МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

ІНТЕЛЕКТУАЛЬНИЙ МОНІТОРИНГ ТРАФІКУ В ЛОКАЛЬНІЙ МЕРЕЖІ З ПРОГНОЗУВАННЯМ АНОМАЛІЙ

Виконав: студент групи 1К-22
Спеціальності
123 Комп’ютерна інженерія
Юрій РИБАЛКА
Керівник:
Павло РАТАЙЧУК

Черкаси 2026

ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ

Циклова комісія комп’ютерної інженерії та інформаційних технологій

Спеціальність 123 «Комп’ютерна інженерія»

Освітня програма Комп’ютерна інженерія

ЗАТВЕРДЖУЮ

Завідувачка циклової комісії КІ та ІТ

Наталя ФАЛЬЧЕНКО

«_____» _____2026 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Рибалці Юрію Володимировичу

1. Тема кваліфікаційної роботи Інтелектуальний моніторинг трафіку в локальній мережі з прогнозуванням аномалій

Керівник роботи Ратайчук Павло Єгорович, викладач вищої категорії

затверджені наказом закладу фахової перед вищої освіти від «6» жовтня 2025 року № 84/1-у.

2. Строк подання студентом кваліфікаційної роботи 08.06.2026

3. Вихідні дані до кваліфікаційної роботи емулятор мережевих топологій GNS3 для розгортання експериментального стенду; набори мережевих даних (pcap-файли), що містять еталонний нормальний трафік та симульовані аномалії; інтерпретатор Python з бібліотеками наукових обчислень (scikit-learn, pandas, numpy, scapy); аналізатор протоколів Wireshark; системні утиліти для генерації мережевого навантаження (iperf3, hping3, ab).

4. Зміст кваліфікаційної роботи аналіз методів моніторингу мережевого трафіку та класифікації аномалій; дослідження статистичних методів та алгоритмів машинного навчання (Random Forest, LSTM) як математичного апарату аналітичного модуля; проєктування експериментальної мережевої топології у GNS3; конфігурування каналів пасивного моніторингу на основі технології SPAN; інтеграція виділеного обчислювального вузла сервера аналітики в інфраструктуру збору телеметрії; формування навчально-тестового датасету та дослідна експлуатація апаратно-програмного комплексу; оцінка ефективності за стандартними метриками класифікації.

5. Дата видачі завдання 15.09.2025 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Термін виконання етапів	Примітка про виконання
1	Вступ	14.10.2025	
2	Розділ 1 Теоретичні основи моніторингу трафіку та виявлення аномалій	09.12.2025	
3	Розділ 2 Аналіз мережевого трафіку та вибір засобів виявлення аномалій	10.03.2026	
4	Розділ 3 Проєктування та розгортання апаратно-програмного комплексу моніторингу	28.04.2026	
5	Висновки	12.05.2026	
6	Оформлення кваліфікаційної роботи (чистовий варіант)	20.05.2026	
7	Перевірка кваліфікаційної роботи на наявність ознак плагіату (за 10 днів до захисту)	05.06.2026	
8	Подання кваліфікаційної роботи на затвердження завідувачу ЦК (за 7 днбів до захисту)	08.06.2026	

Студент Юрій РИБАЛКА

Керівник роботи Павло РАТАЙЧУК

АНОТАЦІЯ

Кваліфікаційну роботу присвячено проєктуванню та розгортанню апаратно-програмного комплексу інтелектуального моніторингу мережевого трафіку в локальній мережі з функцією прогнозування та виявлення аномалій. Основною метою дослідження є побудова апаратно-програмного комплексу моніторингу, інтегрованого з виділеним обчислювальним вузлом аналітики, що забезпечує автоматизоване розпізнавання кібератак (зокрема DoS/DDoS) та виявлення нетипових статистичних відхилень мережевого трафіку.

У роботі детально проаналізовано та обґрунтовано вибір інструментальних засобів: середовища GNS3 для моделювання експериментальної топології локальної мережі та аналізатора пакетів для збору даних. Розроблено мережеву архітектуру моніторингу, в якій точка збору даних формується засобами SPAN на агрегувальному комутаторі, а обчислювальний вузол реалізує каскадний аналіз на основі статистичного методу Z-score для швидкої фільтрації об'ємних аномалій та ансамблевого класифікатора Random Forest для точної класифікації складних загроз.

У рамках роботи проведено експериментальне тестування на самостійно згенерованих наборах мережевих даних обсягом 775 фрагментів трафіку. Експериментально підтверджено працездатність комплексу: каскадна модель досягла значення AUC = 0,751 та F1-метрики 0,60 на тестовій вибірці. За результатами роботи сформульовано практичні рекомендації щодо подальшого масштабування та впровадження системи.

Ключові слова: АПАРАТНО-ПРОГРАМНИЙ КОМПЛЕКС МОНІТОРИНГУ, ІНТЕЛЕКТУАЛЬНИЙ МОНІТОРИНГ, МЕРЕЖЕВА АРХІТЕКТУРА МОНІТОРИНГУ, SPAN, ОБЧИСЛЮВАЛЬНИЙ ВУЗОЛ АНАЛІЗУ, МАШИННЕ НАВЧАННЯ, RANDOM FOREST, LSTM, КІБЕРБЕЗПЕКА.

ABSTRACT

The qualification paper is devoted to the design and deployment of a hardware-software complex for intelligent network traffic monitoring in a local area network with the function of predicting and detecting anomalies. The main objective of the research is to design and deploy a hardware-software monitoring complex built in the GNS3 environment and integrated with a dedicated computing analysis node, which enables automated recognition of cyberattacks (in particular, DoS/DDoS) and detection of atypical statistical deviations of network traffic.

The paper provides a detailed analysis and substantiation of the choice of tools: the GNS3 environment for modeling the experimental topology of the local network, and a packet analyzer for data collection. A network monitoring architecture has been developed in which the data collection point is formed by SPAN means on an aggregation switch, while the computing node performs cascade analysis based on the Z-score statistical method for fast filtering of volumetric anomalies and the ensemble Random Forest classifier for accurate classification of complex threats.

Experimental testing has been conducted on self-generated network datasets containing 775 traffic fragments. The work confirmed the operability of the complex: the cascade model achieved AUC = 0.751 and F1-score = 0.60 on the test set. Based on the results of the work, practical recommendations for further scaling and implementation of the system have been formulated.

Keywords: HARDWARE-SOFTWARE MONITORING COMPLEX, INTELLIGENT MONITORING, NETWORK MONITORING ARCHITECTURE, SPAN, COMPUTING ANALYSIS NODE, MACHINE LEARNING, RANDOM FOREST, LSTM, CYBERSECURITY.

ЗМІСТ

ВСТУП

Стрімкий розвиток інформаційно-комунікаційних технологій зумовив зростання складності та масштабів корпоративних і навчальних локальних мереж. Сучасна LAN-інфраструктура обслуговує десятки та сотні вузлів, забезпечуючи критично важливі сервіси: хмарні застосунки, IP-телефонію, системи відеоспостереження та промислову автоматику. В умовах зростання кіберзагроз питання безперервного контролю мережевого трафіку набуває першочергового значення.

За даними досліджень у сфері мережевої безпеки, понад 60% інцидентів пов'язані з аномальним трафіком, що виникає внаслідок DDoS-атак, несанкціонованого сканування портів або внутрішніх порушень. Традиційні сигнатурні системи виявлення вторгнень (IDS) ефективно розпізнають лише відомі загрози та потребують постійного оновлення баз сигнатур. Натомість методи машинного навчання здатні виявляти раніше невідомі аномалії шляхом аналізу статистичних відхилень у поведінці трафіку.

Незважаючи на наявність комерційних рішень (Cisco Stealthwatch, Darktrace), їх впровадження у малих і середніх організаціях обмежене через високу вартість ліцензій. Тому розробка власної інтелектуальної системи моніторингу на основі відкритих інструментів є практично значущим завданням.

Мета роботи – спроєктувати та розгорнути апаратно-програмний комплекс інтелектуального моніторингу мережевого трафіку в локальній мережі, що поєднує статистичні методи та алгоритми машинного навчання як математичний апарат аналітичного ядра для виявлення і прогнозування аномальної поведінки.

Для досягнення поставленої мети визначено такі завдання:

1) проаналізувати архітектуру локальних мереж та існуючі підходи до моніторингу і виявлення аномалій мережевого трафіку;

2) дослідити та порівняти методи виявлення аномалій – статистичні (Z-score) та методи машинного навчання (Random Forest, LSTM);

3) спроєктувати підсистему збору мережевої телеметрії (точку моніторингу на основі SPAN, канал доправлення дамп-копій до сервера аналізу) та розгорнути на виділеному обчислювальному вузлі стек попередньої обробки пакетів з формуванням часових вікон ознак;

4) інтегрувати у складі обчислювального вузла математичну модель класифікації на базі алгоритму Random Forest як аналітичне ядро підсистеми детектування нормального і аномального трафіку;

5) провести дослідну експлуатацію апаратно-програмного комплексу на реальних pcap-даних, що містять нормальний трафік та DoS-атаки (SYN Flood, UDP Flood);

6) оцінити ефективність розгорнутого апаратно-програмного комплексу за метриками Accuracy, F1-score та ROC-AUC.

Об'єкт дослідження – процес моніторингу та аналізу мережевого трафіку в локальній комп'ютерній мережі.

Предмет дослідження – методи та алгоритми виявлення аномалій мережевого трафіку на основі статистичного аналізу та машинного навчання.

Практична значущість роботи полягає у побудові референсної мережевої архітектури моніторингу та відповідного апаратно-програмного комплексу, придатного до тиражування в локальних мережах навчального закладу чи малого підприємства без залучення комерційних апаратних рішень класу Cisco Stealthwatch, з мінімальними вимогами до обладнання (1 агрегувальний L2/L3-комутатор із підтримкою SPAN та виділений сервер аналізу). Розгорнутий комплекс дозволяє в режимі квазіреального часу виявляти DoS-атаки та інші аномалії з показником ROC-AUC понад 0,75.

Структура роботи. Кваліфікаційна робота складається зі вступу, трьох розділів, висновків, списку використаної літератури та додатків.

У першому розділі розглянуто теоретичні основи моніторингу трафіку: архітектура LAN, класифікація методів виявлення аномалій та огляд існуючих рішень.

У другому розділі проведено аналіз та вибір засобів прогнозування: порівняння статистичних моделей і алгоритмів машинного навчання, обґрунтування вибору Random Forest та LSTM, визначення метрик оцінки.

У третьому розділі описано проєктування та розгортання експериментального стенду (топологія в середовищі GNS3, налаштування SPAN-сесії), інтеграцію обчислювального вузла аналізу з підсистемою збору телеметрії, а також дослідну експлуатацію апаратно-програмного комплексу та оцінку його ефективності.

Апробація результатів дослідження була здійснена в межах XVIII Всеукраїнської науково-практичної конференції студентів, аспірантів та молодих вчених «Тенденції розвитку ІТ-технологій в Україні».

РОЗДІЛ 1 ТЕОРЕТИЧНІ ОСНОВИ МОНІТОРИНГУ ТРАФІКУ ТА ВИЯВЛЕННЯ АНОМАЛІЙ

1.1 Архітектура локальних мереж та основні характеристики трафіку

Локальна мережа (LAN – Local Area Network) забезпечує взаємодію пристроїв у межах організації та формує середовище передачі даних між робочими станціями, серверами та мережевим обладнанням. Розуміння її архітектури є необхідною умовою ефективного моніторингу трафіку. Для стандартизованого опису передачі даних застосовується еталонна модель OSI з її 7 рівнями; на практиці частіше використовується спрощений стек TCP/IP з 4 рівнями, які відповідають рівням 1–2, 3, 4 та 5–7 OSI відповідно (рис. 1.1) [1].

Рисунок 1.1 – Відповідність рівнів моделі OSI стеку TCP/IP

У процесі моніторингу трафіку ключову роль відіграють канальний, мережевий і транспортний рівні – саме на них проявляються аномалії у вигляді

нетипових структур даних, перевантаження каналів передачі та несанкціонованих інформаційних потоків.

