Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

Комп'ютерна система аналізу інтернет-трафіку
Назва теми

Рівень вищої освіти *перший (бакалаврський)*

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *123 «Комп'ютерна інженерія»*
Шифр, назва

Освітня програма *«Комп'ютерна інженерія та програмування»*
Назва

Шифр КвРКІ 022113.22.04.18 ПЗ

Виконав здобувач IV курсу, група КІ2-22-4 — Ілля УСТІНОВ
Підпис — Ініціали, прізвище

Керівник доктор техн. наук, професор — Євген ФЕДОРОВ
Науковий ступінь, учене звання — Підпис — Ініціали, прізвище

Нормоконтролер канд.фіз.-мат.наук, доц. — Тетяна КИСІЛЬ
Науковий ступінь, учене звання — Підпис — Ініціали, прізвище

До захисту допускаю:
завідувач кафедри КІІС — Ольга ПАВЛОВА
Підпис — Ініціали, прізвище
«04» червня 2026 р.
дата

Хмельницький 2026

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

Кафедра КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

Рівень вищої освіти ПЕРШИЙ (БАКАЛАВРСЬКИЙ)

Галузь знань 12 ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ

Спеціальність 123 КОМП'ЮТЕРНА ІНЖЕНЕРІЯ

Освітня програма «КОМП'ЮТЕРНА ІНЖЕНЕРІЯ ТА ПРОГРАМУВАННЯ»

ЗАТВЕРДЖУЮ
Завідувачка кафедри КІІС

_____Ольга ПАВЛОВА

" 10 " 01 2026 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Устінову Іллі Петровичу
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) « Комп'ютерна система аналізу інтернет-трафіку »

Керівник проекту (роботи)Федоров Євген Євгенович, д.т.н., проф.
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 20.01.2026 р. № 7

2. Термін подання здобувачем роботи на кафедру 01.06.2026 р.

3. Вихідні дані до роботи Завдання на кваліфікаційну роботу

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Теоротичні основи аналізу інтернет-трафіку

Методи та засоби аналізу інтернет-трафіку

Проектування комп'ютерної системи аналізу інтренет-трафіку

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Алгоритм подій проведення аналізу даних інтернет трафіку

Класифікація інтернет-трафіку за основними ознаками

Тестування висновків системи

6. Консультанти розділів кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання при

7. Дата видачі завдання «_10_»_____01_____2026 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Прим
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	10.01.2026	викон
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.02.2026	викон
3	Робота над розділом 1 – дослідження предметної області та постановка задачі	01.03.2026	викон
4	Робота над розділом 2 – вибір методу та засоби аналізу інтернет трафіку	01.04.2026	викон
5	Робота над розділом 3 – проєктування системи аналізу інтернет трафіку	29.04.2026	викон
6	Оформлення пояснювальної записки згідно вимог	25.05.2026	викон
7	Попередній захист ВКР	26.05.2026	викон
8	Захист ВКР на засіданні ЕК	Червень 2026 року	

Злобувач _____ Ілля УСТІНОВ
Підпис Ім'я, ПРІЗВИЩЕ

Керівник кваліфікаційної роботи _____ Євген ФЕДОРОВ
Підпис Ім'я, ПРІЗВИЩЕ

№ рядка	формат	Позначення	Найменування	Кіл. листів	№ екз	Примітка
			Текстові документи			
1		КвРКІ 022113.22.04.18 ПЗ	Пояснювальна записка	60		
			Графічні матеріали			
2		КвРКІ 022113.22.04.18 ПЗ	Алгоритм подій	1		
			проведення аналізу даних			
			інтернет трафіку			
3		КвРКІ 022113.22.04.18 ПЗ	ЛістингКоду	1		
4		КвРКІ 022113.22.04.18 ПЗ	Тестування висновків	1		
			системи			

КвРКІ 022113.22.04.18 ПЗ

Зм	Арк	№ докум	Підпис	Дата
Розробив		Устінов		
Перевір.		Федоров		
Н. контр.		Кисіль		
Затв.		Павлова		

Відомість проекту

Літера	Аркуш	Аркушів
У	1	1

ХНУ, КІ2-22-4

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Комп'ютерна система аналізу інтернет-трафіку».

Автор роботи: Устінов Ілля Петрович

Керівник роботи: Федоров Євген Євгенович

Пояснювальна записка: 69 с., 13 рис., табл., 2 дод., 50 джерел.

Графічна частина: 2 креслення.

АЛГОРИТМИ, АНАЛІЗ ТРАФІКУ, DASHBOARD, ІНТЕРНЕТ-ТРАФІК, КОМП'ЮТЕРНА МЕРЕЖА, МЕРЕЖЕВІ ПРОТОКОЛИ, PYTHON, СИСТЕМА АНАЛІЗУ.

Кваліфікаційна робота бакалавра присвячена розробленню комп'ютерної системи аналізу інтернет-трафіку. Актуальність теми зумовлена необхідністю автоматизованого оброблення мережевих даних, контролю активності в комп'ютерних мережах, виявлення підозрілих подій і подання результатів аналізу у зрозумілому вигляді.

