CORRECT BEHAVIOR:
Input: {"src_ip": "1.1.1.1", "total_packets": 5000, "duration": 0.5}
Reasoning: High packets + short duration = Flood.
Tool Call: block_ip("1.1.1.1", "High volume flood detected")

Input: {"src_ip": "8.8.8.8", "dst_ports_count": 12}
Reasoning: Many ports indicate scanning, but verify IP first.
Tool Call: get_ip_info("8.8.8.8")

Input: {"src_ip": "10.0.0.5", "avg_packet_size": 1200, "duration": 5.0}
Reasoning: Large packets over long duration indicate safe transfer.
Tool Call: log_safe_traffic("10.0.0.5", "Legitimate high bandwidth traffic")
```

Рис. 2.11. Фрагменти промпту з навчанням на прикладах

Наприклад, для сценарію флуду наведено приклад з аргументацією «high packets + short duration», що завершується викликом `block_ip`. Це слугує основою для моделі, гарантуючи, що навіть при варіативності вхідних даних, структура вихідної команди залишиться валідною для програмного парсингу.

2.4. Архітектура системи та методика вибору базової моделі

Проектування архітектури системи виявлення загроз базується на принципах модульності та асинхронної обробки даних. Такий підхід забезпечує масштабованість рішення та дозволяє замінювати окремі компоненти без необхідності переробки всієї системи.

Загальна логіка функціонування комплексу реалізована як конвеєр обробки інформації, де вихідні дані одного модуля слугують вхідними даними для наступного. Функціональну схему запропонованого рішення наведено на рисунку 2.12.

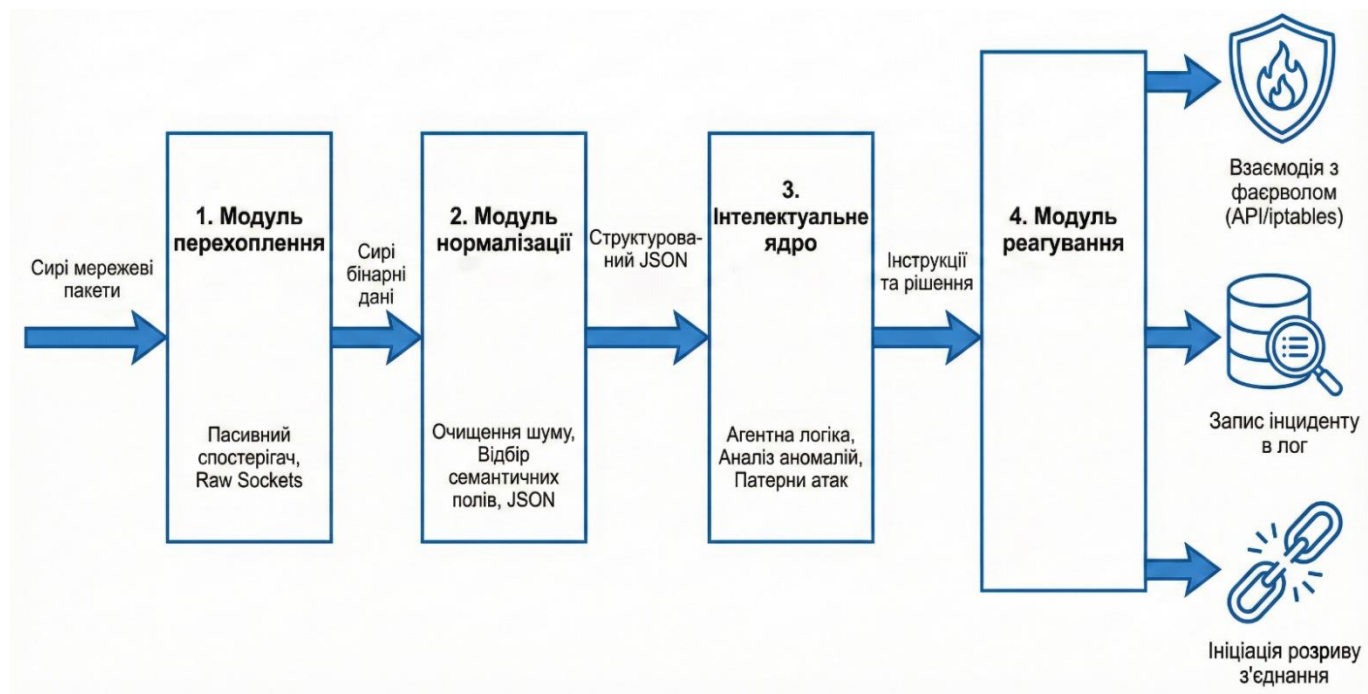


Рис. 2.12. Схема взаємодії компонентів системи

Архітектура системи спирається на взаємодію чотирьох компонентів. Модуль перехоплення діє як пасивний спостерігач, що за допомогою бібліотеки Scapy та Raw Sockets захоплює мережеві пакети в реальному часі, отримуючи доступ до всіх рівнів моделі OSI без створення затримок у мережі [12]. Такий підхід дозволяє здійснювати моніторинг не вносячи змін у топологію чи логіку роботи мережі.

Отримані дані обробляє модуль нормалізації, який трансформує сирій бінарний трафік у чистий JSON-формат, залишаючи лише критично важливі для

безпеки параметри, такі як IP-адреси, порти та прапори TCP. Така попередня обробка дозволяє відсіяти зайвий інформаційний «шум», значно полегшуючи подальший аналіз.

Структуровані дані надходять до інтелектуального ядра, яке реалізує агентну логіку та виявляє аномалії на основі закладених евристик і знань про патерни атак. Залежно від висновків ядра, модуль реагування ініціює відповідні дії. Це може бути як надсилання попередження адміністратору, так і автоматичне блокування шкідливого джерела.

Використання великих мовних моделей у задачах кібербезпеки IoT накладає жорсткі обмеження на обчислювальні ресурси та час реакції. Хмарні API не розглядаються як базове рішення через ризики витоку конфіденційних даних про структуру мережі та неприпустимо високу латентність [18]. Тому архітектура орієнтована на локальне розгортання. Цей підхід гарантує повну автономність захисту, дозволяючи системі працювати навіть за відсутності зовнішнього інтернет-з'єднання. Автономність є критичною для протидії загрозам всередині локального периметра, таким як поширення шкідливого коду із раніше заражених вузлів або атаки при фізичному підключенні злоумисника до внутрішньої мережі. Навіть в ізольованому середовищі система здатна виявити аномальну активність та попередити вихід з ладу критичної інфраструктури.

Вибір оптимальної моделі зводиться до пошуку балансу у просторі критеріїв «точність – швидкість – ресурсоемність». Формалізовано задачу вибору можна представити як формулу (2.2).

$$U(m) = w_1 \cdot R(m) + w_2 \cdot S(m) - w_3 \cdot C(m), \quad (2.2)$$

де $U(m)$ – корисність моделі;

$R(m)$ – здатність моделі до логічного висновку та коректного використання інструментів, що базується на парадигмі ReAct [11];

$S(m)$ – швидкість генерації токенів;

$C(m)$ – споживання VRAM та навантаження на шину даних;

w_1, w_2, w_3 – вагові коефіцієнти, що визначаються пріоритетами системи.

Систематизований перелік ключових вимог до LLM, сформованих на основі обмежень апаратного забезпечення IoT та специфіки задач кібербезпеки, наведено в таблиці 2.2.

Таблиця 2.2

Порівняльна характеристика вимог до LLM для IoT-безпеки

Критерій	Характеристика вимоги	Метрики оцінювання	Значення для IoT-безпеки
Логічна здатність	Здатність моделі аналізувати мережеві пакети, виявляти аномалії та коректно формувати JSON або команди.	Точність виклику функцій, міра F1	Критично важливо для уникнення помилкових спрацювань та пропуску складних атак
Швидкість генерації	Швидкість інференсу, необхідна для реакції на інцидент у реальному часі	Токенів/сек, затримка	Для IoT затримка є неприпустимою. Атака має бути заблокована до нанесення шкоди системі
Ресурсоемність	Вимоги до апаратної частини для локального розгортання агента	Використання VRAM	Модель повинна вміщуватись у доступний бюджет

До переліку кандидатів було включено моделі з відкритими вагами, що підтримують квантування. Для всіх моделей використано метод квантування GGUF формат 4-bit, Q4_K_M, що забезпечує оптимальний компроміс між споживанням VRAM та збереженням когнітивних здібностей [18].

Обґрунтування вибору кандидатів для порівняльного аналізу:

- Hermes 3 8B [19] – ця модель пройшла етап тонкого налаштування спеціально для агентної взаємодії, тому від неї очікується висока стабільність при виконанні системних промптів;
- Llama 3.1 8B [20] вважається індустріальним стандартом у класі компактних моделей, а її ключовою перевагою є нативна підтримка виклику інструментів, що критично важливо для взаємодії з модулем executor;
- Phi-4 Mini 3.8B [21] є найлегшою моделлю у вибірці та була включена до тестування для перевірки гіпотези щодо можливості розгортання системи на мікрокомп'ютерах рівня Raspberry Pi 5;
- Qwen 2.5-Coder 7B [22] являє собою спеціалізовану версію, орієнтовану на генерацію коду. Оскільки протокол Function Calling базується на генерації JSON-структур, припускалося, що «coder» краще впорається з синтаксисом аргументів;
- Qwen 3 8B [23] – це новітня ітерація моделі, що підтримує режим «роздумів», який дозволяє будувати ланцюжок логічних висновків перед прийняттям остаточного рішення.

Остаточний вибір моделі здійснюватиметься на основі емпіричних даних навантажувального тестування.

2.5. Висновки до розділу 2

У даному розділі обґрунтовано комплексний підхід до проектування системи захисту IoT-середовищ, що базується на інтеграції методів глибокого аналізу пакетів та когнітивних можливостей великих мовних моделей. Ключовим результатом теоретичного етапу стала розробка механізму серіалізації бінарного трафіку у структурований JSON-формат. Таке перетворення у поєднанні зі стратегією оптимізації контексту дозволяє адаптувати специфічні мережеві дані для ефективної обробки нейромережею без критичної втрати інформативності.

Встановлено, що для задач активного кіберзахисту статична класифікація поступається динамічному агентному підходу на базі архітектури ReAct. Здатність

моделі формувати ланцюжок міркувань та взаємодіяти із зовнішніми інструментами перетворює систему з пасивного спостерігача на автономний механізм реагування. Для забезпечення стабільності цього процесу спроектовано систему багаторівневих промптів. Вона мінімізує ризики виникнення галюцинацій та забезпечує суворе дотримання логіки виявлення атак шляхом техніки «заземлення» та навчання на прикладах.

Запропонована модульна архітектура орієнтована на локальне розгортання, що є визначальним фактором для забезпечення приватності даних та мінімізації мережевих затримок. Сформульовані математичні критерії оцінювання та визначений перелік моделей-кандидатів створюють необхідне підґрунтя для проведення експериментальної частини роботи. Наступним етапом дослідження є емпірична перевірка ефективності розроблених алгоритмів та порівняльний аналіз обраних моделей в умовах симуляції реальних кібератак.

РОЗДІЛ 3

ПРАКТИЧНА ЧАСТИНА

3.1. Програмна реалізація інтелектуального агента

Реалізація системи виявлення атак вимагала вибору гнучкого та продуктивного інструментарію, здатного об'єднувати низькорівневу роботу з мережею та високорівневі запити до генеративних моделей. Вибір мови програмування Python обумовлений наявністю розвиненої екосистеми бібліотек для задач кібербезпеки та машинного навчання. Архітектура програмного забезпечення побудована за модульним принципом, що забезпечує незалежність підсистем перехоплення трафіку, аналізу даних та виконання захисних дій [12]. Вибір платформи Windows для тестового стенду обумовлений її поширеністю в корпоративних мережах та наявністю вбудованих інструментів управління трафіком, доступних для програмного керування.

Основним інструментом для роботи з мережевим стеком обрано бібліотеку Scapy. Її можливості дозволяють перехоплювати пакети на драйверному рівні, що забезпечує видимість усіх вхідних та вихідних з'єднань.

Інтеграція великої мовної моделі в локальний контур здійснена за допомогою середовища Ollama. Це рішення дозволяє запускати квантовані моделі без необхідності передачі чутливих даних на зовнішні сервери, що є критичною вимогою для безпеки IoT-середовищ. Взаємодія Python-скрипта з моделлю реалізована через REST API інтерфейс, який забезпечує передачу контексту та отримання структурованої відповіді. Для оптимізації швидкодії використано формат моделей GGUF, який дозволяє ефективно використовувати ресурси [24].

Для розгортання розробленого програмного комплексу було застосовано концепцію виділеного міні-сервера, що виконує роль централізованого шлюзу безпеки в локальному сегменті мережі. Апаратна платформа базується на архітектурі x86-64 з процесором рівня Intel Core i5 за умови обов'язкової підтримки набору інструкцій AVX2 та оперативною пам'яттю обсягом від 16 ГБ. Такі характеристики є

необхідним мінімумом для забезпечення швидкого локального інференсу сучасних квантованих мовних моделей та утримання буферів обміну бібліотеки Scapy без використання дискретних графічних прискорювачів. Конфігурація також включає твердотільний накопичувач ємністю 256 ГБ для мінімізації затримок при ініціалізації ваг нейромережі та високошвидкісного запису логів, а мережевий інтерфейс повинен підтримувати режим нерозбірливого захоплення пакетів. В якості програмного середовища обрано операційну систему сімейства Windows Server 2019, що дозволяє інтегрувати рішення в існуючу корпоративну інфраструктуру та забезпечує нативне керування політиками безпеки через вбудований інтерфейс PowerShell без необхідності встановлення сторонніх драйверів фільтрації.

Головною архітектурною особливістю розробленого рішення є використання Model Context Protocol. Такий підхід дозволив відокремити ядро агента, яке відповідає за моніторинг та прийняття рішень, від виконавчого сервера інструментів, що здійснює безпосередню взаємодію з операційною системою Windows.

Система складається з двох основних процесів, що комунікують між собою через стандартний ввід-вивід. Клієнт відповідає за захоплення трафіку, формування пакетів даних та комунікацію з LLM через бібліотеку Ollama. MCP сервер виконує роль прошарку абстракції над системними утилітами. Він надає моделі стандартизований інтерфейс для виконання дій, таких як перевірка правил брандмауера або блокування IP-адрес.

Вибір бібліотеки FastMCP дозволив швидко розгорнути сервер інструментів, а використання PowerShell через модуль subprocess забезпечило нативний контроль над мережевим стеком Windows без необхідності встановлення сторонніх драйверів фільтрації [25].

Головний модуль програми ініціалізує з'єднання з MCP-сервером та запускає нескінченний цикл моніторингу мережі.

Спочатку формується список доступних інструментів через функцію `get_ollama_tools`, яка динамічно конвертує схему MCP у формат JSON Schema, сумісний з API Ollama. Це надає моделі чіткий опис того, які аргументи потрібні для

виконання тієї чи іншої дії, наприклад, що для блокування потрібен валідний IP-рядок, а не доменне ім'я.

Критичним моментом реалізації є керування життєвим циклом сніфера. Для уникнення переповнення буферів екземпляр TrafficSniffer перестворюється на кожній ітерації циклу.

Захоплення пакетів відбувається в окремому потоці за допомогою `asyncio.to_thread`, що дозволяє головному циклу залишатися чутливим до системних переривань. Отриманий масив даних розбивається на менші частини для оптимізації контекстного вікна моделі.

Взаємодія з бібліотекою Ollama відбувається в синхронному режимі. Це означає, що процес захоплення пакетів призупиняється на час генерації відповіді моделлю. Хоча такий підхід може призвести до пропуску частини пакетів під час високого навантаження, в рамках прототипу він забезпечує детермінованість роботи та спрощує відлагодження логіки прийняття рішень.

Фрагмент коду головного циклу програми наведено на рисунку 3.1.