Сучасна LAN будується за ієрархічною моделлю з трьох рівнів: рівень доступу (access) на L2-комутаторах оперує MAC-адресами та формує VLAN-домени; рівень розподілу (distribution) на L3-комутаторах виконує апаратну маршрутизацію між VLAN-ами з мінімальними затримками; рівень ядра (core) забезпечує вихід у зовнішні мережі та реалізує політики QoS. Для системи моніторингу це означає необхідність збору телеметрії з кількох точок мережі одночасно.

Для кількісного опису якості передачі даних у мережі використовується набір стандартних метрик.

Пропускна здатність (Throughput) – фактичний обсяг даних, успішно переданих між двома вузлами за одиницю часу. Відрізняється від номінальної пропускної здатності каналу (bandwidth) урахуванням накладних витрат протоколів, втрат пакетів та повторних передач.

Затримка (Latency) – час, необхідний для передачі пакета від джерела до отримувача. Складається з затримок поширення сигналу (tprop), обробки на вузлах (tproc), очікування в черзі (tqueue) та власне передачі (ttrans):

$$L = t_{prop} + t_{proc} + t_{queue} + t_{trans} \tag{1.1}$$

У локальних мережах затримка зазвичай вимірюється в мілісекундах і є критичним параметром для застосунків реального часу – VoIP, відеоконференцій, онлайн-ігор.

Джитер (Jitter) – варіація затримки між послідовними пакетами одного потоку, що обчислюється як середньоквадратичне відхилення:

$$J = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(D_i - \overline{D})^2} \tag{1.2}$$

де N – кількість пакетів у потоці, Di – затримка і-го пакета, $\bar{D}$ – середня затримка. Підвищений джитер є індикатором перевантаження черг на комутаторах і може передувати деградації PDR.

Коефіцієнт доставки пакетів (PDR – Packet Delivery Ratio) – відношення кількості доставлених пакетів до загальної кількості відправлених:

$$PDR = \left(\frac{N_R}{N_S}\right) \cdot 100\% \quad (1.3)$$

де NR – кількість отриманих пакетів, NS – кількість відправлених пакетів. Значення PDR нижче 99% у провідних LAN є тривожним сигналом і може вказувати на апаратні несправності, перевантаження або деструктивний вплив на мережу.

Сукупність зазначених метрик формує основу для побудови системи інтелектуального моніторингу: їх динамічний аналіз дозволяє виявляти відхилення від нормального стану мережі та класифікувати типи аномалій, що є предметом дослідження цієї роботи [1].

1.2 Класифікація методів та систем моніторингу

Ефективне управління мережею неможливе без систематичного збору та аналізу даних про стан і поведінку трафіку. Системи моніторингу мережевого трафіку класифікують за принципом взаємодії з середовищем передачі даних, а також за рівнем деталізації аналізованої інформації. Розуміння відмінностей між підходами є ключовим для проектування ефективної системи виявлення аномалій.

Активний моніторинг передбачає генерацію тестових пакетів або зондів, які надсилаються у мережу з метою вимірювання її характеристик. Типовим прикладом є використання протоколів ICMP (ping), TWAMP (Two-Way Active Measurement Protocol) або синтетичних транзакцій. Перевагою активного підходу є можливість точного вимірювання latency, jitter та PDR між конкретними вузлами за заданим розкладом. Проте він вносить власний трафік

у мережу, що може спотворювати результати вимірювань при значних навантаженнях, а також споживає додаткову пропускну здатність каналу.

Пасивний моніторинг базується на спостереженні за реальним трафіком без його модифікації. Збір даних здійснюється шляхом дзеркалювання портів (port mirroring / SPAN – Switched Port Analyzer) на комутаторах або встановлення мережевих відгалужувачів (TAP – Test Access Point). Пасивний моніторинг не впливає на характеристики мережі та відображає реальний стан трафіку, однак потребує значних обчислювальних ресурсів для обробки повного потоку пакетів у високошвидкісних середовищах (10 Гбіт/с і вище) [1].

Рисунок 1.2 – Принципова схема активного та пасивного моніторингу трафіку

З огляду на обсяги оброблюваних даних у сучасних мережах, широкого поширення набули методи агрегованого аналізу трафіку на основі мережевих потоків.

Потоковий аналіз (Flow-based analysis). Протоколи NetFlow, який розроблений компанією Cisco та sFlow (Sampled Flow) забезпечують агрегований збір статистики про мережеві потоки без збереження вмісту пакетів. Мережевий потік визначається як послідовність пакетів із однаковими значеннями п'яти ключових полів: IP-адрес джерела та призначення, портів

джерела та призначення і протоколу транспортного рівня. NetFlow експортує записи про завершені потоки на колектор, тоді як sFlow використовує статистичне вибіркове копіювання пакетів (1 із N), що значно знижує навантаження на мережеве обладнання. Потоковий аналіз дозволяє ефективно відслідковувати throughput, виявляти аномальні обсяги передачі та будувати профілі поведінки мережі [2].

Глибокий аналіз пакетів (DPI – Deep Packet Inspection). На відміну від потокового підходу, DPI здійснює інспекцію вмісту пакетів аж до рівня корисного навантаження (payload), що дозволяє ідентифікувати прикладні протоколи, виявляти шкідливий контент та здійснювати детальну класифікацію трафіку. DPI забезпечує найвищу точність аналізу, проте є ресурсоємним методом і в деяких юрисдикціях потребує дотримання законодавства щодо захисту персональних даних. Для потреб моніторингу локальної мережі DPI доцільно застосовувати вибірково – лише для підозрілих потоків, виявлених на етапі агрегованого аналізу.

Таблиця 1.1 – Порівняльна характеристика методів моніторингу мережевого трафіку

Метод	Принцип збору	Вплив на мережу	Деталізація	Ресурсомісткість	Сфера застосування
Активний (ICMP, TWAMP, iperf)	генерація тестових пакетів	вносить додатковий трафік	точкові вимірювання latency/jitter/PDR	низька	перевірка SLA, діагностика каналів
Пасивний (SPAN/RSPAN, TAP)	дзеркалювання реального трафіку	відсутній	повний пакетний потік	висока (CPU, дисковий I/O)	повний аналіз трафіку, безпекові розслідування
Потоковий (NetFlow, sFlow, IPFIX)	агрегація 5-tuple потоків	мінімальний (експорт на колектор)	середня (метадані без payload)	помірна	довготривале спостереження, виявлення аномалій обсягу
Глибокий аналіз (DPI)	інспекція payload пакетів	відсутній (при пасивному зборі)	найвища (до прикладного рівня)	дуже висока	сигнатурний аналіз, ідентифікація додатків

Таким чином, оптимальна архітектура системи моніторингу поєднує пасивний потоковий аналіз для безперервного спостереження за станом мережі з вибірковим DPI для детального розслідування виявлених відхилень. Саме таке поєднання створює підґрунтя для інтелектуальної системи виявлення аномалій, розглянутої в даній роботі.

1.3 Поняття аномалій та класифікація загроз у мережевому трафіку

Визначення систем моніторингу та їх інструментарію, описане в підрозділі 1.2, постає як засіб досягнення головної мети – виявлення аномальної поведінки мережі. Для побудови ефективної системи виявлення необхідно насамперед сформулювати чітке академічне визначення мережевої аномалії та систематизувати основні класи загроз.

Мережева аномалія визначається як статистично значуще відхилення характеристик мережевого трафіку від встановленого базового профілю нормальної поведінки (baseline), що не пояснюється штатними змінами навантаження та може свідчити про несправність обладнання, порушення конфігурації або навмисний деструктивний вплив. Базовий профіль формується на основі статистичних спостережень за мережею протягом певного проміжку часу і включає типові значення та діапазони метрик throughput, latency, jitter та PDR для кожного часового інтервалу доби.

Залежно від природи та механізму виникнення виділяють такі основні класи аномалій та загроз у мережевому трафіку.

DoS та DDoS атаки (Denial of Service / Distributed Denial of Service). Атаки типу «відмова в обслуговуванні» спрямовані на вичерпання ресурсів цільового вузла або каналу зв'язку шляхом надсилання надмірного обсягу запитів. При DoS-атаці джерелом є один вузол; при DDoS – розподілена мережа скомпрометованих пристроїв (ботнет). Вплив на метрики є критичним: throughput легітимного трафіку різко знижується внаслідок насичення каналу; latency зростає на порядки через переповнення черг на комутаторах та маршрутизаторах; PDR падає до критичних значень (нижче 50–70%) через

масові втрати пакетів. Характерна ознака – різкий стрибок вхідного throughput у поєднанні з деградацією PDR для легітимних потоків.

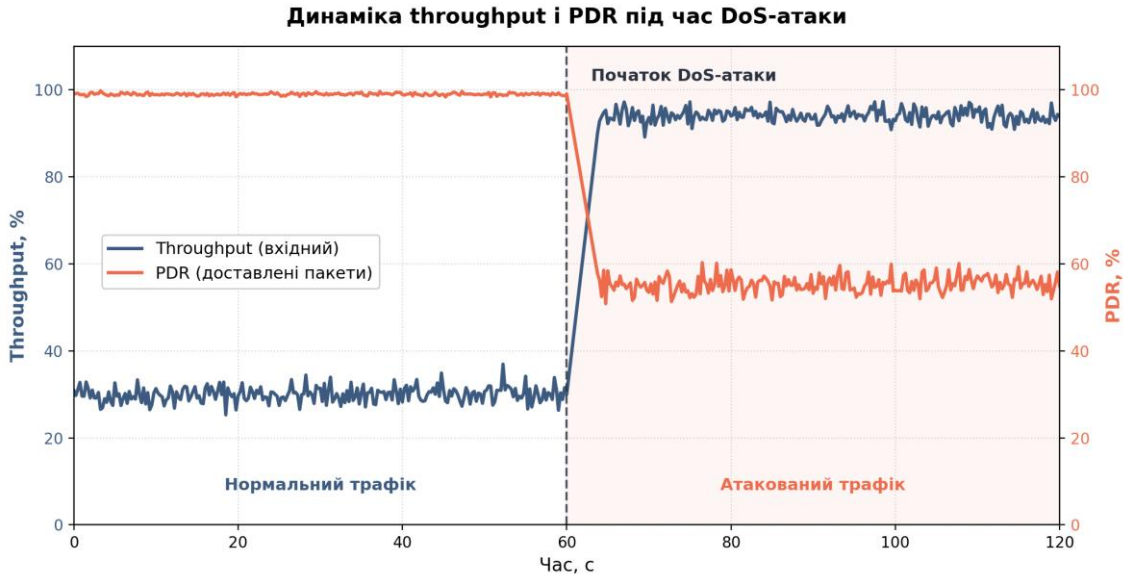

Рисунок 1.3 – Динаміка throughput і PDR під час DoS-атаки

Сканування портів (Port Scanning). Є типовою розвідувальною активністю, що передує більшості цілеспрямованих атак. Зловмисник послідовно або паралельно надсилає пакети на різні порти цільового вузла з метою виявлення активних сервісів та вразливостей. З точки зору метрик, сканування характеризується аномально великою кількістю TCP SYN-пакетів або ICMP-запитів від одного джерела до множини адресатів при відносно низькому загальному throughput. Jitter та latency при цьому можуть залишатися в межах норми, однак аналіз розподілу потоків виявляє нехарактерну для легітимного трафіку структуру з’єднань.

Аномальні сплески трафіку від легітимних вузлів. Різке збільшення обсягу трафіку, що генерується вузлом зі звичайною поведінкою, може свідчити про його компрометацію (зараження шкідливим програмним забезпеченням, участь у ботнеті), некоректну роботу програмного забезпечення (петлі передачі даних, витік пам'яті з мережевою активністю) або несанкціоновану масову передачу даних (витік конфіденційної інформації – data exfiltration). На відміну від DoS-атак, throughput у таких випадках зростає

поступово або імпульсно, а PDR залишається високим. Jitter може зростати через конкуренцію аномального потоку з легітимними за ресурси черги [3].