У роботі розглянуто проблему складності ручного аналізу великих обсягів мережевого трафіку, а також необхідність швидкого визначення активних IP-адрес, використовуваних протоколів, портів, обсягу переданих даних і потенційно небезпечних подій. Такі завдання є важливими для адміністрування мереж, моніторингу навантаження та підвищення рівня інформаційної безпеки.

Метою роботи є проєктування, реалізація та тестування програмного прототипу комп'ютерної системи аналізу інтернет-трафіку. Система забезпечує завантаження підготовлених мережевих даних, їх попередню обробку, розрахунок статистичних показників, виявлення підозрілих ознак, оцінювання рівня ризику подій і візуалізацію результатів у dashboard-інтерфейсі.

Підпис здобувача

30.05.2026

Дата

ЗМІСТ

Зм.	Арк.	№докум.	Підпис	Дата	КвРКІ 022113.22.04.18 ПЗ		
Виконав		Устінов Ілля			Комп'ютерна система аналізу інтернет-трафіку	Літера	Аркуш 2 / Аркушів 69
Перевір.		Федоров Євген				У	
Н.контр.		Тетяна КИСІЛЬ				ХНУ КІ2-22-4	
Затверд.		Ольга ПАВЛОВА		01.06			

ВСТУП

Сучасні комп’ютерні мережі є основою роботи більшості інформаційних систем. Через інтернет передаються вебзапити, файли, повідомлення, мультимедійний контент, службові дані та інформація хмарних сервісів. Кожна така дія формує інтернет-трафік, який відображає взаємодію між користувацькими пристроями, серверами, програмами та мережевим обладнанням.

Аналіз інтернет-трафіку має важливе значення для стабільної та безпечної роботи мережі. Він дозволяє визначати навантаження на канали зв’язку, знаходити найбільш активні IP-адреси, аналізувати використання протоколів і портів, а також виявляти нетипові або підозрілі події. Наприклад, різке збільшення кількості з’єднань, звернення до багатьох портів або передавання великого обсягу даних може свідчити про помилку в роботі системи або потенційну загрозу.

Актуальність теми пояснюється тим, що обсяги мережевого трафіку постійно зростають, а вручну аналізувати велику кількість мережевих записів складно й неефективно. Тому виникає потреба у програмних системах, які можуть автоматизувати обробку трафіку, будувати статистику, визначати підозрілі ознаки та подавати результати у зрозумілому вигляді.

У межах цієї роботи розглядається комп’ютерна система аналізу інтернет-трафіку. Практична частина передбачає створення програмного прототипу мовою Python. Система працює з підготовленими мережевими даними у форматі CSV, виконує їх обробку, розраховує основні статистичні показники, визначає події з підвищеним рівнем ризику та відображає результати у вигляді dashboard-інтерфейсу.

Метою роботи є дослідження методів аналізу інтернет-трафіку та розробка програмного прототипу системи, яка забезпечує обробку мережевих даних, виявлення підозрілих подій і візуалізацію результатів.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Розглянути поняття інтернет-трафіку та його роль у комп'ютерних мережах.
2. Проаналізувати основні мережеві протоколи.
3. Описати класифікацію інтернет-трафіку та типові ознаки підозрілої активності.
4. Дослідити методи збору й аналізу мережевого трафіку.
5. Розглянути сучасні інструменти аналізу трафіку.
6. Сформулювати вимоги до системи аналізу інтернет-трафіку.
7. Спроєктувати архітектуру програмного прототипу.
8. Реалізувати основні алгоритми аналізу трафіку мовою Python.
9. Провести тестування системи на контрольному наборі даних.
10. Об'єктом дослідження є процес передавання та обробки інтернет-трафіку в комп'ютерних мережах.

Предметом дослідження є методи, алгоритми та програмні засоби аналізу інтернет-трафіку.

У роботі використовуються такі методи дослідження: аналіз технічної літератури, порівняння методів аналізу трафіку, системне проєктування, статистична обробка даних, програмна реалізація прототипу та його тестування.

Практичне значення роботи полягає у створенні навчального програмного прототипу системи аналізу інтернет-трафіку. Така система може бути використана для демонстрації базових принципів мережевої аналітики, оцінювання ризику та побудови dashboard-звітів.

Робота складається зі вступу, трьох розділів, висновків, списку використаних джерел і додатків. У першому розділі розглянуто теоретичні основи аналізу інтернет-трафіку. У другому розділі досліджено методи та засоби аналізу трафіку. У третьому розділі виконано проєктування, реалізацію та тестування програмної системи аналізу інтернет-трафіку.