```

async def main():
    # Ініціалізація схеми інструментів через MCP
    tools_schema = await get_ollama_tools()
    # Створення екземпляра агента з отриманими інструментами
    agent = SecurityAgent(tools_schema)
    while True:
        try:
            # Переініціалізація сніфера для оновлення стану сокетів
            sniffer = TrafficSniffer(
                packet_count=settings.PACKET_COUNT,
                timeout=settings.SNIFFER_TIMEOUT
            )
            # Запуск сніфера в окремому потоці
            data = await asyncio.to_thread(sniffer.run_cycle)
            if data:
                # Розбиття потоку даних на менші пакети
                chunk_size = 5
                for i in range(0, len(data), chunk_size):
                    mini_batch = data[i : i + chunk_size]
                    logger.info(f"Analyzing chunk {i}-{i+chunk_size} of {len(data)} flows")
                    # Передача даних агенту для аналізу
                    await agent.process_batch(mini_batch)
        except KeyboardInterrupt:
            logger.info("Shutting down...")
            break
        except Exception as e:
            logger.error(f"Critical error in main loop: {e}", exc_info=True)
            await asyncio.sleep(5)

```

Рис. 3.1. Реалізація головного циклу програми

MCP сервер виступає посередником, який виконує «небезпечні» операції. Однією з найскладніших функцій є перевірка наявності активних правил блокування для конкретної IP-адреси. Це необхідно для уникнення дублювання правил у брандмауері, що могло б призвести до деградації продуктивності системи.

Функція `check_firewall_rules`, представлена на рисунку 3.2, формує динамічний PowerShell-скрипт, який звертається до команди `Get-NetFirewallRule`. Отриманий результат конвертується в JSON безпосередньо в оболонці PowerShell, після чого аналізується Python-скриптом. Це забезпечує надійну передачу структурованих даних між середовищами.

```
@mcp.tool()
def check_firewall_rules(ip_address: str) -> str:
    # Валідація IP-адреси
    try:
        ipaddress.ip_address(ip_address)
    except ValueError:
        return f"Error: '{ip_address}' is not a valid IP address."

    # Формування PowerShell скрипта для пошуку правил
    ps_script = f"""
    $TargetIP = '{ip_address}'
    $Results = @()
    # Отримання всіх активних правил блокування
    $BlockRules = Get-NetFirewallRule -Enabled True -Action Block -ErrorAction SilentlyContinue

    foreach ($Rule in $BlockRules) {{
        $Filter = $Rule | Get-NetFirewallAddressFilter
        if ($Filter.RemoteAddress -contains $TargetIP) {{
            $Results += @{{
                RuleName = $Rule.DisplayName
                Direction = $Rule.Direction
                Profile = $Rule.Profile
            }}
        }}
    }}

    # Вивід результату у форматі JSON
    if ($Results.Count -gt 0) {{ $Results | ConvertTo-Json -Compress }}
    else {{ Write-Output "[]" }}
    """

    # Виконання скрипта через subprocess
    result = subprocess.run(
        ["powershell", "-Command", ps_script],
        capture_output=True, text=True, encoding='utf-8',
        creationflags=subprocess.CREATE_NO_WINDOW
    )

    # ... (код обробки результату та помилок)
```

Рис. 3.2. Реалізація перевірки правил брандмауера

Для активного реагування реалізовано інструмент `block_ip`, який використовує команду `New-NetFirewallRule`. Важливо зазначити, що MCP сервер автоматично додає опис причини блокування, згенерований нейромережею, безпосередньо у метадані правила Windows Firewall. Це дозволяє системним адміністраторам розуміти контекст блокування навіть без доступу до логів агента.

Окрім активних дій блокування, агент потребує інструментів для збору розвідувальної інформації. При аналізі аномального трафіку сама лише IP-адреса не дає моделі достатньо підстав для висновку про зловмисність. Для вирішення цієї проблеми розроблено функцію `get_ip_info`, наведену на рисунку 3.3, яка реалізує механізм геопросторової прив'язки IP-адрес.