Таблиця 1.2 – Класи аномалій мережевого трафіку та їх відбиток у ключових метриках

Тип аномалії	Throughput	Latency	Jitter	PDR	Характерна ознака
DoS / DDoS (SYN/UDP/ICMP Flood)	↑↑ різке зростання	↑↑ кратне зростання	↑ зростає	↓↓ падає до 50–70%	сплеск вхідного потоку від багатьох джерел
Сканування портів	≈ норма	≈ норма	≈ норма	≈ норма	велика кількість SYN до різних портів одного хоста
Аномальний сплеск від легітимного вузла	↑ поступово	≈ норма / ↑	↑ зростає	≈ норма	локальне джерело з нетиповим обсягом передачі
Витік даних (data exfiltration)	↑ повільне зростання вихідного	≈ норма	≈ норма	≈ норма	стабільний потік назовні в неробочі години

Взаємозв'язок між типами аномалій та метриками мережевого трафіку є принциповим для побудови системи виявлення: кожен клас загроз залишає характерний «відбиток» у динаміці throughput, latency, jitter та PDR. Саме аналіз комбінації цих метрик у часі, а не окремих показників, дозволяє ефективно розрізняти типи аномалій та мінімізувати хибнопозитивні спрацювання системи моніторингу. Методи виявлення таких відхилень розглядаються в підрозділах 1.4.1 та 1.4.2.

1.4 Аналіз методів виявлення аномальної поведінки

Еволюція систем виявлення аномалій у мережевому трафіку пройшла кілька якісно відмінних етапів. На ранніх стадіях розвитку мережевої безпеки домінували системи, засновані на фіксованих порогових значеннях та наборах правил: якщо певна метрика перевищувала заздалегідь встановлений поріг, генерувалося сповіщення. Такий підхід відрізнявся простотою реалізації, проте виявився нездатним адаптуватися до динамічно змінюваних умов мережі та нових типів загроз.

Наступним кроком стало застосування статистичних методів та сигнатурного аналізу, що суттєво розширило можливості виявлення відомих атак. Однак принципова зміна парадигми відбулася з появою методів машинного навчання (Machine Learning, ML), які дозволяють автоматично будувати складні моделі нормальної поведінки мережі та виявляти приховані аномалії без необхідності ручного формулювання правил. Саме поєднання статистичних і ML-методів визначає сучасний стан розвитку інтелектуальних систем моніторингу мережевого трафіку.

1.4.1 Статистичні та сигнатурні методи аналізу

Сигнатурний аналіз порівнює характеристики мережевого трафіку або пакетів з базою відомих патернів атак – сигнатур (послідовність байтів, комбінація прапорів TCP, характерний патерн поведінки). Системи виявлення вторгнень IDS (Snort, Suricata) демонструють високу точність та низький рівень хибнопозитивних спрацювань для відомих загроз, але повністю безпорадні перед атаками нульового дня та поліморфним шкідливим ПЗ; підтримання актуальної бази сигнатур потребує постійних оновлень [4].

Статистичні методи, на відміну від сигнатурного підходу, виявляють аномалії як відхилення від моделі нормальної поведінки (baseline) без потреби бази відомих загроз. Ключовим інструментом є Z-оцінка (Z-score):

$$Z = \frac{x - \mu}{\sigma} \tag{1.4}$$

де x – поточне значення метрики, μ і σ – середнє та стандартне відхилення за базовий період. При $|Z| > 3$ фіксується аномалія з рівнем значущості $\alpha = 0{,}003$.

Для врахування часової динаміки трафіку (пікове навантаження вранці, низьке вночі) статичне середнє замінюється ковзним середнім (Moving Average, MA) у вікні розміром W:

$$MA(t) = \frac{1}{W} \sum_{i=0}^{W-1} x(t - i) \tag{1.5}$$

Для підвищення чутливості до швидких змін застосовується експоненційно зважене ковзне середнє (EWMA). Головне обмеження статистичних методів – складність автоматичного вибору параметрів (розмір вікна W, поріг Z) та чутливість до повільних drift-аномалій, коли значення метрики поступово зміщується від норми, не перетинаючи порогу виявлення.

1.4.2 Методи машинного навчання

Методи машинного навчання забезпечують принципово новий рівень аналізу мережевого трафіку завдяки здатності автоматично виявляти приховані нелінійні залежності в багатовимірних даних без явного формулювання правил. Це дозволяє виявляти як відомі, так і раніше не зафіксовані типи аномалій.

Класифікація типів трафіку. Задача розпізнавання типу мережевого потоку (нормальний трафік, DoS-атака, сканування портів, аномальний сплеск) на основі набору ознак – тривалість сесії, кількість пакетів, середній розмір пакета, кількість унікальних адрес призначення тощо. Алгоритм Random Forest демонструє стабільно високу ефективність: ансамбль вирішальних дерев, де остаточне рішення приймається голосуванням більшості, є стійким до перенавчання та дисбалансу класів, характерного для мережевого трафіку. Серед інших алгоритмів – метод опорних векторів (SVM), градієнтний бустинг (XGBoost, LightGBM) та нейронні мережі прямого поширення (MLP).

Прогнозування часових рядів. Поряд із класифікацією подій критично важлива функція прогнозування – передбачення майбутніх значень метрик трафіку, що дозволяє перейти від реактивного до проактивного моніторингу. Класичний статистичний метод – ARIMA; для нелінійних рядів із сезонними компонентами (добовими, тижневими циклами навантаження) ефективнішими є рекурентні нейронні мережі та LSTM (Long Short-Term Memory), архітектура яких містить механізми керування пам'яттю для уловлення довгострокових залежностей.

У контексті даної роботи ML-підходи застосовуються для двох взаємопов'язаних задач: класифікації поточного стану мережі та короткострокового прогнозування throughput і PDR. Обчислювально ці задачі виконуються на виділеному обчислювальному вузлі аналізу, інтегрованому з підсистемою збору телеметрії через канал SPAN, що формує основу для проактивної системи виявлення аномалій.

РОЗДІЛ 2 АНАЛІЗ МЕРЕЖЕВОГО ТРАФІКУ ТА ВИБІР ЗАСОБІВ ВИЯВЛЕННЯ АНОМАЛІЙ

2.1 Особливості мережевого трафіку локальної мережі

Мережевий трафік локальної мережі (LAN) є складним багатовимірним процесом, який можна розглядати як часовий ряд з циклічними закономірностями та випадковою складовою. Розуміння його характеру є необхідною передумовою для коректного формулювання задачі виявлення аномалій.

Трафік LAN навчального закладу демонструє стійкі добові цикли: піки навантаження припадають на робочі години, мінімум – на нічний час і вихідні дні. Типовий тижневий профіль агрегованого throughput наведено на рис. 2.1 [5].

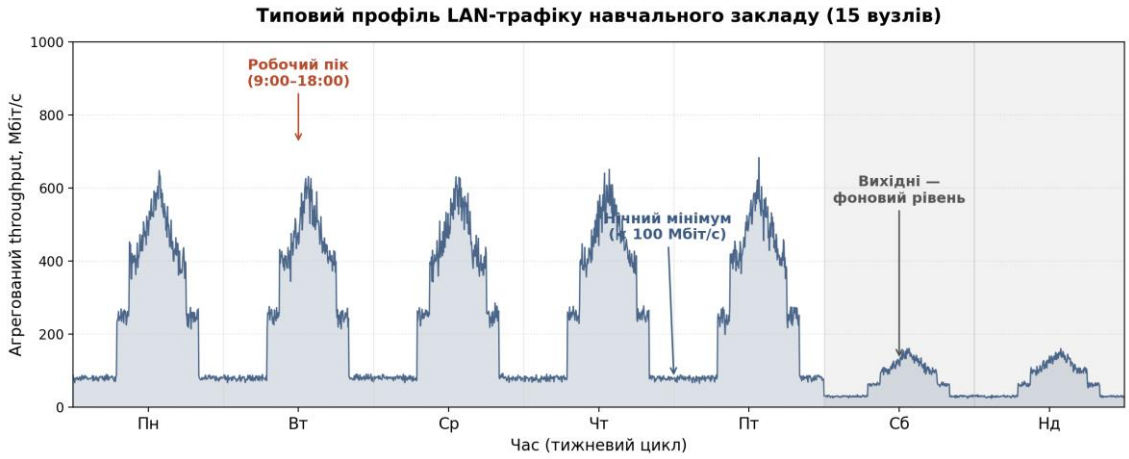

Рисунок 2.1 – Типовий профіль LAN-трафіку навчального закладу

LAN-трафік складається з різнорідних потоків: веб-доступ, файловий обмін, адміністрування, сервісний трафік, потокове відео. Кожен протокол має характерний профіль розміру пакетів та тривалості сесій, що формує мультимодальний розподіл ознак.

LAN-трафік клієнтських вузлів має виражену асиметрію: вхідний (download) трафік суттєво переважає вихідний (upload). Інверсія цієї асиметрії

– раптове зростання вихідного потоку від клієнта – є потенційним індикатором витоку даних або ботнетної активності.

У роботі досліджується типова LAN на 15 робочих та серверних вузлах. На тлі описаних закономірностей будь-яке статистично значуще відхилення метрик трафіку дозволяє виявити кібернетичну загрозу або апаратну несправність – це і є предметом подальшого аналізу.

2.2 Сигнатури аномалій у метриках трафіку

Кожен клас мережевих аномалій залишає характерну сигнатуру у вимірюваних метриках трафіку. Розуміння цих сигнатур дозволяє цілеспрямовано формувати вектор ознак для моделі виявлення. Зведена якісна характеристика основних класів наведена у таблиці 2.1.

Таблиця 2.1 – Сигнатури основних класів мережевих аномалій у метриках трафіку

Тип аномалії	Throughput	PPS	IAT	Asymmetry
DDoS SYN Flood	↑↑ різке	↑↑ різке	↓↓ короткий	↑↑ дисбаланс
DDoS UDP Flood	↑↑ різке	↑↑	↓↓	≈ норма / ↑
Port Scan	≈ норма	≈ або ↑	≈ норма	≈ норма
Data Exfiltration	↑ вихідний	≈ норма	≈ норма	↓↓ інверсія

DDoS-атаки (SYN/UDP Flood) – атаки на вичерпання ресурсів: різке зростання вхідного PPS і BPS, часто з мінімальним розміром пакетів, виражена асиметрія між відправленими та підтвердженими з'єднаннями [6].

Сканування портів (Port Scan) – PPS і BPS можуть залишатися у межах норми, проте різко зростає кількість унікальних портів призначення на одному цільовому хості; джерело – один IP, мета – множина портів. Виявлення потребує аналізу багатьох ознак одночасно.

Витік даних (Data Exfiltration) – окремі метрики залишаються в межах норми, але змінюється напрямок переважаючого трафіку – вихідний потік від клієнтського вузла зростає, особливо у неробочі години. Статистичні методи на одній ознаці виявити цей клас не здатні [7].

Описані сигнатури показують, що частина класів (DDoS) детектується вже за рівнем однієї метрики, тоді як інші (Port Scan, Data Exfiltration) – лише за комбінацією ознак. Це формує природний поділ між статистичними методами і моделями машинного навчання.

2.3 Базлайн і статистичні методи виявлення відхилень

Виявлення відхилень потребує наявності еталонного профілю – базлайну (англ. baseline), що описує нормальний стан мережі. Базлайн є статистичною моделлю значень метрик у кожному часовому слоті з урахуванням добових і тижневих циклів. Над кожною метрикою у вікні *W* обчислюються середнє, стандартне відхилення та процентилі; адаптивність забезпечується методом ковзного оновлення.

Для виділення тренду серед короткочасних флуктуацій застосовується ковзне середнє у вікні *W*:

$$MA(t) = \frac{1}{W}\sum_{i=0}^{W-1} x(t-i) \qquad (2.1)$$

де *MA(t)* – значення ковзного середнього у момент часу t; *W* – розмір вікна (кількість відліків); *x(t−i)* – значення метрики у момент часу t−i.

Розмір *W* – компроміс між згладжуванням і швидкістю реакції. Для динамічних середовищ застосовується експоненційне ковзне середнє (EMA), що швидше реагує на актуальні зміни:

$$EMA(t) = \alpha \cdot x(t) + (1-\alpha) \cdot EMA(t-1) \qquad (2.2)$$

де *α* – коефіцієнт згладжування, α ∈ (0, 1); *x(t)* – поточне значення метрики; *EMA(t−1)* – значення EMA на попередньому кроці.

Ковзна дисперсія оцінює мінливість метрики навколо тренду:

$$Var(t) = \frac{1}{W}\sum_{i=0}^{W-1}\big(x(t-i) - MA(t)\big)^2 \qquad (2.3)$$

де *Var(t)* – ковзна дисперсія у момент часу t; *W* – розмір вікна; *x(t−i), MA(t)* – позначення відповідно до формули (2.1).