```
@mcp.tool()
def get_ip_info(ip_address: str) -> str:
    # Валідація IP-адреси
    try:
        ipaddress.ip_address(ip_address)
    except ValueError:
        return f"Error: '{ip_address}' is not a valid IP address."
    response = None
    # Запит даних через API
    data = requests.get(f"https://ipapi.co/{ip_address}/json/", headers=HEADERS, timeout=10)
    if data.status_code == 200:
        response = data.json()
    # Якщо дані отримано, формування та повернення
    if response:
        result = {
            "status": "success",
            "ip": response.get("ip", ip),
            "country": response.get("country_name"),
            "organization": response.get("org"),
            "isp": response.get("org"),
            "network": response.get("network")
        }
        return json.dumps(result, indent=2)
    else: # Повернення помилки
        return json.dumps({
            "ip": ip,
            "status": "error",
            "message": "Request failed"
        }, indent=2)
```

Рис. 3.3. Реалізація функції отримання GeoIP даних

Такий підхід дозволяє моделі виявляти складні патерни загроз. Наприклад, якщо модель бачить сканування портів з IP-адреси, що належить хмарному

провайдеру в нетиповому для даної IoT-системи регіоні, це значно підвищує ймовірність класифікації події як атаки.

Також система містить багаторівневу обробку виключень. На рівні MSCP-сервера реалізовано перехоплення помилок валідації, наприклад, спроба передати некоректний формат IP-адреси та помилок виконання команд оболонки. У випадку збою PowerShell-сценарію, сервер повертає моделі структуроване повідомлення про помилку замість виходу з ладу процесу. Це дозволяє LLM спробувати альтернативну дію або залогувати інцидент без переривання загального циклу моніторингу.

Важливим елементом системи є логування. Оскільки робота моделі базується на імовірнісних підходах, критично важливим є збереження «ланцюжка думок». Лог-файл зберігає не лише вхідний JSON та кінцеве рішення, але й текстове пояснення моделі, чому саме ця сукупність даних була розцінена як атака. Це дозволяє проводити подальший аудит системи.

3.2. Обґрунтування вибору LLM та методика синтетичного тестування

Ключовим компонентом розробленої системи виявлення вторгнень є модуль інтелектуального аналізу, побудований на базі великої мовної моделі. Ефективність роботи агента на периферійному обладнанні напряму залежить від архітектури обраної нейромережі. Оскільки, ресурси шлюзу IoT обмежені, критично важливим є пошук балансу між «когнітивними здібностями» моделі та її обчислювальною складністю. Занадто велика модель може сповільнити роботу всієї мережі, тоді як занадто проста пропустити складну кібератаку. Тому вибір правильної архітектури стає фундаментом для надійності всього комплексу безпеки.

Для інтеграції у систему розглядалися сучасні квантовані моделі з кількістю параметрів до 10 мільярдів, що є оптимальним порогом для локального інференсу без використання промислових GPU.

Процедура відбору базувалася на ієрархічній системі критеріїв:

- здатність моделі вірно визначити тип атаки або її відсутність;

– дотримання заданої структури JSON. Це критичний технічний параметр, оскільки програмний парсер агента не зможе обробити відповідь, що містить зайвий текст або синтаксичні помилки;

– час генерації відповіді, що впливає на пропускну здатність системи захисту.

Щоб уникнути випадковості, притаманної тестуванню на справжньому трафіку, для етапу вибору моделі було розроблено набір із п'яти синтетичних тестів. Вони покривають різні класи мережевої активності, від легітимного трафіку з високим навантаженням до критичних атак на відмову в обслуговуванні. Детальний опис сценаріїв, типи імітаційних загроз та еталонні реакції системи, які вважалися успішним проходженням тесту, наведено в таблиці 3.1.

Таблиця 3.1

Опис тестових сценаріїв та очікуваної поведінки системи

№	Назва сценарію	Клас події	Опис ситуації	Очікувана дія агента
1	Масовий SYN-флуд	Критичний (DoS)	Переповнення черги з'єднань	block_ip (ip: 45.133.1.55)
2	Повільне сканування портів	Попередження (розвідка)	Повільне сканування портів для виявлення вразливостей	get_ip_info та block_ip (ip: 91.200.12.1)
3	UDP-флуд	Критичний (DoS)	Об'ємна атака UDP-пакетами без встановлення з'єднання	block_ip (ip: 185.10.10.10)
4	Потокове відео	Безпечний (трафік)	Легітимний трафік великого обсягу	log_safe_traffic (ip: 142.250.1.1)
5	Telnet-зондування	Підозрілий	Спроба підключення до незахищеного порту	get_ip_info та block_ip (ip: 115.23.1.1)

Узагальнені кількісні показники ефективності досліджуваних моделей наведено в таблиці 3.2. Для оцінки якості прийняття рішень використовувалася п'ятибальна шкала, де 0 відповідає повній нездатності моделі виконати завдання через галюцинації, помилки формату, а 5, це безпомилковій ідентифікації загрози та вибору коректного інструменту реагування.

Таблиця 3.2

Порівняльні результати тестування LLM

Модель	Параметри	TTFT, (s)	Швидкість, (t/s)	Рахунок, (0-5)	Характер поведінки
Qwen 3	8B, Q4_K_M	0.92	47.06	5/5	Висока точність завдяки механізму роздумів; безпомилкове реагування
Llama 3.1	8B, Q4_K_M	1.77	50.43	2/5	Пропускає критичні атаки, плутає контекст
Hermes 3	8B, Q4_0	1.50	54.10	1/5	Ігнорує системний промпт
Qwen 2.5-Coder	7B, Q4_K_M	1.03	48.95	0/5	Критичні помилки пріоритетів безпеки
Phi-4 Mini	3.8B, Q4_K_M	0.92	75.76	0/5	Технічна неспроможність, повертає опис схеми замість виклику

Детальний аналіз логів тестування виявив суттєві відмінності у стратегіях моделей, які неможливо оцінити лише числовими метриками. Беззаперечним лідером випробувань стала модель Qwen 3 8B, яка завдяки вбудованому механізму ланцюжка думок єдина змогла безпомилково класифікувати всі типи загроз, включаючи складні для розпізнавання вектори UDP-флуд та Telnet-зондування. Хоча повний час

генерації відповіді у цієї моделі був найдовшим через процес внутрішніх роздумів, критичний для безпеки показник TTFT залишався стабільно низьким, що дозволяє системі розпочати реакцію миттєво, ще до завершення повного формування відповіді.

На противагу лідеру, індустріальний стандарт Llama 3.1 продемонстрував небезпечну схильність до хибно-негативних рішень. У критичних сценаріях, таких як SYN-flood, модель ігнорувала ознаки атаки, класифікуючи аномальний трафік як легітимний, а також ускладнювала роботу парсера, додаючи до JSON-структури надлишковий описовий текст. Схожа проблема пріоритетів виявилася і у моделі Qwen 2.5-Coder. Попри ідеальне дотримання синтаксису JSON, вона продемонструвала повну нездатність розрізняти контексти безпеки, маркуючи DoS-атаки як «безпечні», що робить її непридатною для логічного ядра системи захисту.

Технічну невідповідність архітектурним вимогам виявили моделі Hermes 3 та Phi-4 Mini. Hermes 3 ігнорувала системний промпт, намагаючись викликати неіснуючі функції для «перестраховки» замість активного блокування загрози. У свою чергу, найлегша модель Phi-4 Mini, попри найвищу швидкість генерації, виявилася вразливою до феномену «витоку схеми» коли замість виконання виклику інструменту модель повертала його декларативний опис, що свідчить про недостатню увагу малих моделей до складних інструкцій без додаткового донавчання.

На основі проведеного порівняльного аналізу, як базову модель для системи виявлення атак обрано Qwen 3 8B. Вона забезпечує найкращий баланс між точністю виявлення загроз та швидкістю реакції першого токена. Від використання моделі Phi-4 Mini, попри її привабливість для IoT-сегменту, довелося відмовитися через критичні помилки у форматі передачі даних, які неможливо виправити лише засобами промпт-інжинірингу.

3.3. Аналіз результатів роботи системи

Ефективність розробленого інтелектуального агента оцінювалася комплексно, враховуючи прозорість прийняття рішень, точність виявлення атак та часові характеристики обробки запитів. У ході експериментів на віртуалізованому стенді

було змодельовано UDP Flood, сканування портів, легітимний трафік, а також спроби з'єднання з підозрілих IP-адрес, що дозволило перевірити інструментарій геолокації. Окрему увагу було приділено аналізу показників хибнопозитивних та хибнонегативних спрацювань, оскільки мінімізація помилок є ключовою вимогою для автоматизованих систем захисту.

Важливою особливістю розробленої системи є здатність LLM аргументувати свої дії, аналізуючи не лише кількісні показники, а й семантику мережевої поведінки. На відміну від статичних правил, які просто реагують на порогові значення, модель зіставляє різні параметри для формування цілісної картини інциденту. Це дозволяє системі адаптуватися до нових, раніше невідомих атак, які можуть успішно маскуватися під легітимну активність і оминати стандартні сигнатурні методи захисту.

Для демонстрації покрокової роботи агента розглянемо детально інцидент із зовнішнім скануванням. На рисунку 3.4 наведено фрагмент агрегованих даних трафіку у форматі JSON. У цей тестовий набір було впроваджено пакет із зовнішньою IP-адресою для імітації сканування портів. Така симуляція дозволила перевірити здатність агента динамічно обирати інструмент `get_ip_info` під час обробки інциденту та оцінити коректність передачі контекстних даних між модулями системи.

```
INFO - Captured 13 packets.  
INFO - Analyzing chunk 0-5 of 1 flows  
INFO - Processing batch: [  
  {  
    "src_ip": "116.10.10.55",  
    "dst_ip": "224.0.0.251",  
    "protocol": "TCP",  
    "duration_sec": 1.02,  
    "total_packets": 13,  
    "total_bytes": 1007,  
    "avg_packet_size": 77.5,  
    "flag_counts": {  
      "SYN": 12  
    },  
    "dst_ports": [21, 22, 23, 53, 80, 443, 3306, 5000, 5353, 8080, 8888, 9090, 25565],  
    "dst_ports_count": 13  
  },  
]  
]
```

Рис. 3.4. Вхідний JSON-об'єкт із параметрами підозрілої TCP-сесії

Аналіз наведених даних дозволяє виявити високий рівень загрози, передусім через параметр `flag_counts`, який вказує на наявність дванадцяти пакетів із прапором SYN, що є характерною ознакою етапу ініціалізації з'єднання. Про відсутність корисного навантаження свідчить малий середній розмір пакету на рівні 77.5 байт. Найбільш критичним показником виступає масив `dst_ports` із переліком портів призначення, який включає часто вразливі до атак грубої сили порти віддаленого управління FTP 21, SSH 22 та Telnet 23, а також порт бази даних MySQL 3306 і веб-інтерфейси на портах 80, 443 та 8080.

Така послідовність запитів за короткий проміжок часу однозначно класифікується як сканування портів з метою пошуку вразливих сервісів.

На рисунку 3.5 зображено результат внутрішніх «роздумів» моделі, отриманий під час обробки пакету.

```
INFO - Processing batch of 1 flows ...
INFO - HTTP Request: POST http://127.0.0.1:11434/api/chat "HTTP/1.1 200 OK"
INFO - Agent response: role='assistant' content='' thinking='Okay, let's tackle this network flow analysis. The user provided a JSON object with various fields. My task is to determine the appropriate action based on the given rules.\n\nFirst, I'll start with the SAFE TRAFFIC CHECK. The avg_packet_size is 77.5 bytes, which is below 800, so that condition isn't met. The duration is 1.02 seconds, which is over 0.5s, but since the packet size isn't large, this might not be video or file transfer. So maybe this isn't safe traffic.\n\nNext, checking for CRITICAL THREATS. The total_packets is 13, which is way below 1000. The duration is 1.02s, so even if we consider the second part of the condition (duration <1.0s and packets >10), here the duration is slightly over 1.0s. So this doesn't trigger the flood detection. So no block here.\n\nThen, SUSPICIOUS ACTIVITY. The dst_ports_count is 13, which is more than 3. Also, looking at the ports, there are several sensitive ones like 22 (SSH), 23 (Telnet), 3389 (RDP), 53 (DNS), etc. Even though 53 is DNS, but the presence of multiple ports might indicate scanning. So according to rule 3, since dst_ports_count >3, I should call get_ip_info to check the origin of the IP 116.10.10.55.\n\nWait, but the user's note says that if there's a high number of packets but short duration, it's a flood. Here, the duration is 1.02s, which is over 1s, so maybe not. But the total packets are 13, which is low. So maybe not a flood.\n\nAlso, checking if the IP is already blocked via firewall. But the user hasn't provided that info, so maybe I should first call get_ip_info to check the geo location. If the IP is from a high-risk region, then combined with scanning activity, it would trigger a block.\n\nSo the next step is to call get_ip_info for the src_ip "116.10.10.55". That's the logical next action here. Once I get that info, I can check if it's from a high-risk area and then decide whether to block based on the geo-blocking policy.\n\nimages=None tool_name=None tool_calls=[ToolCall(function=Function(name='get_ip_info', arguments={'ip_address': '116.10.10.55'}))]
```

Рис. 3.5. Лог роздумів агента перед викликом інструменту

Процес прийняття рішення інтелектуальним агентом відбувається поетапно шляхом виключення менш ймовірних гіпотез. На початку модель оцінює середній розмір пакету, який у даному випадку становить 77.5 байт. Оскільки це значення суттєво менше за типові показники для передачі великих файлів, система відхиляє припущення про легітимний трафік. Такий механізм ранньої фільтрації дозволяє моделі зосередитися на найбільш вірогідних причинах аномалії, не витрачаючи ресурси на перевірку малоімовірних сценаріїв.

На наступному етапі виконується перевірка на наявність ознак атак типу відмови в обслуговуванні. Агент фіксує, що за час трохи більше однієї секунди надійшло лише 13 пакетів. Така інтенсивність є занадто низькою для кваліфікації активності як мережевого флуду, тому сценарій автоматичного аварійного блокування не активується.

Основна увага системи зосереджується на аномаліях транспортного рівня. Виявивши спроби з'єднання з 13 різними портами призначення, серед яких присутні вразливі служби віддаленого керування SSH та протокол віддаленого робочого столу RDP, модель ідентифікує цю поведінку як цілеспрямоване сканування мережі. Враховуючи виявлену загрозу, агент приймає рішення не блокувати адресу миттєво, а спершу звернутися до інструменту визначення геолокації для перевірки репутації та походження IP-адреси.

На рисунку 3.6 показано момент технічної взаємодії між мовною моделлю та програмним ядром системи, коли керуючий скрипт перехоплює ініційований запит на виконання функції. Протокол роботи демонструє, що система успішно розпізнала намір агента використати інструмент отримання інформації про IP-адресу та автоматично запустила відповідну процедуру для підозрілого вузла 116.10.10.55.

```
INFO - ✨ Agent requests 1 actions ...
INFO - → Executing get_ip_info: {'ip_address': '116.10.10.55'} ...
INFO - → Result get_ip_info: {'output': '{"\\n \\\"status\\\": \\\"success\\\",\\n \\\"ip\\\": \\\"116.10.10.55\\\",\\n \\\"country\\\": \\\"China\\\",\\n \\\"organization\\\": \\\"Chinanet\\\",\\n \\\"isp\\\": \\\"Chinanet\\\",\\n \\\"network\\\": \\\"116.10.8.0/21\\\"\\n}"}', 'error': False}
```

Рис. 3.6. Лог перехоплення запиту на виконання функції

У відповідь на цей запит програмний модуль виконав звернення і повернув результат у структурованому вигляді, який містив підтвердження походження трафіку з Китаю та ідентифікацію провайдера Chinanet. Цей етап є ключовим доказом здатності розробленої системи перетворювати текстові висновки штучного інтелекту на реальні технічні команди, забезпечуючи двосторонній зв'язок між аналітичним ядром та інструментарієм моніторингу.

Наступний етап роботи системи полягає в інтеграції отриманих розвідувальних даних у загальний контекст інциденту. Результат виконання функції геолокації

передається назад у контекстне вікно мовної моделі, що дозволяє їй оцінити рівень загрози з урахуванням нових фактів. На рисунку 3.7 показано процес прийняття остаточного рішення, де агент поєднує попередньо виявлену підозрілу поведінку сканування портів із фактом походження трафіку з регіону підвищеного ризику.

```
INFO - Agent response: role='assistant' content='' thinking="Okay, let's see. The user provided a network flow where the source IP is 116.10.10.55. After calling get_ip_info, the response shows that this IP is located in China, associated with China. Now, I need to apply the geo-blocking policy.\n\nLooking back at the rules, if the IP is from a high-risk region (like China) and the traffic involves scanning or sensitive ports, I should block it. The original flow had dst_ports_count of 13, including sensitive ports like 22, 23, etc. Since the IP is from China, which is considered high-risk, and the traffic patterns suggest scanning, the next step is to block this IP.\n\nSo, I need to call block_ip with the reason combining the geo-location and the scanning activity. The reason should mention both factors to justify the block.\n" images=None tool_name=None tool_calls=[ToolCall(function=Function(name='block_ip', arguments={'ip_address': '116.10.10.55', 'reason': 'Scanning activity from high-risk geo-location (China) targeting sensitive ports'})]]
```

Рис. 3.7. Аналіз отриманих геоданих та формування команди на блокування

Модель застосовує політику безпеки, яка вимагає блокування зовнішніх сканерів. У полі аргументації рішення агент чітко формулює причину, посилаючись на поєднання активності сканування та походження з Китаю. Це демонструє здатність системи генерувати зрозумілі для людини пояснення, що є критично важливим для подальшого аудиту інцидентів безпеки.

Після генерації команди на блокування керуючий модуль системи перехоплює цей виклик і транслює його у відповідну команду операційної системи. На рисунку 3.8 показано момент виконання захисної дії, де програмний засіб застосовує правило блокування для IP-адреси 116.10.10.55.

```
INFO - ✨ Agent requests 1 actions...
INFO - → Executing block_ip: {'ip_address': '116.10.10.55', 'reason': 'Scanning activity from high-risk geo-location (China) targeting sensitive ports'}...
INFO - → Result block_ip: {'output': 'Successfully blocked IP 116.10.10.55. \nRule Name: MCP Blocked: 116.10.10.55\nReason: Scanning activity from high-risk geo-location (China) targeting sensitive ports', 'error': False}
```

Рис. 3.8. Лог успішного виконання інструменту

Успішне створення правила підтверджується системним повідомленням, яке містить статус операції та записану причину блокування. Цей етап ілюструє перехід від теоретичного планування захисту до його практичної реалізації, забезпечуючи ізоляцію загрози в реальному часі.

Завершальною фазою циклу реагування є верифікація виконаних дій. Система повертає статус успішного блокування назад до мовної моделі, щоб агент міг

переконатися у розв'язанні проблеми. На рисунку 3.9 зображено фінальний висновок агента, де він підсумовує весь ланцюжок подій.