Різке зростання дисперсії при незмінному середньому – характерна сигнатура початку флуду або сканування.

Метод Z-score оцінює, на скільки стандартних відхилень поточне значення метрики відхиляється від норми:

$$Z(t) = \frac{x(t)-\mu}{\sigma} \quad (2.4)$$

де $Z(t)$ – Z-оцінка у момент часу t; $x(t)$ – поточне значення метрики; μ – середнє значення метрики за базовий період; σ – стандартне відхилення метрики за той самий період.

За нормального розподілу ~99,7% значень потрапляють у $|Z| \leq 3$, тому $|Z(t)| > 3$ прийнято вважати викидом (рис. 2.2). Конкретний поріг визначається емпірично залежно від допустимого рівня хибних тривог [8].

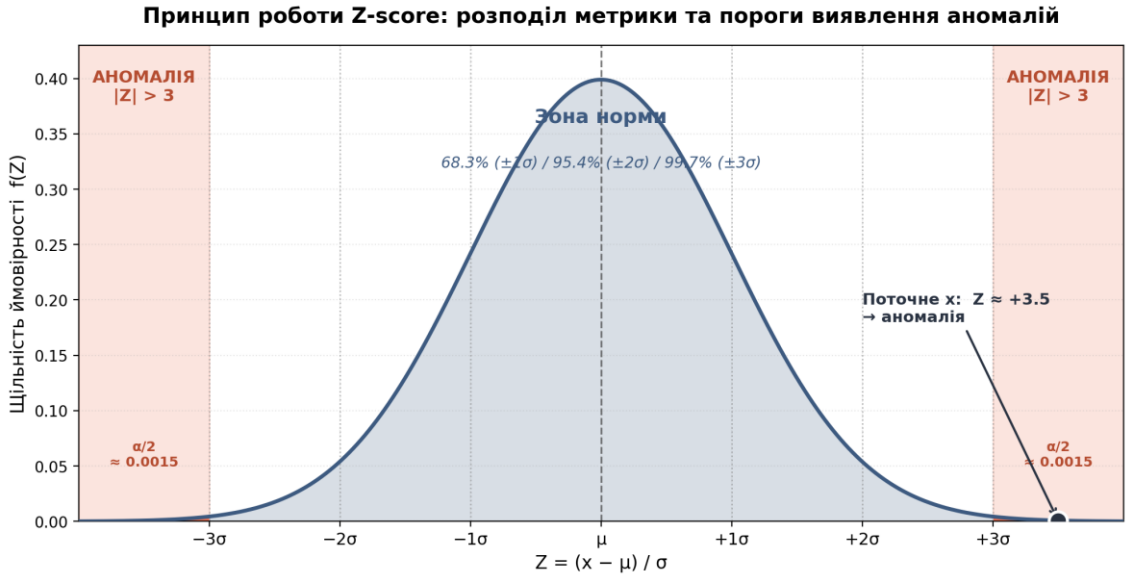

Рисунок 2.2 – Принцип роботи Z-score: розподіл значень метрики з порогами виявлення аномалій

Статистичні методи ефективно виявляють об'ємні аномалії (DDoS-атаки), де відхилення проявляється в одній метриці. Багатовимірні аномалії – сканування портів і витік даних – потребують моделей, які аналізують комбінацію ознак одночасно (підрозділ 2.4).

2.4 Алгоритми машинного навчання для класифікації мережевих потоків

Статистичні методи безпорадні перед аномаліями, які проявляються в комбінаціях ознак. Для них застосовуються алгоритми машинного навчання, що автоматично виділяють приховані патерни з багатовимірних даних трафіку. Задача формулюється як класифікація: на вхід моделі подається вектор ознак трафіку для кожного часового вікна; на виході – мітка класу (нормальний трафік / аномалія).

Random Forest – ансамблевий алгоритм машинного навчання, що будує велику кількість незалежних дерев рішень на випадкових підвибірках даних. Кожне дерево виносить власне рішення про клас вхідного вектора ознак; остаточна мітка формується голосуванням більшості дерев. Архітектура показана на рис. 2.3 [9].

Рисунок 2.3 – Архітектура ансамблю Random Forest для класифікації мережевого трафіку

Алгоритм природно пасує задачам мережевої безпеки з кількох причин. По-перше, він стійкий до перенавчання на невеликих датасетах – критично для лабораторної топології. По-друге, RF природно підтримує незбалансовані класи через відповідні параметри навчання – це важливо, бо аномалії

становлять малу частку від загального трафіку. По-третє, алгоритм надає оцінку важливості кожної ознаки, що дозволяє визначити, які саме метрики найбільш інформативні для виявлення кожного типу аномалії.

Налаштування моделі полягає у виборі трьох основних параметрів: кількості дерев в ансамблі, максимальної глибини окремого дерева та мінімального розміру підвибірки для розгалуження вузла. Підбір цих значень виконується методом крос-валідації – модель навчається кілька разів на різних розбиттях даних, щоб уникнути випадкового підлаштування під одну вибірку. Технологічна реалізація через бібліотеку scikit-learn винесена у Додаток А.

Random Forest вирішує задачу класифікації поточного стану, то для проактивного виявлення трендових аномалій потрібна модель часових рядів. LSTM (Long Short-Term Memory) – рекурентна нейронна мережа, що моделює довгострокові часові залежності завдяки внутрішньому механізму пам'яті. На відміну від класичних статистичних моделей, LSTM здатна враховувати поточні значення метрик у контексті довгих попередніх послідовностей [10].

У задачі моніторингу LSTM навчається на нормальному трафіку прогнозувати майбутні значення ключових метрик (BPS, PPS). Якщо фактичне значення суттєво відхиляється від прогнозу, це сигналізує про аномалію. Такий підхід дозволяє виявляти повільні трендові аномалії – наприклад, поступове наростання витоку даних чи зараження ботнетом, – які пропускаються статистичними методами на одній метриці.

Поєднання Random Forest та LSTM утворює дворівневу схему: перша модель забезпечує швидку реактивну класифікацію поточного стану, друга – проактивне прогнозування трендів. Обидві розміщуються на сервері аналізу і працюють як єдине аналітичне ядро АПК.

2.5 Метрики оцінки виявлення мережевих аномалій

Об'єктивна оцінка моделей виявлення аномалій є обов'язковим етапом розробки. Специфіка задачі мережевої безпеки накладає особливі вимоги до метрик: класи суттєво незбалансовані (аномалії становлять малу частку

трафіку), а вартість помилок різна – пропущена атака коштує значно більше за хибну тривогу.

Основа для розрахунку всіх метрик – матриця помилок, що зіставляє фактичний клас з передбаченим (рис. 2.4). Її елементи мають конкретне значення для безпеки: *TP* – реально виявлені атаки; *TN* – коректно ідентифікований нормальний трафік; *FP* – хибна тривога (перевантажує адміністратора); *FN* – пропущена атака (критична помилка з позиції безпеки).

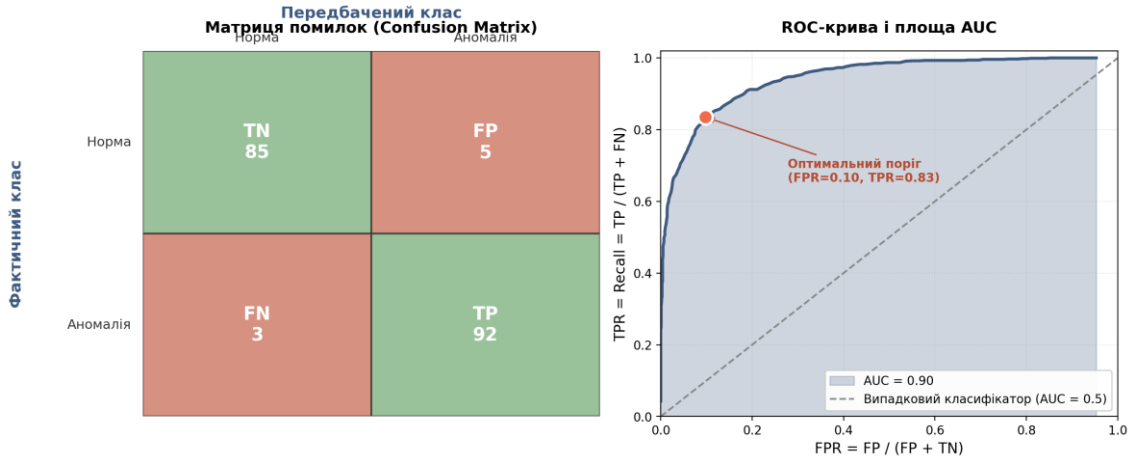

Рисунок 2.4 – Матриця помилок (Confusion Matrix) та ROC-крива

Базові метрики обчислюються за матрицею помилок таким чином:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \tag{2.5}$$

$$Precision = \frac{TP}{TP+FP} \tag{2.6}$$

$$Recall = \frac{TP}{TP+FN} \tag{2.7}$$

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision+Recall} \tag{2.8}$$

де *TP*, *TN* – кількість істинно-позитивних та істинно-негативних спрацювань; *FP*, *FN* – кількість хибно-позитивних та хибно-негативних спрацювань.

Accuracy за умов незбалансованих класів є оманливою: класифікатор, що завжди передбачає «норму», отримає високе значення з нульовою практичною цінністю. *Precision* показує частку справжніх атак серед усіх спрацювань, *Recall* – частку виявлених реальних атак. Між ними існує фундаментальний компроміс: зниження порогу класифікації підвищує Recall, але знижує

Precision. F1-score як гармонічне середнє між ними є основною метрикою якості для незбалансованих класів [11].

ROC-крива показує співвідношення між частотою хибних тривог і часткою виявлених атак при зміні порогу класифікації від 0 до 1. *AUC* (Area Under the Curve) – площа під цією кривою; чим ближче до 1, тим краще модель розпізнає аномалії незалежно від обраного порогу. *AUC* = 0,5 відповідає випадковому передбаченню; для систем виявлення мережевих аномалій орієнтиром є *AUC* > 0,9. У роботі основними метриками обрано *F1-score* (як показник якості) та *ROC-AUC* (для порівняння моделей).

2.6 Вимоги до апаратно-програмного комплексу моніторингу

Результати аналізу трафіку, методів виявлення та метрик оцінки є основою для формулювання вимог до апаратно-програмного комплексу (АПК) інтелектуального моніторингу. Вимоги поділяються на функціональні (поведінка) та нефункціональні (якісні характеристики). Зведений перелік наведено у таблиці 2.2.

Таблиця 2.2 – Зведений перелік функціональних і нефункціональних вимог до АПК моніторингу

Категорія	Вимога	Конкретизація
Функціональна	Збір трафіку	Безперервний пасивний збір метрик у вікнах 10–60 с без втрат пакетів
Функціональна	Базлайн	Автоматичне формування й ковзне оновлення (добові, тижневі цикли)
Функціональна	Каскадне виявлення	Z-score → Random Forest, затримка ≤ 2 часові вікна
Функціональна	Прогнозування	LSTM, горизонт 5–30 хв для BPS, PPS
Функціональна	Класифікація типів	Багатокласова: Normal / DDoS / Port Scan / Data Exfiltration
Нефункціональна	Продуктивність	Затримка аналізу одного вікна ≤ 5 с на платформі 4 ядер CPU + 8 ГБ RAM
Нефункціональна	Апаратний вузол	≥ 2 ядер CPU, ≥ 2 ГБ RAM, NIC 1 Гбіт/с для SPAN (без IP-адресації)
Нефункціональна	Ізоляція модулів	Контейнери на обчислювальному вузлі, стабільний ресурсний профіль
Нефункціональна	Масштабованість	Горизонтальна – додавання нових агентів збору
Нефункціональна	Відмовостійкість	Відтворення базлайну та моделей після перезапуску

Ключові функціональні вимоги охоплюють пасивний збір метрик у часових вікнах, автоматичне формування й оновлення базлайну, каскадне виявлення (статистична фільтрація → ML-класифікатор), прогнозування ключових метрик та багатокласову класифікацію типів аномалій.

Ключові нефункціональні вимоги визначають квазіреальний час обробки, мінімальну апаратну специфікацію обчислювального вузла, ізольовані середовища виконання модулів, масштабованість і відмовостійкість з відтворенням стану після перезапуску.