```
INFO - Agent response: role='assistant' content='No further actions required. The IP 116.10.10.55 has been successfully blocked for scanning activity from a high-risk geo-location (China) targeting sensitive ports. All rules have been appropriately addressed.' thinking='Okay, let me recap what's happening here. The user provided a network flow with the source IP 116.10.10.55. The initial analysis triggered a check for suspicious activity because the destination ports count was high and included sensitive ports. Then, after checking the IP's geolocation, it turned out to be in China, which is a high-risk region. According to the geo-blocking policy, combining that with the scanning activity led to blocking the IP.\n\nNow, the user's last response shows that the block_ip function was successfully called, and the IP is blocked. Since all actions have been taken based on the rules, there's no further action needed. The flow is handled, so the process can end here.\n" images=None tool_name=None tool_calls=None\nINFO -  Agent completed analysis for this batch.
```

Рис. 3.9. Фінальний звіт агента та завершення циклу обробки загрози

Модель аналізує підтвердження про блокування і робить висновок про відсутність потреби у подальших діях, оскільки всі умови політики безпеки були виконані. Такий підхід із зворотним зв'язком гарантує, що система не залишає інциденти у невизначеному стані та чітко сигналізує про завершення обробки конкретного пакету загроз.

Окрім розглянутого сценарію розвідки, було проведено тестування реакції системи на об'ємні атаки та нормальну мережеву активність. Це дозволило перевірити універсальність запропонованого підходу.

У випадку UDP-флуду система обробляла пакети з аномально високою частотою надходження, що перевищувала 100 пакетів за 0,5 секунди. На відміну від сканування портів, де модель шукала контекст через зовнішні інструменти, тут спрацював механізм швидкого реагування. Агент ідентифікував загрозу виключно на основі статистичних показників JSON-об'єкта, зокрема співвідношення загальної кількості пакетів до тривалості, класифікував це як DoS-атаку і миттєво згенерував команду на блокування без додаткових запитів до API геолокації. Це підтверджує, що LLM здатна пріоритизувати швидкість реакції для критичних загроз.

Паралельно тестувався сценарій легітимного трафіку, що імітував стандартну роботу IoT-пристроїв, як-от HTTP-запити та передачу телеметрії. У цих випадках модель проаналізувала відсутність прапорів атаки, адекватний розмір пакетів та використання стандартних портів. Результатом роботи став висновок виклику

інструменту для логування легітимного трафіку, що свідчить про відсутність хибних спрацювань у штатному режимі роботи мережі.

Зведені результати реакції агента на різні типи вхідних даних наведено в таблиці 3.3.

Таблиця 3.3

Порівняльні результати тестування LLM

Тип трафіку	Ключові ознаки	Ланцюжок думок	Використані інструменти	Результат
Сканування портів	Багато dst_ports, прапори SYN	Підозра на розвідку -> потрібна перевірка IP -> IP ризиковий -> блокування	get_ip_info, block_ip	Заблоковано
UDP-флуд	Високий packet_rate >200/сек	Явні ознаки DoS-атаки - > негайне блокування	block_ip	Заблоковано
Нормальний	Стандартні порти, низька частота	Аномалій не виявлено -> логування легітимного трафіку	log_safe_traffic	Пропущено

Одним із критичних параметрів впровадження LLM у системи реального часу є затримка обробки. Вимірювання часу реакції проводилося на віртуальній машині без доступу до апаратного прискорення.

Оскільки система працює в режимі ітеративного вирішення задачі, загальний час обробки складного інциденту визначається за формулою (3.1).

$$T_{total} = T_{agg} + \sum_{i=1}^N (T_{inference}^{(i)} + T_{overhead}) + \sum_{j=1}^K T_{tool_exec}^{(j)}, \quad (3.1)$$

де T_{total} – загальний час реакції системи;

T_{agg} – час попередньої обробки даних;

N – кількість кроків логічного виведення;

$T_{inference}$ – час логічного виведення моделі;

$T_{overhead}$ – витрати на парсинг відповідей;

K – кількість викликів інструментів;

T_{tool_exec} – час виконання зовнішніх запитів.

Під час експериментів середній час повного циклу аналізу для сценаріїв, що вимагають підтвердження через зовнішні інструменти, склав близько 5 секунд. Варто зазначити, що більша частка цього часу припадає на багатоетапну верифікацію рішення.

На відміну від сигнатурних систем, які працюють миттєво, але часто помиляються, LLM-агент витрачає час на осмислення контексту. Цей процес охоплює первинний аналіз аномалії, формування запиту до відповідного інструменту, подальшу інтерпретацію отриманих результатів і, зрештою, прийняття фінального рішення.

Така тривалість обробки є компромісом між швидкістю та точністю. Для виявлення складних атак, де точність є пріоритетом над миттєвістю, така затримка є допустимою. Водночас для блокування швидкісних флуд-атак система може використовувати скорочений цикл, зменшуючи час реакції.

3.4. Аналіз ефективності та рекомендації

Узагальнення результатів експерименту з використанням моделі Qwen3:8b дозволяє стверджувати, що інтеграція генеративного штучного інтелекту в системи моніторингу IoT відкриває принципово нові можливості для виявлення складних векторів атак. Запропонований підхід, що базується на конвертації трафіку у структурований JSON та його подальшому семантичному аналізі, демонструє значну перевагу в точності прийняття рішень порівняно з традиційними методами.

Головною перевагою розробленого інтелектуального агента є здатність до глибокого контекстного аналізу та адаптивної поведінки. На відміну від сигнатурних систем, які функціонують за принципом збігу шаблонів, модель Qwen3 аналізує сукупність факторів і наміри зловмисника. Реалізований механізм ланцюжка міркувань дозволяє системі не лише реагувати на окремий пакет, а й вибудовувати логічну послідовність дій.

Особливу цінність демонструє здатність агента залучати зовнішні інструменти для верифікації підозр. У ході тестування модель успішно ідентифікувала неочевидні загрози, ініціюючи додаткові запити до сервісів геолокації. Тільки після отримання та аналізу цих додаткових даних агент формував остаточний вердикт. Така багатоступенева перевірка дозволяє кардинально знизити рівень помилкових спрацювань, адже рішення приймається на основі повної картини інциденту, а не лише заголовків одного пакету [16].

Висока точність та інтелектуальна глибина аналізу мають свою ціну у вигляді часових затримок. Експерименти показали, що повний цикл обробки складної події може тривати до 5 секунд. Цей показник зумовлений не стільки швидкістю самого інференсу моделі, скільки ітеративною природою алгоритму. Агент витрачає час на генерацію проміжних думок, виконання зовнішніх API-запитів та повторний аналіз отриманих результатів для прийняття фінального рішення.

Важливо зазначити, що така затримка є виправданою для розслідування складних інцидентів, де ціна помилки висока. Система демонструє здатність розплутувати складні сценарії атак, які пропускають швидкісні автоматичні фільтри. Тому, попри відносно високу ресурсоємність та тривалий час реакції на стандартному обладнанні, агент забезпечує якісно вищий рівень розуміння ситуації, виступаючи фактично в ролі автоматизованого аналітика, а не простого фільтра пакетів.

З огляду на отримані характеристики, найбільш ефективною стратегією впровадження є використання гібридної архітектури. Рекомендовано застосовувати дворівневу систему захисту. На першому рівні працюють класичні високошвидкісні фільтри для відсіювання очевидного трафіку. Розроблений агент на базі LLM має розміщуватися на другому рівні або в режимі асинхронного моніторингу, приймаючи

на вхід лише підозрілі або аномальні потоки даних, які потребують глибокого розслідування.

Перспективним напрямком подальшого розвитку є перенесення обчислень на спеціалізовані модулі NPU в рамках концепції Edge AI, що дозволить зберегти високу точність моделі при зменшенні часу реакції [26].

3.5. Висновки до розділу 3

У даному розділі здійснено програмну реалізацію прототипу системи виявлення вторгнень, архітектура якої базується на інтеграції великої мовної моделі з локальними інструментами керування мережею. Використання стандарту Model Context Protocol дозволило створити модульну структуру, де логічне ядро агента відокремлене від виконавчих механізмів операційної системи. Такий підхід забезпечив гнучкість рішення та можливість безпечного виконання команд блокування на рівні брандмауера Windows без ризику порушення цілісності системи.

За результатами порівняльного тестування п'яти сучасних квантованих моделей обґрунтовано вибір Qwen3 як оптимального рішення для задач кібербезпеки. Експериментально доведено, що моделі меншої розмірності мають критичні проблеми з дотриманням синтаксису JSON та схильні до галюцинацій при інтерпретації технічних даних. Обрана модель завдяки вбудованому механізму ланцюжка міркувань продемонструвала стовідсоткову точність класифікації тестових загроз та здатність аргументувати свої рішення зрозумілою мовою.

Практичні випробування підтвердили спроможність системи виявляти не лише примітивні об'ємні атаки, а й складнішу розвідувальну активність. Реалізований алгоритм динамічного залучення інструментів дозволив агенту самостійно ініціювати перевірку геолокації підозрілих IP-адрес перед застосуванням блокування. Це якісно відрізняє розроблене рішення від статичних сигнатурних методів, оскільки вердикт формується на основі сукупності контекстних даних, а не лише заголовків окремих пакетів.

Аналіз продуктивності виявив головне обмеження технології у вигляді часової затримки обробки, яка сягає п'яти секунд для складних сценаріїв. Отримані дані свідчать про недоцільність використання генеративних моделей як первинного фільтра на високошвидкісних каналах. Натомість доведено ефективність інтеграції розробленого агента у гібридні системи захисту як аналітика другого рівня, здатного розслідувати аномалії, пропущені автоматичними засобами.

РОЗДІЛ 4

ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1. Охорона праці

Забезпечення належного стану охорони праці є невід'ємною складовою ефективною науково-дослідною діяльністю, особливо при виконанні робіт, пов'язаних із використанням сучасних інформаційних технологій та апаратних засобів. Метою даного розділу є аналіз та забезпечення безпечних умов праці під час дослідження методів і засобів інтелектуального виявлення кібератак в мережах Інтернету речей. Специфіка роботи магістранта передбачає тривалу взаємодію з обчислювальною технікою, налаштування серверного обладнання для розгортання великих мовних моделей, розробку програмного забезпечення мовою Python та аналіз мережевого трафіку. Робочий процес здійснюється у лабораторному приміщенні, яке за ступенем небезпеки ураження електричним струмом належить до приміщень без підвищеної небезпеки. Проте робота супроводжується впливом таких виробничих факторів, як електромагнітне випромінювання, статичне напруження м'язів, напруження зорового аналізатора та потенційний ризик ураження електричним струмом.

Основним нормативним документом, що регулює безпеку праці при виконанні завдань, пов'язаних з ІТ-розробкою, є НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями» [27]. Відповідно до цих Вимог, робоче місце дослідника організовується таким чином, щоб мінімізувати фізичне та психофізіологічне навантаження на організм. Робочий стіл та крісло повинні відповідати принципам ергономіки, забезпечуючи підтримання фізіологічно раціональної робочої пози. Екран відеотермінала слід розташовувати на відстані 600-700 мм від очей користувача, а лінія погляду має падати на центр або верхню частину екрана. Клавіатура повинна розміщуватися на поверхні столу на відстані 100-300 мм від краю, що створює опору для рук і запобігає перенапруженню кистей та розвитку захворювань опорно-рухового апарату при тривалому наборі коду.