РОЗДІЛ 3 ПРОЄКТУВАННЯ ТА РОЗГОРТАННЯ АПАРАТНО-ПРОГРАМНОГО КОМПЛЕКСУ МОНІТОРИНГУ

3.1 Проєктування експериментальної мережевої топології

Проєктування експериментальної мережевої топології виконано у середовищі емуляції комп'ютерних мереж GNS3. Вибір GNS3 обґрунтований повноцінною підтримкою віртуалізації мережевих пристроїв та можливістю інтеграції з гіпервізором VirtualBox для розгортання повноцінних віртуальних машин у ролі кінцевих вузлів та сервера моніторингу.

Спроєктована мережева інфраструктура складається з 15 активних вузлів, центрального комутатора, маршрутизатора, сервера застосунків та виділеного сервера моніторингу. Загальний вигляд топології у середовищі GNS3 наведено на рис. 3.1.

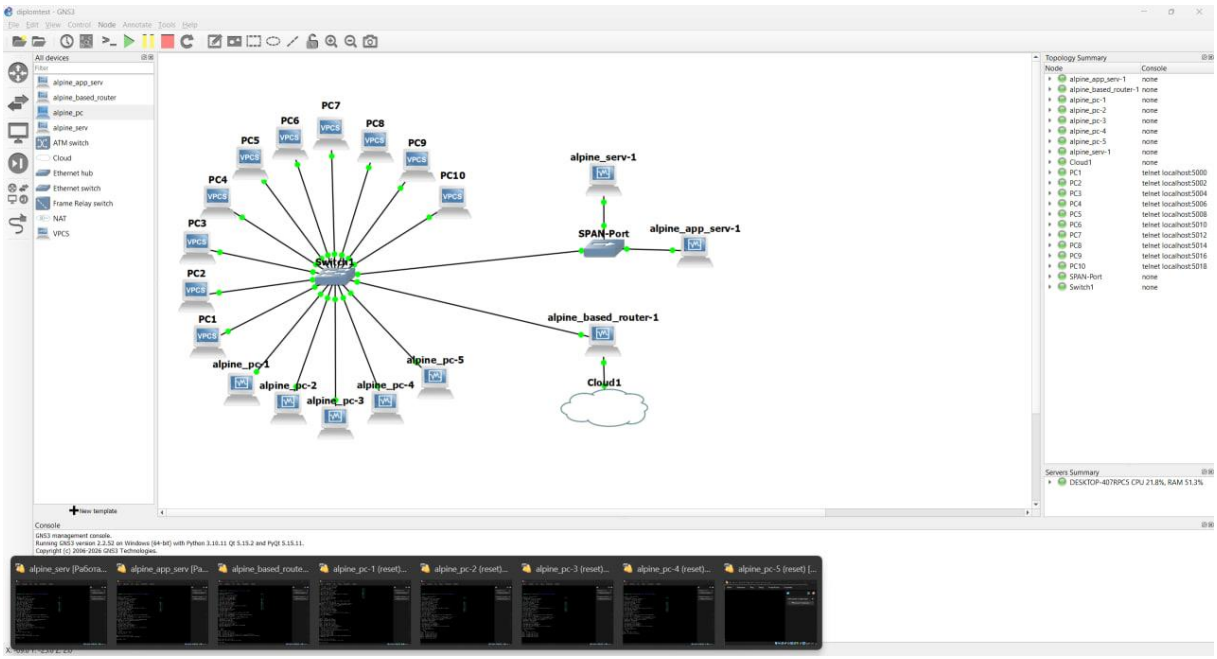

Рисунок 3.1 – Експериментальна мережева топологія у середовищі GNS3

Усі вузли підключені до центрального комутатора Switch1, який виконує функції рівня доступу. Маршрутизатор alpine_based_router-1 з'єднує внутрішню мережу із зовнішнім сегментом (Cloud), забезпечуючи симуляцію виходу до зовнішніх мереж. Сервер застосунків alpine_app_serv-1 виступає у

ролі цільового хоста для генерованого трафіку (HTTP-запити, файлові передачі, об'єкт атак). Сервер моніторингу alpine_serv-1 – виділений обчислювальний вузол, що приймає дзеркальну копію всього трафіку через окремий SPAN-вузол. Склад топології наведено в таблиці 3.1.

Таблиця 3.1 – Склад вузлів експериментальної мережевої топології

Тип вузла	К-ть	Платформа	Призначення
Клієнтська станція (VPCS)	10	PC1–PC10	Генерація фонового ICMP-трафіку
Клієнтська станція (Alpine)	5	alpine_pc-1…5	Генерація легітимного навантаження (iperf3, ab) та атак (hping3)
Сервер застосунків	1	alpine_app_serv-1	Цільовий хост – приймач трафіку і об'єкт атак
Маршрутизатор	1	alpine_based_router-1	Шлюз внутрішньої мережі
Комутатор доступу	1	Switch1	Центральний агрегатор клієнтського трафіку
Відгалужувач (SPAN-tap)	1	SPAN-Port	Дзеркалювання трафіку сервера застосунків
Сервер моніторингу	1	alpine_serv-1	Прийом дзеркальної копії, обчислювальний вузол аналізу

Базовий адресний план інфраструктури використовує приватну підмережу 192.168.1.0/24 для клієнтського сегмента та окремий канал керування для сервера моніторингу. Маршрутизація між сегментами реалізована засобами маршрутизатора без додаткових протоколів динамічної маршрутизації – для топології з обмеженою кількістю вузлів достатньо статичних маршрутів за замовчуванням.

3.2 Конфігурування каналу пасивного моніторингу

Ключовою інженерною особливістю спроєктованої топології є реалізація пасивного моніторингу через окремий вузол-відгалужувач SPAN-Port. У стандартному комутаторі рівня доступу функція дзеркалювання портів (Switched Port Analyzer) активується через системну команду виду monitor session 1 source interface ... destination interface ..., однак базовий емулятор

Ethernet switch у середовищі GNS3 не підтримує повну функціональність SPAN-сесії.

Для подолання цього обмеження застосовано класичний інженерний прийом – у топологію введено додатковий комутаційний пристрій (SPAN-Port), що фізично розташований у розриві каналу між основним комутатором Switch1 та сервером застосунків alpine_app_serv-1. Цей пристрій працює у режимі hub-відгалужувача: будь-який вхідний або вихідний кадр, що проходить через канал зв'язку до сервера застосунків, дублюється на додатковий порт, до якого підключений виділений сервер моніторингу alpine_serv-1.

Таке рішення повністю відповідає логіці SPAN з режимом копіювання both (rx+tx) – на сервер моніторингу потрапляє повний двонаправлений трафік цільового вузла. Перевагою цього підходу є відсутність впливу на основний комутатор, його матрицю комутації та QoS-політики. Недоліком – обмеження на масштабованість: для додаткових цільових вузлів потрібні окремі SPAN-tap пристрої. Для лабораторної топології з єдиним сервером застосунків ця архітектура є оптимальною.

Сервер моніторингу alpine_serv-1 приймає дзеркальний трафік через інтерфейс, налаштований у режимі promiscuous mode (приймання всіх кадрів незалежно від MAC-адреси призначення). Це забезпечує повне захоплення мережевого потоку без втрат на канальному рівні.

3.3 Розгортання сервера моніторингу на Alpine Linux

У ролі виділеного обчислювального вузла моніторингу обрано віртуальну машину з операційною системою Alpine Linux, розгорнуту через гіпервізор VirtualBox та інтегровану у топологію GNS3. Вибір Alpine Linux обґрунтований мінімалістичним характером дистрибутиву (базовий образ < 200 МБ), низьким споживанням оперативної пам'яті (< 256 МБ у штатному режимі) та повноцінною підтримкою мережевих утиліт, необхідних для захоплення трафіку (libpcap, tcpdump, scapy-сумісне середовище).

Конфігурація обчислювального вузла включає два мережеві інтерфейси з різними призначеннями. Перший інтерфейс підключений до SPAN-tap пристрою у режимі promiscuous, не має IP-адреси з боку джерельного домену та використовується виключно для приймання дзеркальної копії трафіку. Другий інтерфейс налаштований у режимі NAT (Network Address Translation) у VirtualBox і забезпечує канал керування – доставку зібраних pcap-файлів на хост-машину та віддалене адміністрування. Така ізоляція мережевих площин (data plane vs management plane) є базовою інженерною практикою для систем моніторингу і запобігає формуванню зворотного трафіку у дзеркальний канал.

На сервері моніторингу запущено вбудований у Python http-сервер (модуль http.server), що дозволив здійснити експорт зібраних pcap-файлів через інтерфейс localhost з хост-машини Windows. Цей підхід забезпечує просту й надійну передачу даних без необхідності монтування спільних томів VirtualBox або налаштування додаткових файлових служб.

3.4 Подолання інженерних проблем розгортання інфраструктури

Розгортання комбінованої інфраструктури з використанням емулятора мережевих топологій GNS3, гіпервізора VirtualBox та операційної системи Alpine Linux на хост-машині під керуванням Windows виявило низку системних та мережевих проблем, які потребували окремих інженерних рішень. Опис цих проблем і способів їх подолання є складовою частиною дослідної експлуатації апаратно-програмного комплексу.

Під час створення вузлів-клонів на основі базового образу Alpine спостерігалися помилки виду «VDI disk not found» та неможливість створення знімків стану (snapshot). Першопричина – невідповідність між внутрішнім реєстром VirtualBox (Machines.xml) та посиланнями на віртуальні машини у конфігураційному файлі проєкту GNS3 (project.gns3 у форматі JSON). Розв'язання реалізоване у три кроки: ручне очищення реєстру VirtualBox від некоректних посилань на видалені або переміщені VDI-файли; створення проміжних «пустих» вузлів у GNS3 для перереєстрації клонів; пряме

редагування JSON-конфігурації проєкту з виправленням UUID віртуальних машин.

Сервер моніторингу alpine_serv-1, інтегрований у внутрішню топологію GNS3, не мав прямого мережевого виходу до хост-операційної системи Windows для експорту зібраних pcap-файлів. Розв'язання реалізоване через додавання другого мережевого адаптера віртуальної машини з режимом NAT у налаштуваннях VirtualBox. Конфігурація IP-адресації нового інтерфейсу виконана засобами утиліти iproute2 безпосередньо у середовищі Alpine. Перенаправлення TCP-портів (Port Forwarding) налаштоване у профілі NAT VirtualBox, що забезпечило доступність сервісів сервера моніторингу через інтерфейс localhost хост-машини.

Для зручної доставки великих pcap-файлів (сумарно понад 880 МБ) на хост-машину розгорнуто легковагий http-сервер засобами стандартного модуля Python (python3 -m http.server). Цей інструмент є частиною базового дистрибутиву Python і не потребує додаткової конфігурації, що відповідає принципу мінімізації службового програмного забезпечення на вузлі моніторингу.

Описані проблеми та способи їх подолання демонструють реальні інженерні рішення мережевого та системного характеру, що супроводжують розгортання апаратно-програмного комплексу у комбінованому середовищі віртуалізації.

3.5 Обчислювальний конвеєр обробки мережевого трафіку

Аналітичне ядро обчислювального вузла моніторингу реалізовано у вигляді виконуваного модуля main.py, що утворює технологічний конвеєр обробки мережевого трафіку. Виконуваний модуль розгорнуто у середовищі WSL (Windows Subsystem for Linux) на хост-машині після експорту pcap-файлів із сервера моніторингу. Такий розподіл обчислень – захоплення на Alpine VM, аналіз на потужнішому хості – відповідає типовій інженерній практиці розділення функцій збору та аналізу між вузлами різної потужності.

Технологічний конвеєр складається з чотирьох послідовних обчислювальних етапів. На першому етапі засобами бібліотеки scapy виконується потокове декодування pcap-дампу з групуванням пакетів у часові вікна тривалістю 1 секунда (параметр WINDOW_SIZE = 1.0). Для кожного вікна обчислюється вектор з семи кількісних ознак: загальна кількість пакетів, сумарний обсяг переданих байтів, середній розмір пакета, кількість унікальних IP-адрес джерел, кількість TCP-пакетів з прапором SYN, кількість UDP-пакетів та похідна метрика bytes_per_sec (середнє навантаження каналу).