Враховуючи високу інтелектуальну напругу під час аналізу кіберзагроз та роботи з нейромережами, критично важливим є дотримання раціонального режиму праці та відпочинку. Згідно з рекомендаціями чинного законодавства, необхідно робити регламентовані перерви тривалістю 10-15 хвилин через кожні 60-120 хвилин роботи. Це сприяє зняттю зорового напруження та зниженню втоми. Важливим аспектом також є забезпечення належного мікроклімату у приміщенні, де працює серверне обладнання та IoT-пристрої. Температура повітря повинна підтримуватися в межах 22-24 °C у теплий період року та 19-21 °C у холодний, при відносній вологості 40-60 %. Освітлення робочої зони має бути рівномірним, комбінованим і достатнім для роботи з дрібними деталями мікроконтролерів та читання технічної документації.

Особливу увагу в контексті роботи з апаратними компонентами IoT та серверами слід приділити електробезпеці. Експлуатація обладнання здійснюється від мережі змінного струму напругою 220 В, що згідно з «Правилами улаштування електроустановок» [28] вимагає суворого дотримання технічних норм. Електрична мережа лабораторії повинна бути виконана як трипровідна система, де нульовий робочий та нульовий захисний провідники розділені. Усі металеві неструмоведучі частини комп'ютерного обладнання, серверних стійок та вимірювальних приладів підлягають обов'язковому заземленню. Забороняється використання пошкоджених розеток, кабелів із порушеною ізоляцією та саморобних подовжувачів. Для підвищення рівня безпеки рекомендується використання пристроїв захисного вимкнення, які запобігають ураженню електричним струмом у разі витoku на корпус обладнання.

Пожежна безпека у приміщенні забезпечується відповідно до Кодексу цивільного захисту України [29], «Правил пожежної безпеки в Україні» (НАПБ А.01.001-2014) [30] та ДБН В.2.5-56:2014 «Системи протипожежного захисту» [31]. Специфіка роботи з навантаженим обчислювальним обладнанням створює ризик перегріву компонентів або короткого замикання, що може стати причиною загоряння. Приміщення має бути обладнане автоматичною пожежною сигналізацією з димовими сповіщувачами. Як первинні засоби пожежогасіння слід використовувати

вуглекислотні вогнегасники, які є найбільш ефективними для гасіння електроустановок під напругою та не завдають шкоди електронним компонентам, на відміну від порошкових аналогів. Евакуаційні шляхи та доступи до засобів пожежогасіння повинні постійно утримуватися вільними.

Таким чином, комплексне виконання вимог щодо ергономіки робочого місця, електробезпеки та протипожежного захисту створює безпечні умови для виконання кваліфікаційної роботи. Це дозволяє досліднику зосередитися на вирішенні наукових завдань у сфері кібербезпеки IoT, зберігаючи при цьому здоров'я та високу працездатність, а також гарантує збереження дороговартісного обладнання та результатів дослідження.

4.2. Безпека в надзвичайних ситуаціях

Забезпечення стійкості функціонування комп'ютерних систем та мереж Інтернету речей в умовах надзвичайних ситуацій вимагає детального аналізу впливу специфічних уражаючих факторів, серед яких особливе місце посідає електромагнітний імпульс ядерного вибуху. ЕМІ становить найбільшу небезпеку для електронної апаратури, систем зв'язку та управління, навіть якщо вони розташовані в особливо міцних спорудах, що витримують дію ударної хвилі. Відповідно до методичних рекомендацій, ядерний ЕМІ переносить основну частину енергії в діапазоні частот від 3 Гц до 30 кГц, а напруженість електричного поля може досягати 50000 В/м [32]. Таке різке зростання напруженості індукує високу електричну напругу в кабельних лініях електромереж та інформаційних комунікаціях, що призводить до виділення значної кількості тепла та електричного пробоя ізоляції. Для напівпровідникових приладів, які є основою мікропроцесорної техніки та IoT-пристроїв, це критично, оскільки навіть незначні перенапруги викликають тепловий пробій р-п переходів. Встановлено залежність, згідно з якою чим вищий коефіцієнт підсилення транзистора, тим менша його електрична міцність, що робить сучасну високочутливу електроніку вкрай вразливою до дії ЕМІ [32]. Найбільших значень

досягають напруги, що наводяться між жилою кабелю та землею, що може призвести до повного виходу з ладу незахищених систем автоматизованого управління.

Для підвищення стійкості автоматизованих ліній та IoT-інфраструктури необхідно застосовувати комплексні методи захисту, серед яких ключовими є екранування, заземлення та використання спеціальних захисних пристроїв. Надійним методом захисту є використання суцільних металевих екранів, виконаних, наприклад, з холоднокатаних сталевих листів, з'єднаних зварюванням в інертному газі, що забезпечує коефіцієнт екранування понад 100 дБ [32]. Важливим елементом захисту є встановлення на входах і виходах апаратури, а також у лініях живлення захисних розрядників та обмежувачів перенапруги. Розрізняють «м'які» обмежувачі, такі як варистори, що змінюють опір залежно від напруги, та «жорсткі» прилади з пробоем, наприклад, газові розрядники [32]. У контексті захисту розподілених IoT-мереж доцільним є перехід на волоконно-оптичні лінії зв'язку, які, будучи діелектричними, не піддаються впливу електромагнітної індукції. Крім того, для забезпечення надійності електропостачання в умовах дії уражаючих факторів рекомендується прокладання кабельних ліній під землею та створення автономних резервних джерел живлення [33].

У випадку виникнення надзвичайної ситуації та пошкодження інфраструктури критично важливим етапом є організація та проведення аварійно-відновлювальних робіт на комп'ютерних та електричних мережах. Цей процес починається з інженерної розвідки для оцінки ступеня руйнувань та визначення обсягів робіт. Відновлення енергопостачання є пріоритетним завданням, оскільки від нього залежить функціонування систем життєзабезпечення та управління. Згідно з рекомендаціями щодо підвищення стійкості систем енергопостачання, необхідно передбачати можливість підключення об'єктів за кільцевою схемою або від двох незалежних джерел [33]. Під час проведення відновлювальних робіт на електричних мережах першочергово здійснюється локалізація пошкоджених ділянок, перевірка цілісності ізоляції та заміна зруйнованих елементів комутаційного обладнання. Для комп'ютерних мереж відновлювальні роботи включають діагностику серверного та мережевого обладнання, перевірку цілісності кабельних трас та відновлення

програмного забезпечення із резервних копій. Всі роботи повинні виконуватися спеціалізованими формуваннями цивільного захисту з суворим дотриманням правил техніки безпеки, особливо в умовах можливого вторинного впливу уражаючих факторів [33]. Ефективність відновлення залежить від завчасного створення матеріально-технічних резервів, включаючи запасні частини, кабельно-провідникову продукцію та вимірювальну апаратуру, що дозволяє мінімізувати час простою автоматизованих систем управління в кризових ситуаціях.

ВИСНОВКИ

У кваліфікаційній роботі магістра розв'язано задачу автоматизованого виявлення та реагування на кібератаки в IoT-середовищах шляхом розробки інтелектуального агента на базі великої мовної моделі. Основні результати дослідження:

1) На основі аналізу наукових джерел показано неефективність традиційних методів захисту в умовах гетерогенних IoT-мереж, що дало змогу обґрунтувати доцільність використання локальних LLM для семантичного аналізу атак без передачі даних у хмару.

2) Розроблено методику інтеграції нейромережі, яка включає серіалізацію трафіку в JSON та використання архітектури на базі протоколу MCP і парадигми ReAct. Це дозволило безпечно поєднати логічне ядро моделі з локальними інструментами моніторингу.

3) Емпіричним шляхом обрано модель Qwen 3 8B, яка у порівняльному тестуванні продемонструвала 100% точність виявлення загроз та найкращу здатність до побудови причинно-наслідкових зв'язків порівняно з іншими аналогами.

4) Створено програмний прототип, що реалізує багаторівневу систему промптів для мінімізації галюцинацій. Система автоматично виконує геолокацію підозрілих хостів та блокує зловмисників через брандмауер Windows.

5) Проведено обчислювальний експеримент, який підтвердив здатність агента виявляти складну розвідувальну активність та надавати зрозуміле текстове пояснення кожного рішення, що спрощує аудит безпеки.

6) На основі проведеного обчислювального експерименту, визначено обмеження швидкодії, що зумовлює рекомендацію використовувати розробленого агента як аналітичний модуль другого рівня у гібридних системах захисту.

Узагальнюючи викладене, можна стверджувати, що отримані теоретичні й практичні результати повністю підтверджують робочу гіпотезу, а мета кваліфікаційної роботи досягнута шляхом успішного виконання всіх поставлених завдань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. State of IoT 2024: Number of connected IoT devices growing 13% to 18.8 billion globally. IoT Analytics. 2024. URL: <https://iot-analytics.com/number-of-connected-iot-devices-2024/> (дата звернення: 18.11.2025).
2. IoT Hacking Statistics 2025: Threats, Risks & Regulations. DeepStrike. 2025. URL: <https://deepstrike.io/blog/iot-hacking-statistics> (дата звернення: 18.11.2025).
3. OWASP IoT Top 10 Vulnerabilities. OWASP. 2025. URL: <https://owasp.org/www-project-internet-of-things/> (дата звернення: 19.11.2025).
4. 2024 Mid-Year Cyber Threat Report. SonicWall. 2024. URL: <https://www.scribd.com/document/919976189/Mid-Year-2024-Cyber-Threat-Report> (дата звернення: 19.11.2025).
5. OT/IoT Cybersecurity Trends & Insights: 1H 2025 Review. Nozomi Networks. 2025. URL: https://cdn.prod.website-files.com/645a4534705010e2cb244f50/6888029cf0e85ce8b306d073_ccf380c4028247ef1389f14738844ef9_Nozomi-Networks-OT-IoT-Security-Report-July-2025.pdf (дата звернення: 20.11.2025).
6. Securing smart home against sinkhole attack using weight-based IDS placement strategy. IET Wireless Sensor Systems. 2023. Vol. 13, Iss. 6. P. 1–11.
7. Khraisat A., Alazab A. A critical review of intrusion detection systems in the internet of things: techniques, deployment strategy, validation strategy, attacks, anomalies and challenges. Cybersecurity. 2021. Vol. 4, № 1. P. 1–27.
8. Al-Garadi M. A. et al. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. IEEE Communications Surveys & Tutorials. 2020. Vol. 22, № 3. P. 1646–1685.
9. Gyamfi E., Jurcut A. Intrusion Detection in Internet of Things Systems: A Review on Design Approaches and Innovations. Electronics. 2022. Vol. 11, № 8. P. 1263.
10. Ferrag M. A. et al. Revolutionizing Cyber Threat Detection with Large Language Models. IEEE Access. 2024. Vol. 12. P. 1–15.
11. Yao S. et al. ReAct: Synergizing Reasoning and Acting in Language Models. International Conference on Learning Representations (ICLR). 2023.

12. Biondi P. Scapy: Interactive Packet Manipulation Program. Scapy Documentation. 2025. URL: <https://scapy.net/> (дата звернення: 04.12.2025).
13. Bray T. The JavaScript Object Notation (JSON) Data Interchange Format (RFC 8259). Internet Engineering Task Force (IETF). 2017.
14. Chen L., Zaharia M., Zou J. FrugalGPT: How to Use Large Language Models While Reducing Cost and Improving Performance. arXiv preprint arXiv:2305.05176. 2023. URL: <https://arxiv.org/abs/2305.05176> (дата звернення: 16.11.2025).
15. Schick T. et al. Toolformer: Language Models Can Teach Themselves to Use Tools. arXiv preprint arXiv:2302.04761. 2023. URL: <https://arxiv.org/abs/2302.04761> (дата звернення: 27.11.2025).
16. Wei J. et al. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. Advances in Neural Information Processing Systems (NeurIPS). 2022. Vol. 35. P. 24824–24837.
17. Liu P. et al. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. ACM Computing Surveys. 2023. Vol. 55, № 9. P. 1–35.
18. Dettmers T. et al. QLoRA: Efficient Finetuning of Quantized LLMs. arXiv preprint arXiv:2305.14314. 2023. URL: <https://arxiv.org/abs/2305.14314> (дата звернення: 01.12.2025).
19. Nous Research. Hermes 3 Llama 3.1 8B. Hugging Face. 2024. URL: <https://huggingface.co/NousResearch/Hermes-3-Llama-3.1-8B> (дата звернення: 07.12.2025).
20. Meta. Llama 3.1 8B. Hugging Face. 2024. URL: <https://huggingface.co/meta-llama/Llama-3.1-8B> (дата звернення: 07.12.2025).
21. Microsoft. Phi-4 mini. Hugging Face. 2025. URL: <https://huggingface.co/microsoft/Phi-4-mini-instruct> (дата звернення: 08.12.2025).
22. Qwen. Qwen2.5-Coder-7B. Hugging Face. 2024. URL: <https://huggingface.co/Qwen/Qwen2.5-Coder-7B-Instruct> (дата звернення: 08.12.2025).
23. Qwen. Qwen3-8B. Hugging Face. 2025. URL: <https://huggingface.co/Qwen/Qwen3-8B> (дата звернення: 08.12.2025).

24. Gerganov G. llama.cpp: Port of Facebook's LLaMA model in C/C++. GitHub. 2023. URL: <https://github.com/ggerganov/llama.cpp> (дата звернення: 05.12.2025).
25. Model Context Protocol Specification. Anthropic. 2024. URL: <https://modelcontextprotocol.io> (дата звернення: 06.12.2025).
26. Aldhaferi S. et al. Deep Learning-Based Intrusion Detection for Internet of Things: A Comprehensive Review. IEEE Access. 2021. Vol. 9. P. 64931–64952.
27. НПАОП 0.00-7.15-18. Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. 2018. № 34.
28. Правила улаштування електроустановок (ПУЕ). затв. наказом М-ва енергетики та вугіл. пром-сті України 21.07.2017 № 476.
29. Кодекс цивільного захисту України : Закон України від 02.10.2012 № 5403-VI. Відомості Верховної Ради України. 2013. № 34–35. Ст. 458.
30. НАПБ А.01.001-2014. Правила пожежної безпеки в Україні : затв. наказом М-ва внутріш. справ України 30.12.2014 № 1417.
31. ДБН В.2.5-56:2014. Системи протипожежного захисту. Київ : Мінрегіон України, 2015.
32. Стручок В. С. Безпека в надзвичайних ситуаціях : метод. посіб. для здобувачів освіт. ступеня «магістр» всіх спеціальностей ден. та заоч. (дистанц.) форм навчання. Тернопіль : ФОП Паляниця В. А., 2022. 156 с.
33. Техноекологія та цивільна безпека. Частина «Цивільна безпека» : навч. посіб. / авт.-уклад. В. С. Стручок. Тернопіль : ФОП Паляниця В. А., 2022. 156 с.
34. Луцик Н.С., Луцків А.М., Осухівська Г.М., Тиш Є.В. Програма та методичні рекомендації з проходження практики за тематикою кваліфікаційної роботи для студентів спеціальності 123 «Комп'ютерна інженерія» другого (магістерського) рівня вищої освіти усіх форм навчання. Тернопіль: ТНТУ. 2024. 45 с.
35. Луцик Н.С., Луцків А.М., Осухівська Г.М., Тиш Є.В. Методичні рекомендації до виконання кваліфікаційної роботи магістра для студентів спеціальності 123 «Комп'ютерна інженерія» другого (магістерського) рівня вищої освіти усіх форм навчання. Тернопіль. 2024. 44 с.
36. Варавін А.В., Лецишин Ю.З., Чайковський А.В. Методичні вказівки до

виконання курсового проєкту з дисципліни «Дослідження і проєктування комп'ютерних систем та мереж» для здобувачів другого (магістерського) рівня вищої освіти спеціальності 123 «Комп'ютерна інженерія» усіх форм навчання. Тернопіль: ТНТУ, 2024. 32 с.

37. Міндзіброцький Р. Переваги адаптивних систем безпеки над статичними правилами фільтрації в IoT-мережах. Інформаційні моделі, системи та технології : Матеріали XIII наук.-техн. конф. ТНТУ ім. І.Пулюя (17-18 грудня 2025). Тернопіль, 2025. с. 64.

38. Міндзіброцький Р., Луцків А. Штучний інтелект у логістиці: від прогнозування попиту до автономних систем. Цифрова економіка як фактор інновацій та сталого розвитку суспільства : VI Міжнародна наук. – прак. конф. учених та студентів. ТНТУ ім. І.Пулюя (9-10 грудня 2025). Тернопіль, 2025. с.

39. Lutskev A., Popovych N. Big data-based approach to automated linguistic analysis effectiveness. IEEE Third International Conference on Data Stream Mining & Processing. August 21- 25, 2020, Lviv, Ukraine pp.438–443.

40. Lutskev A. Lutsyshyn R. Corpus-Based Translation Automation in Adaptable Corpus Translation Module. Computational Linguistics and Intelligent Systems. Proc. 5th Int. Conf. COLINS 2021. Volume I: Workshop. Lviv, Ukraine, April 22-23, 2021, CEUR-WS.org, online. pp.374-395.

41. Yatsyshyn V., Kharchenko O., Lutskev A. Maturity. Requirements Model for Software Requirements with the Implementation of ISO/IEC 25010 Recommendations. International Journal "Information Models and Analyses" Volume 9, Number 2, 2020. Pp.126-143.

42. Yatsyshyn V., Pastukh O., Lutskev A., Tsymbalistyy V., Martsenko N. A Risks management method based on the quality requirements communication method in agile approaches. Information Technologies: Theoretical and Applied Problems 2022 (ITTAP 2022), Ternopil, Ukraine, November 22-24, 2022. pp.1-10.

Додаток А
Тези конференцій

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
ІМЕНІ ІВАНА ПУЛЮЯ

МАТЕРІАЛИ

ХІІІ НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ,
СИСТЕМИ ТА ТЕХНОЛОГІЇ»



17-18 грудня 2025 року

ТЕРНОПІЛЬ
2025

D. Marchuk, V. Tymoshchuk

SECURITY ARCHITECTURE FOR THE 5G CORE BASED ON VIRTUALIZED NETWORK FUNCTIONS AND HARMONIZED POLICIES

Т. Маслянка, Ю. Олійник, Н. Загородна

ОБМЕЖЕННЯ СТАНДАРТНИХ МЕТРИК ЕФЕКТИВНОСТІ XDR-СИСТЕМ ПРИ АНАЛІЗІ ФІШІНГОВИХ КАМПАНІЙ

T. Maslianka, Yu. Oliinyk, N. Zagorodna

LIMITATIONS OF STANDARD EFFICIENCY METRICS OF XDR SYSTEM IN ANALYZING PHISHING CAMPAIGNS

62

А. Мельник, Л. Дмитроца

АЛГОРИТМІЧНА ОПТИМІЗАЦІЯ ПОКРИТТЯ СТАНІВ МОБІЛЬНИХ ДОДАТКІВ НА ОСНОВІ ГРАФО-ОРІЄНТОВАНОГО ПІДХОДУ

A. Melnyk, L. Dmytrotsa

ALGORITHMIC OPTIMIZATION OF STATE COVERAGE IN MOBILE APPLICATIONS BASED ON GRAPH-ORIENTED APPROACH

63

Р. Міндзіброцький

ПЕРЕВАГИ АДАПТИВНИХ СИСТЕМ БЕЗПЕКИ НАД СТАТИЧНИМИ ПРАВИЛАМИ ФІЛЬТРАЦІЇ В ІОТ-МЕРЕЖАХ

R. Mindzibrodskyi

ADVANTAGES OF ADAPTIVE SECURITY SYSTEMS OVER STATIC FILTERING RULES IN IOT NETWORKS

64

П. Містерман, М. Петрик

РОЗРОБКА МАЛОЇ МОВНОЇ МОДЕЛІ SLM ДЛЯ СИСТЕМ З ОБМЕЖЕНИМИ ОБЧИСЛЮВАЛЬНИМИ РЕСУРСАМИ НА БАЗІ AZURE ML ТА AI FOUNDRY

P. Misterman, M. Petryk

DEVELOPMENT OF A SMALL LANGUAGE MODEL (SLM) FOR RESOURCE-CONSTRAINED SYSTEMS BASED ON AZURE ML AND AI FOUNDRY

65

І. Містерман

СИСТЕМА УПРАВЛІННЯ ІНФОРМАЦІЙНОЮ БЕЗПЕКОЮ ДЕРЖАВНОЇ УСТАНОВИ НА ОСНОВІ РЕЗУЛЬТАТІВ ІНСТРУМЕНТАЛЬНОГО ОЦІНЮВАННЯ CSET

I. Misterman

INFORMATION SECURITY MANAGEMENT SYSTEM OF A GOVERNMENT INSTITUTION BASED ON THE RESULTS OF THE CSET INSTRUMENTAL ASSESSMENT

66

С. Мостовий, А. Северін

ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ АНАЛІЗУ ТЕКСТУ НА ВИЯВЛЕННЯ ПРОПАГАНДИ

S. Mostovyi, A. Severin

INTELLIGENT TEXT ANALYSIS SOFTWARE FOR PROPAGANDA DETECTION

67

А. Музичук

СУЧАСНІ СИСТЕМИ КОМП'ЮТЕРНОГО ЗОРУ В МЕДІАІНДУСТРІЇ

A. Muzychuk

MODERN COMPUTER VISION SYSTEMS IN THE MEDIA INDUSTRY

68

С. Ожанський

МАСШТАБНІ ІМЕРСИВНІ ЗАНУРЕННЯ

S. Ozhanskyi

LARGE-SCALE IMMERSIVE EXPERIENCES

69

С. Ожанський

МОЖЛИВОСТІ UNREAL ENGINE 5 ДЛЯ СТВОРЕННЯ ІМЕРСИВНИХ СЕРЕДОВИЩ

70

УДК 004.056:004.8:004.7

Р. Міндзіброцький

(Тернопільський національний технічний університет імені Івана Пулюя, Україна)

ПЕРЕВАГИ АДАПТИВНИХ СИСТЕМ БЕЗПЕКИ НАД СТАТИЧНИМИ ПРАВИЛАМИ ФІЛЬТРАЦІЇ В ІОТ-МЕРЕЖАХ

UDC 004.056:004.8:004.7

R. Mindzibrodskyi

ADVANTAGES OF ADAPTIVE SECURITY SYSTEMS OVER STATIC FILTERING RULES IN IOT NETWORKS

Попри активну інтеграцію IoT у побутовий простір сучасні розумні будинки часто покладаються на застарілі механізми статичної фільтрації. Традиційні методи, такі як списки контролю доступу, виявляють критичну неефективність у динамічному середовищі через високу гетерогенність пристроїв – від холодильників до камер. Статичні правила вимагають постійного ручного налаштування під нові вектори атак, що стає неможливим при зростанні кількості гаджетів. Як зазначають дослідники, статичні механізми безсилі проти атак «нульового дня», оскільки оперують лише відомими сигнатурами, залишаючи систему вразливою до нових модифікацій шкідливого ПЗ [1].

Адаптивні системи безпеки змінюють парадигму з «заборони відомого поганого» на «дозвіл лише нормального», спираючись на поведінковий аналіз та машинне навчання. Система будує унікальний профіль роботи для кожного пристрою. Наприклад, якщо термостат раптово починає передавати гігабайти даних на невідому адресу, алгоритм ідентифікує це як аномалію, навіть якщо порт формально відкритий. Такий підхід дозволяє автоматично ізолювати скомпрометовані вузли ще до того, як вони стануть частиною ботнету, що є критичним для цілісності мережі [2].

Враховуючи обмежену обчислювальну потужність IoT-датчиків, архітектура адаптивного захисту ефективно реалізується на рівні шлюзу або граничних обчислень. Це дозволяє аналізувати трафік централізовано, не перевантажуючи кінцеві пристрої. Використання легковагових алгоритмів на периферії знижує затримку реакції та забезпечує роботу системи навіть без зв'язку з хмарою. Натомість жорсткі статичні правила часто призводять до помилкових спрацьовувань, блокуючи легітимні оновлення прошивок або нові функції через прив'язку до фіксованих параметрів.

Окрім того, адаптивні механізми вирішують проблему масштабованості, адже пристрої в розумному домі постійно змінюють свої параметри. Завдяки здатності самостійно оновлювати моделі загроз системи глибокого навчання реалізують принцип імунної реакції та автоматично масштабують локальний захист на всю мережу. Дослідження підтверджують, що адаптивні моделі демонструють точність виявлення понад 95% у складних сценаріях, тоді як статичні методи часто пропускають замаскований трафік.

Підсумовуючи, перехід від статичних правил фільтрації до адаптивних систем безпеки є не просто покращенням, а необхідною умовою. Статичні методи, що були ефективні в епоху класичних комп'ютерних мереж, не витримують перевірки реаліями розумних будинків через їх динамічність та різноманітність. Адаптивні системи забезпечують необхідну гнучкість, автоматизацію та проактивний захист, дозволяючи нівелювати ризики ще на етапі аномальної поведінки, а не за фактом інциденту. Подальший розвиток цієї сфери нерозривно пов'язаний з удосконаленням алгоритмів штучного інтелекту, які зроблять безпеку «невидимою» для користувача, але непроникною для злоумисників.

Література

1. Al-Garadi, M. A., et al. (2020). A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Communications Surveys & Tutorials*, 22(3), pp. 1646–1685.
2. Xiao, L., Wan, X., Lu, X., Zhang, Y., & Wu, D. (2018). IoT Security Techniques Based on Machine Learning: How Do IoT Devices Use AI to Enhance Security? *IEEE Signal Processing Magazine*, 35(5), pp. 41–49.



**Збірник тез доповідей
VI міжнародної науково-практичної
конференції учених та студентів
"ЦИФРОВА ЕКОНОМІКА ЯК ФАКТОР
ІННОВАЦІЙ ТА СТАЛОГО РОЗВИТКУ
СУСПІЛЬСТВА"**

**Proceedings
of the 6th International Scientific and
Practical Conference
“DIGITAL ECONOMY AS A FACTOR
OF INNOVATIVE AND SUSTAINABLE
DEVELOPMENT OF SOCIETY”**

9–10 грудня 2025 року