На другому етапі виконується статистичний аналіз ознаки bytes_per_sec методом Z-score з порогом виявлення $|Z| > 3$ (за теоретичним обґрунтуванням, наведеним у підрозділі 2.3). Виявлені аномальні вікна позначаються міткою anomaly_z = 1.

На третьому етапі вектор ознак подається на вхід математичної моделі Random Forest (ансамбль з 100 вирішальних дерев, random_state = 42). Датасет розділяється на навчальну та тестову підмножини у співвідношенні 70/30 із стратифікацією за міткою класу для збереження пропорції нормальних та аномальних зразків в обох підмножинах. Модель класифікує кожне часове вікно як «нормальний трафік» (клас 0) або «атакований трафік» (клас 1).

На четвертому етапі формуються метрики якості (матриця помилок, Accuracy, Precision, Recall, F1-score, ROC-AUC), оцінка важливості ознак (feature importance) та візуалізації результатів у вигляді графічних файлів. Технологічна реалізація конвеєра засобами бібліотек scapy, pandas, numpy, scikit-learn та matplotlib наведена у Додатку А (лістинг А.1).

3.6 Генерація еталонного та атакованого мережевого трафіку

Для формування репрезентативного датасету було згенеровано два сценарії мережевого трафіку – еталонний нормальний та атакований. Інструментарій генерації трафіку наведено в таблиці 3.2.

Таблиця 3.2 – Інструментарій генерації мережевого трафіку

Тип трафіку	Утиліта	Джерело	Цільовий вузол
Фоновий ICMP	ping (стандартна)	PC1–PC10 (10 VPCS)	Кругова відправка
Файлові передачі	iperf3	alpine_pc-1…5	alpine_app_serv-1
HTTP-запити до сервера	ab (Apache Benchmark)	alpine_pc-1…5	alpine_app_serv-1
SYN Flood (DoS)	hping3 -S --flood	alpine_pc-1…5	alpine_app_serv-1
UDP Flood (DoS)	hping3 --udp --flood	alpine_pc-1…5	alpine_app_serv-1

Кожен сценарій записувався тривалістю приблизно 15 хвилин (близько 800–850 часових вікон по 1 секунді кожне). Об'єм еталонного нормального трафіку склав 690 МБ, атакованого – 191 МБ. Така висока щільність нормального трафіку досягнута завдяки активним передачам великих файлів через iperf3 між клієнтськими станціями та сервером застосунків.

У сценарії атакованого трафіку п'ять клієнтських станцій на базі Alpine Linux (alpine_pc-1 – alpine_pc-5) виступали у ролі локального ботнету, що здійснював розподілену DoS-атаку на сервер застосунків alpine_app_serv-1. Розподіленість атаки забезпечує реалістичну варіацію за ознакою unique_ips (кількість унікальних адрес джерел), що є важливою інформативною метрикою для алгоритму класифікації Random Forest.

Сумарний об'єм датасету склав 775 розмічених часових вікон: 549 з міткою «нормальний трафік» (target = 0) та 226 з міткою «атакований трафік» (target = 1). Дисбаланс класів приблизно 70/30 відповідає реалістичному співвідношенню для систем виявлення аномалій, у яких атаки складають меншість від загального трафіку.

3.7 Дослідна експлуатація апаратно-програмного комплексу

Після збирання датасету проведено дослідну експлуатацію апаратно-програмного комплексу з оцінкою ефективності обох рівнів виявлення – статистичного фільтра Z-score та ML-класифікатора Random Forest.

Часовий ряд показника bytes_per_sec на всьому періоді запису наведено на рис. 3.2. На графіку чітко простежуються два інтенсивні сплески навантаження – перший близько вікон 100–150 з рівнем приблизно 6 МБ/с (легітимна файлова передача засобами iperf3), другий близько вікон 620–700 з рівнем приблизно 12 МБ/с (UDP Flood атака). Червона крива ковзного середнього з вікном 10 згладжує короткочасні флуктуації та виділяє стійкі рівні навантаження.

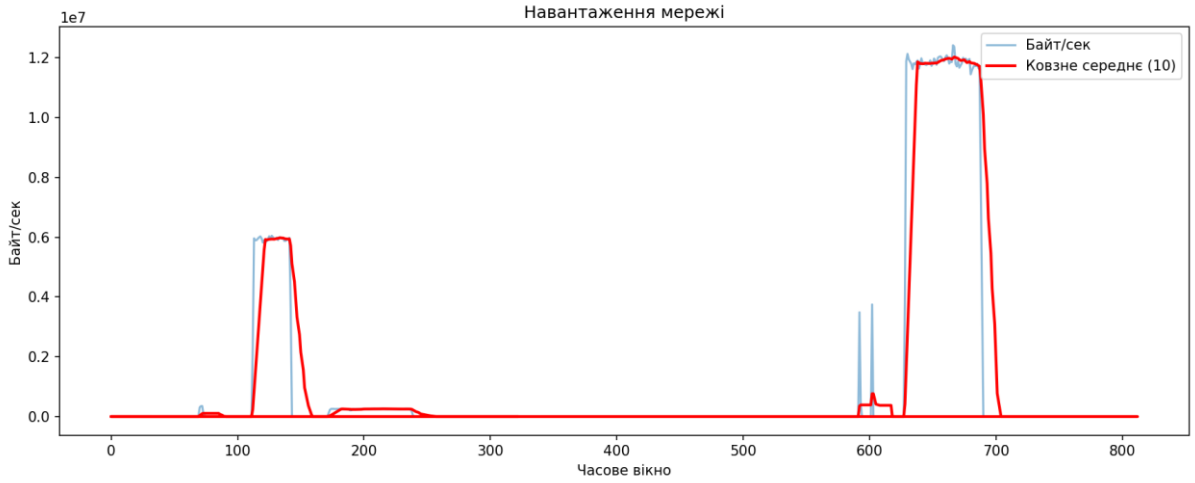

Рисунок 3.2 – Динаміка мережевого навантаження протягом запису (bytes_per_sec)

Результати застосування методу Z-score з порогом $|Z| > 3$ до показника bytes_per_sec наведено на рис. 3.3. На графіку видно, що поріг перетинається лише для невеликої частини вікон під час UDP Flood атаки (значення Z досягають 3,3–3,5). Перший сплеск (легітимна iperf3-передача) дає $Z \approx 1{,}5$, що нижче порогу – це коректно, оскільки навантаження легітимне. Однак значна частина атакованих вікон залишається у межах $|Z| \leq 3$ – це обмеження одновимірного статистичного фільтра, що обґрунтовує необхідність переходу до багатовимірної ML-класифікації.

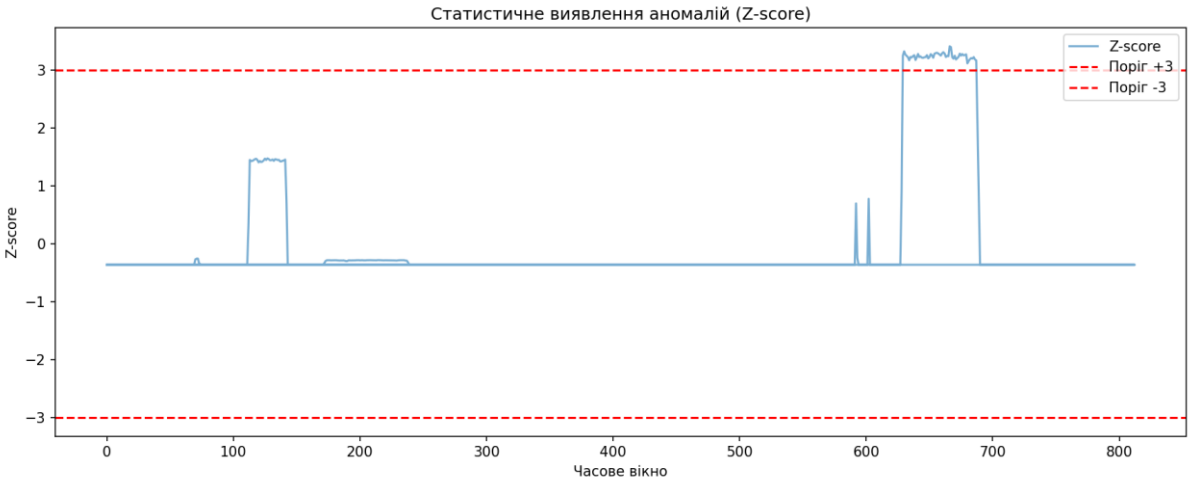

Рисунок 3.3 – Статистичне виявлення аномалій методом Z-score

Матриця помилок, отримана на тестовій підмножині (30% датасету), наведена на рис. 3.4. Числові показники: TP = 30, TN = 163, FP = 2, FN = 38.

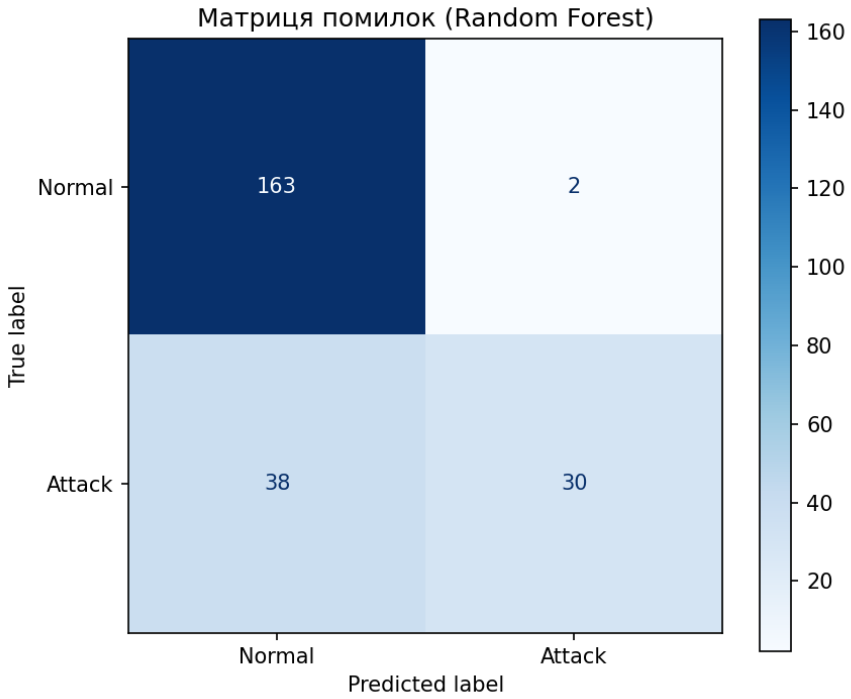

Рисунок 3.4 – Матриця помилок моделі Random Forest на тестовій підмножині

Розрахунок основних метрик якості наведено в таблиці 3.3.

Таблиця 3.3 – Метрики ефективності апаратно-програмного комплексу

Метрика	Значення	Інтерпретація
Accuracy	0,83	Загальна частка правильних класифікацій
Precision	0,94	Серед спрацювань – 94% реальних атак
Recall	0,44	Виявлено 44% реальних атак
F1-score	0,60	Збалансована оцінка для незбалансованих класів
ROC-AUC	0,751	Дискримінаційна здатність моделі
False Positive	2	Лише 2 хибні тривоги на 165 нормальних вікон

ROC-крива моделі Random Forest наведена на рис. 3.5. Площа під кривою AUC = 0,751 свідчить про задовільну дискримінаційну здатність моделі – суттєво краще за випадкове передбачення (AUC = 0,5), але нижче орієнтиру 0,9, типового для високопродуктивних промислових систем. Це базовий результат для конфігурації Random Forest з типовими параметрами без подальшої оптимізації.