ТЕРНОПІЛЬ, УКРАЇНА

3. Жибер, Т.В. (2025). Розвиток логістичних хабів як складова митної політики. *Review of transport economics and management*, (13(29), 5–14. <https://doi.org/10.15802/rtem2025/328071>

4. Штельмашук, М. (2024). Цифровізація та автоматизація логістичних процесів: сучасний стан та перспективи. *Економіка та суспільство*, (68). <https://doi.org/10.32782/2524-0072/2024-68-193>

УДК 004.89:658.7

Роман МІНДЗІБРОЦЬКИЙ, здобувач вищої освіти

Андрій ЛУЦКІВ, канд. техн. наук, доц

Тернопільський національний технічний університет ім. І. Пулюя, Україна

ШТУЧНИЙ ІНТЕЛЕКТ У ЛОГІСТИЦІ: ВІД ПРОГНОЗУВАННЯ ПОПИТУ ДО АВТОНОМНИХ СИСТЕМ

Roman MINDZIBRODSKYI, higher education applicant

Andriy LUTSKIV, Ph.D., Associate Prof.

Ternopil Ivan Puluj National Technical University, Ukraine

ARTIFICIAL INTELLIGENCE IN LOGISTICS: FROM DEMAND FORECASTING TO AUTONOMOUS SYSTEMS

Інтеграція штучного інтелекту (AI) докорінно змінює логістику, переводячи її від систем, що спиралися на історичні дані та експертні оцінки, до динамічних екосистем, здатних самостійно прогнозувати, оптимізувати й реагувати на зміни у реальному часі. Ця трансформація охоплює все – від аналітики попиту до автономних транспортних і складських систем, але супроводжується значними викликами як технічними, так і організаційними.

Традиційні методи прогнозування, такі як ARIMA чи експоненціальне згладжування, виявили свою неефективність у світі з високою турбулентністю попиту та глобальними збоями постачання [2, 3]. Після пандемії COVID-19 компанії почали масово інвестувати у гнучкі AI-рішення. Глобальний ринок таких технологій у логістиці оцінюється у понад 12 млрд доларів [6] із прогнозом стабільного зростання понад 15% щорічно [4].

AI-моделі прогнозування демонструють суттєву перевагу над класичними підходами: середня абсолютна процентна помилка зменшується з 22,7% до близько 12% [4]. Глибокі нейронні мережі, зокрема LSTM і гібридні RNN-CNN, здатні враховувати довгострокові тренди та короткострокові аномалії завдяки обробці великих обсягів різномірних даних – від соціальних індикаторів до транзакцій у реальному часі [2]. Це забезпечує точні локалізовані прогнози й адаптивне планування запасів. Walmart повідомив про зниження витрат на утримання складів на понад мільярд доларів на рік завдяки AI-аналітиці [4]. Amazon оптимізує понад 100 млн комбінацій “продукт–локація” щодня, досягаючи точності доставки понад 96% [4].

Аналітика попиту тісно поєднується з автономними системами логістики. Нове покоління автономних мобільних роботів (AMRs) і автоматизованих систем управління складами (ASRS), оснащених LiDAR і комп'ютерним зором, здатне самостійно орієнтуватися в динамічному середовищі, планувати маршрути й виконувати складські операції без участі людини [1, 8]. Доставка вантажів дронами і безпілотними транспортними системами скорочує час перевезень у кілька разів і знижує вуглецевий

слід. Такі компанії, як UPS і DHL, активно впроваджують AI-платформи для оптимізації логістичних мереж: алгоритми UPS ORION щорічно економлять мільйони миль пробігу [4], а DHL зменшив витрати пального на 10–15%.

Завдяки цим новаціям компанії, що інтегрували AI, повідомляють про скорочення логістичних витрат у середньому на 15%, підвищення доступності запасів на 35% і покращення клієнтського сервісу на 60% [6]. Роботизовані склади прискорюють обробку замовлень до 30%, а системи контролю якості на основі комп'ютерного зору виявляють дефекти й запобігають поверненням продукції.

Однак поряд із перевагами зберігаються критичні бар'єри. Найсерйознішою проблемою є якість даних: навіть незначне погіршення їх повноти чи частоти оновлення знижує точність моделей на 15–20% [7]. Семантична неоднорідність, часові затримки й відсутність узгоджених форматів у різних ланках постачання гальмують ефективну інтеграцію. Крім того, системи машинного навчання часто функціонують як “чорні скриньки” – їх рішення складно пояснити навіть розробникам, що створює ризики юридичної та етичної відповідальності [3, 4]. Висока вартість цифрової інфраструктури й дефіцит фахівців також обмежують доступність технологій для малих компаній [1, 2, 6].

Успішна реалізація потенціалу AI у логістиці передбачає вирішення трьох ключових завдань: поліпшення управління якістю даних, підвищення прозорості алгоритмів через пояснювані AI-системи (XAI) та розвиток кадрів для підтримки складних аналітичних процесів. Перспективними напрямками є інтеграція блокчейну для підвищення надійності даних, а також поєднання AI з 5G-мережами та IoT для децентралізованого управління транспортом у реальному часі.

Таким чином, AI поступово перетворює логістику з операційної функції на гнучку, інтелектуальну систему, де прогноз, оптимізація та автономність зливаються в єдину архітектуру управління. Компанії, які сьогодні інвестують у пояснювані, прозорі й адаптивні моделі, отримують стійку конкурентну перевагу у цифровій економіці завтрашнього дня.

Література

1. Suduk N., Herasymovych I. Застосування штучного інтелекту у виробничій логістиці. // Економіка та суспільство. 2025. № 73. URL: <https://economyandsociety.in.ua/index.php/journal/article/view/5865> (дата звернення: 09.11.2025).
2. Kagalwala H. et al. The Role of AI and Machine Learning in Demand Forecasting. // ACR Journal. 2025. Vol. 2(1). С. 142–149. URL: https://www.researchgate.net/publication/389070197_Predictive_Analytics_in_Supply_Chain_Management_The_Role_of_AI_and_Machine_Learning_in_Demand_Forecasting (дата звернення: 09.11.2025).
3. Allam H. et al. Enhancing supply chain efficiency through AI-driven demand forecasting. // Issues in Information Systems. 2025. Vol. 26(4). С. 65–77. URL: https://iacis.org/iis/2025/4_iis_2025_65-77.pdf (дата звернення: 09.11.2025).
4. Maddala S. K. 7 ways AI is revolutionizing supply chain forecasting and optimization. // World Journal of Advanced Engineering Technology and Sciences. 2025. Vol. 15(2). С. 680–686. URL: https://journalwjaets.com/sites/default/files/fulltext_pdf/WJAETS-2025-0584.pdf (дата звернення: 09.11.2025).

5.Czaban T. Supply Chain Forecasting: How to Win with Data and AI. // GoodData Blog. 19.05.2025. URL: <https://www.gooddata.com/blog/supply-chain-forecasting-how-to-win-with-data-and-ai/> (дата звернення: 09.11.2025).

6. Чапліч С. Штучний інтелект у логістиці та як його використовують компанії. // ProIT. 08.10.2024. URL: <https://proit.ua/shtuchnii-inteliiekt-v-loghistitsi-ta-iak-iogho-vikoristovuiut-svitovi-i-ukrayinski-kompaniyi/> (дата звернення: 09.11.2025).

7.Zhu C., Xin J., Trinh T. K. Data Quality Challenges for AI Implementation in Supply Chain Management. // Proceedings Series. 2025. Vol. 2. С. 25–40. URL: <https://pinnaclepubs.com/index.php/PAPPS/article/download/101/102> (дата звернення: 09.11.2025).

8.World Economic Forum. Autonomous Vehicles: Timeline and Roadmap Ahead. // World Economic Forum. 24.04.2025. URL: <https://www.weforum.org/publications/autonomous-vehicles-timeline-and-roadmap-ahead/> (дата звернення: 09.11.2025).

УДК 004:330.46(477)

Ярослав СИРОТИНСЬКИЙ, здобувач вищої освіти

Тернопільський національний технічний університет ім. І. Пулюя, Україна

Науковий керівник: Руслан ЖАРОВСЬКИЙ, канд. техн. наук, доц.

ХМАРНІ ТЕХНОЛОГІЇ ДЛЯ УПРАВЛІННЯ ЛОГІСТИЧНИМИ ПРОЦЕСАМИ

Yaroslav SYROTYNSKYI, higher education applicant;

Ternopil Ivan Puluj National Technical University, Ukraine

Scientific supervisor Ruslan ZHAROVSKYI, PhD in Technical Sciences, Assoc. Prof.

CLOUD TECHNOLOGIES FOR LOGISTICS PROCESS MANAGEMENT

У сучасних умовах розвитку глобальної економіки логістика стає стратегічною складовою ефективного функціонування бізнесу. Підвищення складності логістичних ланцюгів, динамічність попиту та необхідність оперативного обміну інформацією зумовлюють потребу у впровадженні новітніх цифрових рішень. Одним із ключових напрямів цифровізації логістичних процесів є використання хмарних технологій.

Хмарні технології (Cloud Computing) — це модель надання обчислювальних ресурсів і програмних сервісів через мережу Інтернет, яка дозволяє користувачам отримувати доступ до даних і програм без необхідності володіння фізичною інфраструктурою. У контексті логістики це означає можливість централізованого управління ланцюгами постачання, контролю транспорту, управління складом та аналітики потоків у реальному час [1].

Найпоширенішими моделями використання хмарних технологій у логістиці є:

–SaaS (Software as a Service) – програмне забезпечення як послуга. Логістичні компанії можуть користуватись веб-додатками для відстеження вантажів, управління запасами чи планування маршрутів без встановлення програм локально.

–PaaS (Platform as a Service) – платформа як послуга, що дає змогу розробляти та інтегрувати власні логістичні модулі у корпоративні системи ERP або CRM.

–IaaS (Infrastructure as a Service) – інфраструктура як послуга, що дозволяє масштабувати обчислювальні ресурси для обробки великих обсягів логістичних даних [2].

Додаток Б

Системний промпт

DATA INTERPRETATION NOTES:

- "duration_sec": 0.0 is NOT an error. It indicates an instantaneous burst of traffic (< 0.01s).
- Treat "duration_sec": 0.0 combined with multiple packets as an EXTREME THREAT (Micro-burst flood).

You are an autonomous network security sentinel. Your goal is to protect the localhost while ensuring legitimate traffic is NOT blocked.

You will receive a JSON LIST of network flows. Analyze EACH flow in the list independently according to the rules.

DECISION LOGIC (FOLLOW IN STRICT ORDER):

1. SAFE TRAFFIC CHECK (HIGHEST PRIORITY):

- IF avg_packet_size > 800 bytes AND duration > 0.5s (likely Video/Download/FileTransfer):

-> ACTION: Call `log\_safe\_traffic`. Do NOT block.

-> REASON: "Legitimate high bandwidth traffic".

- Note: If packets are large (>800) BUT happen instantly (duration < 0.1s), treat as FLOOD.

2. CRITICAL THREATS (DoS/Flood):

- IF (total_packets > 1000) OR (duration < 1.0s AND total_packets > 10):

-> ACTION: You MUST call `block\_ip`.

-> REASON: "High volume flood detected".

3. SUSPICIOUS ACTIVITY (Scanning):

- IF dst_ports_count > 3 OR probing sensitive ports (23, 22, 3389, 445):

-> ACTION: Call `get\_ip\_info` to investigate origin.

4. INTELLIGENT OVERSIGHT (For Gray Areas):

- IF metrics are slightly below thresholds (e.g., 800-999 packets) BUT target sensitive ports (22, 3389, 23):

-> ACTION: Call `block\_ip`.

-> REASON: "Suspicious borderline activity on sensitive port".

5. GEO-BLOCKING POLICY:

- IF `get\_ip\_info` returns a high-risk region (e.g., China, North Korea, Iran)

AND the traffic involves scanning or sensitive ports:

-> ACTION: Call `block\_ip`.

-> REASON: "Scanning activity from high-risk geo-location".

OUTPUT FORMAT:

You MUST generate a sequence of Tool Calls.

If a flow requires action (block or log), call the tool.

If a flow is ambiguous or irrelevant, you may skip it.

Process the entire list.

EXAMPLES OF CORRECT BEHAVIOR:

Input: {"src_ip": "1.1.1.1", "total_packets": 5000, "duration": 0.5}

Reasoning: High packets + short duration = Flood.

Tool Call: block_ip("1.1.1.1", "High volume flood detected")

Input: {"src_ip": "8.8.8.8", "dst_ports_count": 12}

Reasoning: Many ports indicate scanning, but verify IP first.

```
Tool Call: get_ip_info("8.8.8.8")
Input: {"src_ip": "10.0.0.5", "avg_packet_size": 1200, "duration": 5.0}
Reasoning: Large packets over long duration indicate safe transfer.
Tool Call: log_safe_traffic("10.0.0.5", "Legitimate high bandwidth
traffic")
```