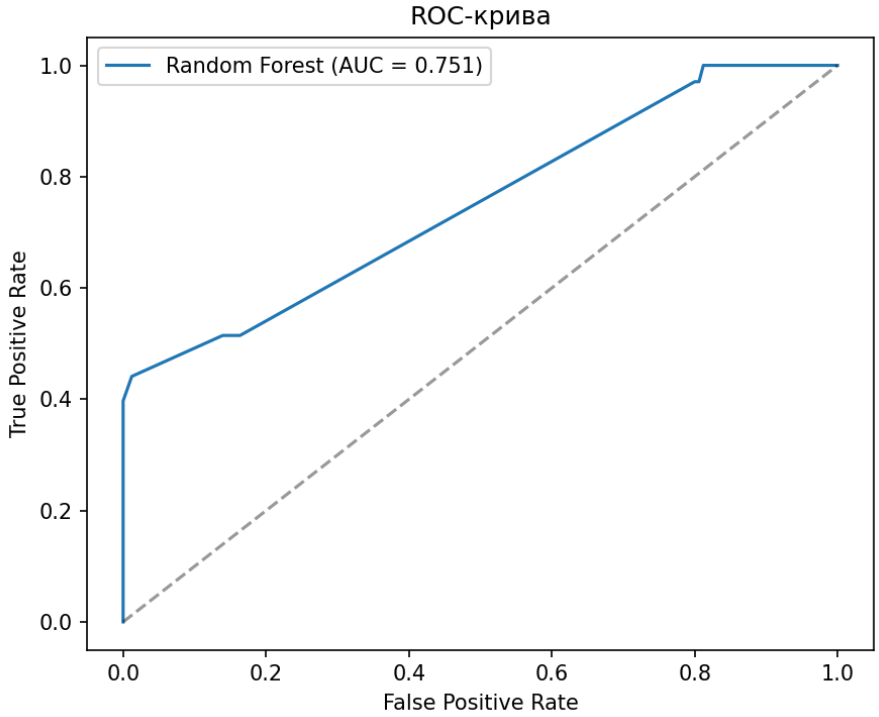

Рисунок 3.5 – ROC-крива моделі Random Forest

Розподіли ключових інженерних метрик (packet_count, bytes_per_sec, syn_count, udp_count) для нормального та атакованого трафіку наведено на рис. 3.6. На графіках видно чітке розділення класів за більшістю ознак: атакований трафік формує окремі скупчення на високих значеннях, що

пояснює здатність Random Forest до точної класифікації за умови комбінованого аналізу всіх ознак.

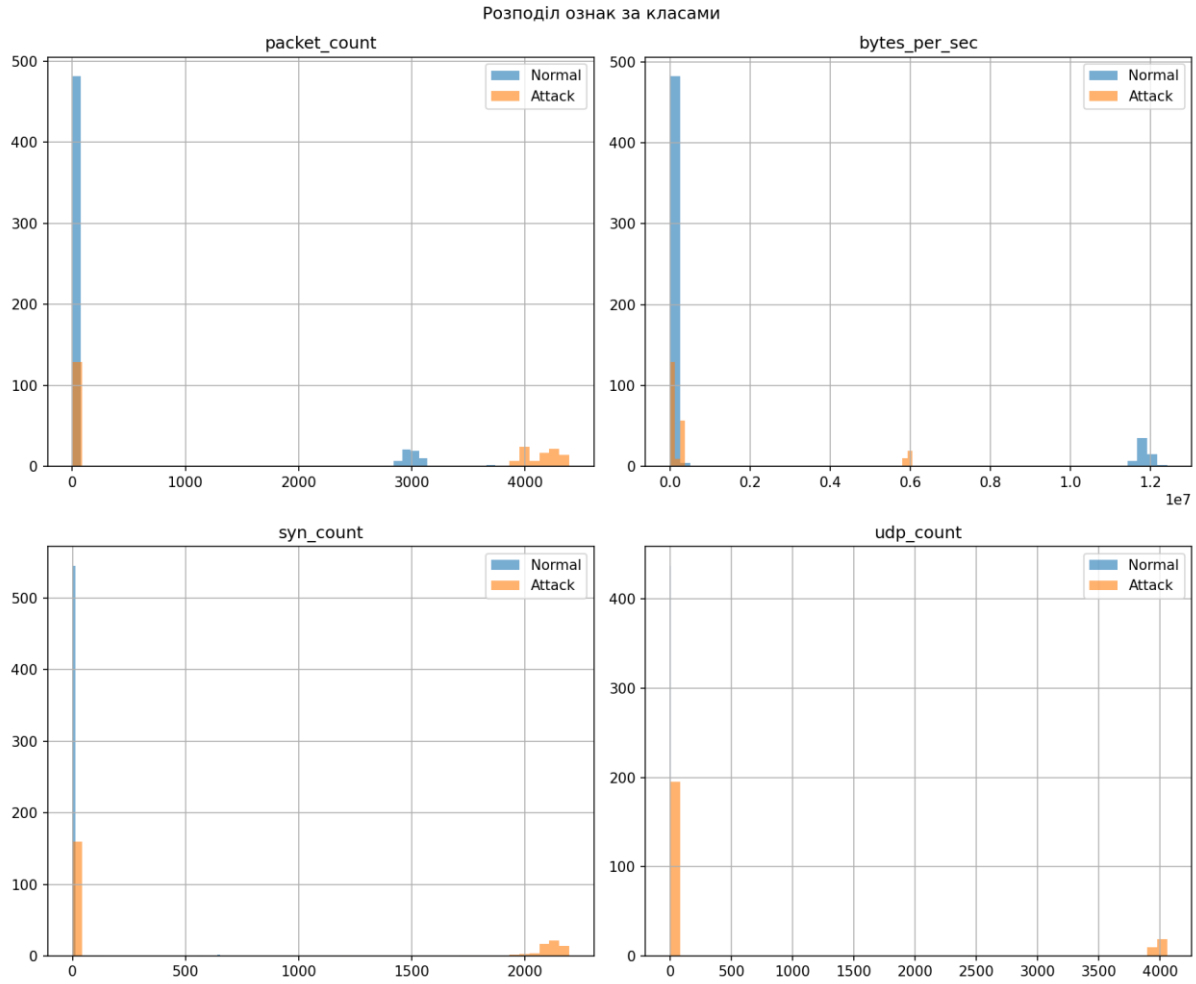

Рисунок 3.6 – Розподіл основних інженерних ознак трафіку за класами

Дослідна експлуатація виявила два очікувані обмеження поточної конфігурації. По-перше, низький показник Recall (44%) пояснюється пропуском атакованих вікон у перехідні моменти – на початку (зародження атаки) та в кінці (затухання), коли ознаки ще не виражені на повну. По-друге, помірний показник AUC (0,751) є наслідком використання базової конфігурації Random Forest з типовими гіперпараметрами (n_estimators = 100) без тюнінгу. Шляхи покращення – застосування методу k-fold cross-validation для оптимізації гіперпараметрів та збільшення розміру навчальної вибірки – обґрунтовані як напрям подальшого розвитку апаратно-програмного комплексу.

Висока специфічність моделі (лише 2 хибні тривоги на 165 нормальних вікон) є практично цінною характеристикою, оскільки запобігає перевантаженню адміністратора нерелевантними сповіщеннями та забезпечує довіру до системи виявлення.

ВИСНОВКИ

У кваліфікаційній роботі розв'язано науково-практичне завдання підвищення спостережуваності локальної мережі за рахунок розроблення апаратно-програмного комплексу інтелектуального моніторингу трафіку з функціями прогнозування аномалій. Систематизовано теоретичні основи моніторингу – моделі OSI та TCP/IP, технології пасивного знімання трафіку (SPAN, RSPAN, TAP), класифікацію мережевих аномалій та підходи до їх виявлення.

На основі аналізу властивостей мережевого трафіку обґрунтовано перелік інформативних ознак – інтенсивність пакетів, обсяг переданих байтів, частку SYN-пакетів та кількість унікальних адрес джерела. Виконано порівняння статистичних і машинно-навчальних методів виявлення аномалій та запропоновано каскадну архітектуру комплексу, що поєднує статистичний фільтр Z-score і класифікатор Random Forest. Така структура дозволяє знизити частку хибно-позитивних рішень за рахунок попередньої фільтрації шуму без відчутних втрат чутливості.

Спроєктовано та реалізовано експериментальну топологію локальної мережі у середовищі GNS3 з виділеним SPAN-портом, розгорнуто сервери на базі Alpine Linux та реалізовано аналітичне ядро мовою Python з використанням бібліотек scapy, pandas і scikit-learn. Сформовано навчально-тестовий датасет із 775 фрагментів трафіку, отриманих засобами iperf3, ab та hping3 з подальшим маркуванням за класами. Експериментальна перевірка комплексу засвідчила його працездатність: каскадна модель досягла значення AUC = 0,751 та F1-метрики 0,60 на тестовій вибірці, що відповідає очікуваним показникам для базових систем виявлення аномалій з обмеженою кількістю ознак.

Практична цінність роботи полягає у можливості використання розробленого апаратно-програмного комплексу у складі інфраструктури моніторингу локальних мереж навчальних закладів, малих та середніх

організацій, де відсутні комерційні рішення класу IDS/IPS. Шляхами подальшого розвитку визначено розширення ознакового простору за рахунок міжсесійних статистик, інтеграцію рекурентних нейронних мереж LSTM для врахування часових залежностей у трафіку, перехід до потокової обробки в режимі реального часу та верифікацію комплексу на більших мережевих сегментах. Поставлена у роботі мета досягнута, а всі поставлені завдання виконано у повному обсязі.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Мікітишин А. Г., Митник М. М. Комп'ютерні мережі. Книга 1 : навчальний посібник. Львів : Магнолія 2006, 2013. 256 с. URL: https://elartu.tntu.edu.ua/bitstream/123456789/16930/5/Mykytyshyn_A_G_Mytnyk_M_M_Kompjuterni_merezhi_Knyga_1.pdf (дата звернення: 22.10.2025).

2. Hofstede R., Čeleda P., Trammell B., Drago I., Sadre R., Sperotto A., Pras A. Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. IEEE Communications Surveys & Tutorials. 2014. Vol. 16, No. 4. P. 2037–2064. DOI: https://doi.org/10.1109/COMST.2014.2321898. URL: https://research.utwente.nl/files/6519118/tutorial.pdf (дата звернення: 05.11.2025).

3. Najafimehr M., Zarifzadeh S., Mostafavi S. DDoS attacks and machine-learning-based detection methods: A survey and taxonomy. Engineering Reports. 2023. Vol. 5, No. 12. Article ID e12697. DOI: https://doi.org/10.1002/eng2.12697. URL: https://onlinelibrary.wiley.com/doi/pdfdirect/10.1002/eng2.12697 (дата звернення: 19.11.2025).

4. Qutqut M. H., Taqi M. K. A Comparative Evaluation of Snort and Suricata for Detecting Data Exfiltration Tunnels in Cloud Environments. Journal of Cybersecurity and Privacy. 2025. Vol. 6, No. 1. Article 17. DOI: https://doi.org/10.3390/jcp6010017. URL: https://www.mdpi.com/2624-800X/6/1/17/pdf (дата звернення: 03.12.2025).

5. Meng X. Behavior Pattern Mining from Traffic and Its Application to Network Anomaly Detection. Security and Communication Networks. 2022. Vol. 2022. Article ID 9139321. DOI: https://doi.org/10.1155/2022/9139321. URL: https://onlinelibrary.wiley.com/doi/pdfdirect/10.1155/2022/9139321 (дата звернення: 14.01.2026).

6. Babu N. R. Y., Aravind A., Govindarajan U. H. A Machine-Learning-Based Approach for the Detection and Mitigation of Distributed Denial-of-Service Attacks in Internet of Things Environments. Applied Sciences. 2025. Vol. 15, No. 11. Article

6012. DOI: https://doi.org/10.3390/app15116012. URL: https://www.mdpi.com/2076-3417/15/11/6012/pdf (дата звернення: 28.01.2026).

7. Willems D., Kohls K., van der Kamp B., Vranken H. Data Exfiltration Detection on Network Metadata with Autoencoders. Electronics. 2023. Vol. 12, No. 12. Article 2584. DOI: https://doi.org/10.3390/electronics12122584. URL: https://www.mdpi.com/2079-9292/12/12/2584/pdf (дата звернення: 11.02.2026).

8. Kumari D. та ін. A Comprehensive Investigation of Anomaly Detection Methods in Deep Learning and Machine Learning: 2019–2023. IET Information Security. 2024. Vol. 2024. Article ID 8821891. DOI: https://doi.org/10.1049/2024/8821891. URL: https://ietresearch.onlinelibrary.wiley.com/doi/pdfdirect/10.1049/2024/8821891 (дата звернення: 18.02.2026).

9. Breiman L. Random Forests. Machine Learning. 2001. Vol. 45, No. 1. P. 5–32. DOI: https://doi.org/10.1023/A:1010933404324. URL: https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf (дата звернення: 25.02.2026).

10. Hochreiter S., Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997. Vol. 9, No. 8. P. 1735–1780. DOI: https://doi.org/10.1162/neco.1997.9.8.1735. URL: https://www.bioinf.jku.at/publications/older/2604.pdf (дата звернення: 04.03.2026).

11. Chicco D., Jurman G. The Matthews correlation coefficient (MCC) should replace the ROC AUC as the standard metric for assessing binary classification. BioData Mining. 2023. Vol. 16. Article 4. DOI: https://doi.org/10.1186/s13040-023-00322-4. URL: https://link.springer.com/content/pdf/10.1186/s13040-023-00322-4.pdf (дата звернення: 25.03.2026).

ДОДАТКИ

Додаток А – Текст сценарію для побудови датасету та навчання моделі

```
"""
Інтелектуальний моніторинг трафіку в локальній мережі з прогнозуванням
аномалій.

Скрипт для:
1. Парсингу pcap-файлів (нормальний та аномальний трафік)
2. Витягу ознак (feature extraction) у часових вікнах
3. Формування датасету traffic_dataset.csv
4. Статистичного виявлення аномалій (Z-score)
5. Навчання моделі Random Forest
6. Оцінки ефективності та візуалізації результатів
"""


import sys
from collections import defaultdict
from pathlib import Path


import matplotlib
matplotlib.use("Agg")
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from scapy.all import IP, TCP, UDP, PcapReader
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import (
    ConfusionMatrixDisplay,
    classification_report,
    confusion_matrix,
    roc_auc_score,
    roc_curve,
)
from sklearn.model_selection import train_test_split


# ---------------------------------------------------------------------------
# Константи
# ---------------------------------------------------------------------------
WINDOW_SIZE = 1.0  # часове вікно в секундах
NORMAL_PCAP = Path("normal_traffic.pcap")
ATTACK_PCAP = Path("attack_traffic.pcap")
DATASET_CSV = Path("traffic_dataset.csv")
PLOTS_DIR = Path("plots")



# ---------------------------------------------------------------------------
# 1. Feature Extraction – витяг ознак з pcap (потокова обробка)
# ---------------------------------------------------------------------------
def extract_features(pcap_path: Path, label: int) -> list[dict]:
    """Читає pcap потоково та групує пакети у часові вікна."""
    print(f"  Обробка {pcap_path} (label={label}) ...")

    windows: dict[int, dict] = defaultdict(
        lambda: {
            "packet_count": 0,
            "total_bytes": 0,
```

```
        "sizes": [],
        "src_ips": set(),
        "syn_count": 0,
        "udp_count": 0,
    }
)

base_time = None
pkt_total = 0

with PcapReader(str(pcap_path)) as reader:
    for pkt in reader:
        pkt_total += 1
        if pkt_total % 500_000 == 0:
            print(f"    ... оброблено {pkt_total:,} пакетів")

        ts = float(pkt.time)
        if base_time is None:
            base_time = ts

        window_id = int((ts - base_time) / WINDOW_SIZE)
        w = windows[window_id]

        pkt_len = len(pkt)
        w["packet_count"] += 1
        w["total_bytes"] += pkt_len
        w["sizes"].append(pkt_len)

        if pkt.haslayer(IP):
            w["src_ips"].add(pkt[IP].src)

        if pkt.haslayer(TCP):
            flags = pkt[TCP].flags
            if flags & 0x02:  # SYN
                w["syn_count"] += 1

        if pkt.haslayer(UDP):
            w["udp_count"] += 1

print(f"    Готово: {pkt_total:,} пакетів, {len(windows)} вікон")

rows = []
for wid in sorted(windows):
    w = windows[wid]
    sizes = w["sizes"]
    rows.append(
        {
            "window_id": wid,
            "packet_count": w["packet_count"],
            "total_bytes": w["total_bytes"],
            "avg_packet_size": np.mean(sizes) if sizes else 0,
            "unique_ips": len(w["src_ips"]),
            "syn_count": w["syn_count"],
            "udp_count": w["udp_count"],
            "bytes_per_sec": w["total_bytes"] / WINDOW_SIZE,
            "target": label,
        }
    )
return rows
```

```
# ---------------------------------------------------------------------------
# 2. Побудова датасету
# ---------------------------------------------------------------------------
def build_dataset() -> pd.DataFrame:
    """Створює датасет з двох pcap-файлів і зберігає у CSV."""
    if DATASET_CSV.exists():
        print(f"Датасет {DATASET_CSV} вже існує, завантажуємо...")
        return pd.read_csv(DATASET_CSV)

    print("=" * 60)
    print("ЕТАП 1: Витяг ознак з pcap-файлів")
    print("=" * 60)

    rows = []
    rows.extend(extract_features(NORMAL_PCAP, label=0))
    rows.extend(extract_features(ATTACK_PCAP, label=1))

    df = pd.DataFrame(rows)
    df.to_csv(DATASET_CSV, index=False)
    print(f"\nДатасет збережено: {DATASET_CSV}  ({len(df)} рядків)")
    return df


# ---------------------------------------------------------------------------
# 3. Статистичне виявлення аномалій (Z-score)
# ---------------------------------------------------------------------------
def zscore_analysis(df: pd.DataFrame) -> pd.DataFrame:
    """Додає Z-score та мітку аномалії за порогом ±3."""
    print("\n" + "=" * 60)
    print("ЕТАП 2: Статистичне виявлення аномалій (Z-score)")
    print("=" * 60)

    mean = df["bytes_per_sec"].mean()
    std = df["bytes_per_sec"].std()
    df["z_score"] = (df["bytes_per_sec"] - mean) / std
    df["anomaly_z"] = (df["z_score"].abs() > 3).astype(int)

    total_anomalies = df["anomaly_z"].sum()
    print(f"  Знайдено аномалій (Z-score > 3): {total_anomalies}")
    return df


# ---------------------------------------------------------------------------
# 4. ML-модель – Random Forest
# ---------------------------------------------------------------------------
FEATURE_COLS = [
    "packet_count",
    "total_bytes",
    "avg_packet_size",
    "unique_ips",
    "syn_count",
    "udp_count",
    "bytes_per_sec",
]
```

```
def train_random_forest(df: pd.DataFrame):
    """Тренує RandomForestClassifier та повертає модель і метрики."""
    print("\n" + "=" * 60)
    print("ЕТАП 3: Навчання моделі Random Forest")
    print("=" * 60)

    X = df[FEATURE_COLS]
    y = df["target"]

    X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.3, random_state=42, stratify=y
    )
    print(f"  Train: {len(X_train)}, Test: {len(X_test)}")

    model = RandomForestClassifier(n_estimators=100, random_state=42)
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)
    y_proba = model.predict_proba(X_test)[:, 1]

    print("\n  Confusion Matrix:")
    print(confusion_matrix(y_test, y_pred))
    print("\n  Classification Report:")
    print(classification_report(y_test, y_pred, target_names=["Normal",
"Attack"]))

    roc_auc = roc_auc_score(y_test, y_proba)
    print(f"  ROC-AUC: {roc_auc:.4f}")

    # Важливість ознак
    importances = pd.Series(model.feature_importances_, index=FEATURE_COLS)
    importances = importances.sort_values(ascending=False)
    print("\n  Важливість ознак:")
    for feat, imp in importances.items():
        print(f"    {feat}: {imp:.4f}")

    return model, X_test, y_test, y_pred, y_proba


# ---------------------------------------------------------------------------
# 5. Візуалізація
# ---------------------------------------------------------------------------
def plot_all(df: pd.DataFrame, y_test, y_pred, y_proba):
    """Генерує всі графіки для диплома."""
    print("\n" + "=" * 60)
    print("ЕТАП 4: Побудова графіків")
    print("=" * 60)

    PLOTS_DIR.mkdir(exist_ok=True)

    # --- Графік 1: Мережеве навантаження з ковзним середнім ---
    fig, ax = plt.subplots(figsize=(12, 5))
    ax.plot(df["window_id"], df["bytes_per_sec"], alpha=0.5,
label="Байт/сек")
    rolling = df["bytes_per_sec"].rolling(window=10, min_periods=1).mean()
```

```
    ax.plot(df["window_id"], rolling, color="red", linewidth=2, label="Ковзне
середнє (10)")
    ax.set_xlabel("Часове вікно")
    ax.set_ylabel("Байт/сек")
    ax.set_title("Навантаження мережі")
    ax.legend()
    fig.tight_layout()
    fig.savefig(PLOTS_DIR / "01_network_load.png", dpi=150)
    plt.close(fig)
    print("  -> plots/01_network_load.png")

    # --- Графік 2: Z-score ---
    fig, ax = plt.subplots(figsize=(12, 5))
    ax.plot(df["window_id"], df["z_score"], alpha=0.6, label="Z-score")
    ax.axhline(y=3, color="r", linestyle="--", label="Поріг +3")
    ax.axhline(y=-3, color="r", linestyle="--", label="Поріг -3")
    ax.set_xlabel("Часове вікно")
    ax.set_ylabel("Z-score")
    ax.set_title("Статистичне виявлення аномалій (Z-score)")
    ax.legend()
    fig.tight_layout()
    fig.savefig(PLOTS_DIR / "02_zscore.png", dpi=150)
    plt.close(fig)
    print("  -> plots/02_zscore.png")

    # --- Графік 3: Матриця помилок ---
    fig, ax = plt.subplots(figsize=(6, 5))
    ConfusionMatrixDisplay.from_predictions(
        y_test, y_pred, display_labels=["Normal", "Attack"], ax=ax,
cmap="Blues"
    )
    ax.set_title("Матриця помилок (Random Forest)")
    fig.tight_layout()
    fig.savefig(PLOTS_DIR / "03_confusion_matrix.png", dpi=150)
    plt.close(fig)
    print("  -> plots/03_confusion_matrix.png")

    # --- Графік 4: ROC-крива ---
    fpr, tpr, _ = roc_curve(y_test, y_proba)
    roc_auc = roc_auc_score(y_test, y_proba)
    fig, ax = plt.subplots(figsize=(6, 5))
    ax.plot(fpr, tpr, label=f"Random Forest (AUC = {roc_auc:.3f})")
    ax.plot([0, 1], [0, 1], "k--", alpha=0.4)
    ax.set_xlabel("False Positive Rate")
    ax.set_ylabel("True Positive Rate")
    ax.set_title("ROC-крива")
    ax.legend()
    fig.tight_layout()
    fig.savefig(PLOTS_DIR / "04_roc_curve.png", dpi=150)
    plt.close(fig)
    print("  -> plots/04_roc_curve.png")

    # --- Графік 5: Розподіл пакетів за класами ---
    fig, axes = plt.subplots(2, 2, figsize=(12, 10))
    for ax, col in zip(axes.flat, ["packet_count", "bytes_per_sec",
"syn_count", "udp_count"]):
        df[df["target"] == 0][col].hist(ax=ax, bins=50, alpha=0.6,
label="Normal")
        df[df["target"] == 1][col].hist(ax=ax, bins=50, alpha=0.6,
label="Attack")
        ax.set_title(col)
```

```
        ax.legend()
    fig.suptitle("Розподіл ознак за класами")
    fig.tight_layout()
    fig.savefig(PLOTS_DIR / "05_feature_distributions.png", dpi=150)
    plt.close(fig)
    print("  -> plots/05_feature_distributions.png")


# ---------------------------------------------------------------------------
# Main
# ---------------------------------------------------------------------------
def main():
    for pcap in (NORMAL_PCAP, ATTACK_PCAP):
        if not pcap.exists():
            print(f"ПОМИЛКА: файл {pcap} не знайдено!")
            sys.exit(1)

    df = build_dataset()
    print(f"\nДатасет: {df.shape[0]} рядків, {df.shape[1]} стовпців")
    print(df.describe())

    df = zscore_analysis(df)

    model, X_test, y_test, y_pred, y_proba = train_random_forest(df)

    plot_all(df, y_test, y_pred, y_proba)

    print("\n" + "=" * 60)
    print("ГОТОВО! Результати збережено:")
    print(f"  Датасет:  {DATASET_CSV}")
    print(f"  Графіки:  {PLOTS_DIR}/")
    print("=" * 60)


if __name__ == "__main__":
    main()
```