1 ТЕОРЕТИЧНІ ОСНОВИ АНАЛІЗУ ІНТЕРНЕТ-ТРАФІКУ

1.1 Поняття інтернет-трафіку та його роль у комп’ютерних мережах

Інтернет-трафік є одним із основних понять у сфері комп’ютерних мереж. У загальному вигляді його можна визначити як сукупність даних, які передаються між різними пристроями через мережу. До таких пристроїв належать персональні комп’ютери, смартфони, сервери, маршрутизатори, мережеві сховища, хмарні сервіси, пристрої інтернету речей та інші елементи сучасної цифрової інфраструктури. Коли користувач відкриває вебсторінку, переглядає відео, надсилає повідомлення, завантажує файл або працює з онлайн-сервісом, у мережі створюється певний обсяг трафіку.

Інтернет-трафік не передається одним суцільним потоком. У комп’ютерних мережах дані поділяються на окремі частини, які називаються пакетами. Кожен пакет містить не тільки корисну інформацію, а й службові дані, потрібні для його доставлення до адресата. До таких службових даних належать адреса джерела, адреса призначення, відомості про протокол, порти, розмір пакета та інші технічні параметри. Саме ці характеристики мають велике значення під час аналізу трафіку, оскільки вони допомагають зрозуміти, які пристрої взаємодіють між собою, які сервіси використовуються та наскільки активною є мережева взаємодія [1].

Особливістю інтернет-трафіку є його багаторівнева структура. На нижчому рівні можна досліджувати окремі пакети, їхні заголовки, розміри, часові мітки та протоколи. На вищому рівні доцільно аналізувати вже не окремі пакети, а потоки даних, тобто групи пакетів, які належать до одного з’єднання або сеансу обміну. Наприклад, під час відкриття звичайної вебсторінки комп’ютер користувача може встановити кілька з’єднань із різними серверами, отримати HTML-документ, зображення, стилі, скрипти та інші елементи сторінки. Усі ці процеси формують мережевий трафік, який можна аналізувати

як детально, на рівні пакетів, так і узагальнено, на рівні активності користувача або пристрою.

Для практичного аналізу трафіку важливими є такі параметри, як IP-адреса джерела, IP-адреса призначення, порт джерела, порт призначення, протокол, час передавання, кількість пакетів, обсяг переданих даних і тривалість з'єднання. IP-адреси дозволяють визначити, які вузли беруть участь у мережевій взаємодії. Порти показують, до яких служб або застосунків відбувається звернення. Протокол дає змогу зрозуміти тип мережевої взаємодії. Обсяг даних і кількість пакетів допомагають оцінити інтенсивність трафіку та помітити можливі відхилення від звичайної поведінки мережі.

У комп'ютерних мережах інтернет-трафік виконує не лише технічну, а й аналітичну роль. З одного боку, він є засобом передавання даних між користувачами, серверами та сервісами. З іншого боку, трафік можна розглядати як джерело інформації про стан мережі. Якщо правильно зібрати та обробити мережеві дані, можна отримати уявлення про навантаження на канали зв'язку, активність окремих пристроїв, частоту використання певних протоколів, популярність сервісів і наявність підозрілих подій.

Аналіз інтернет-трафіку має велике значення для адміністрування мереж. Мережевий адміністратор може використовувати результати такого аналізу для контролю пропускної здатності, виявлення перевантажених ділянок мережі, пошуку помилок у налаштуванні обладнання та оцінювання якості роботи сервісів. Наприклад, якщо окремий пристрій раптово починає створювати дуже великий обсяг трафіку, це може бути пов'язано з оновленням програмного забезпечення, активним завантаженням файлів, неправильною роботою застосунку або потенційно небезпечною активністю. Без аналізу трафіку такі ситуації важко швидко помітити.

Не менш важливим є значення аналізу трафіку у сфері інформаційної безпеки. У мережевих даних можуть проявлятися ознаки сканування портів, спроб несанкціонованого доступу, поширення шкідливого програмного

забезпечення, взаємодії з підозрілими серверами або підготовки до атаки на певний ресурс. Системи виявлення та запобігання вторгненням використовують мережеві дані для пошуку таких ознак і формування попереджень для адміністратора [2]. При цьому аналіз може виконуватися як за відомими сигнатурами, так і за поведінковими або статистичними характеристиками.

Слід враховувати, що в сучасних умовах значна частина інтернет-трафіку є зашифрованою. Через це система аналізу не завжди може бачити зміст переданих повідомлень, файлів або запитів. Однак навіть у такому випадку корисними залишаються метадані: адреси, порти, часові мітки, обсяг переданих даних, кількість з'єднань, напрямок трафіку та частота звернень. Тому багато сучасних систем аналізу орієнтуються не тільки на вміст пакетів, а й на статистичні показники, структуру потоків і загальні закономірності поведінки мережі.

Рисунок 1.1 – Узагальнена схема передавання інтернет-трафіку в комп'ютерній мережі

У практичному сенсі аналіз інтернет-трафіку можна умовно поділити на кілька рівнів. Перший рівень - це базовий моніторинг, під час якого визначається загальний обсяг трафіку, швидкість передавання даних і завантаженість мережі. Другий рівень - це структурний аналіз, коли трафік групується за IP-адресами, портами, протоколами або напрямками передавання. Третій рівень пов'язаний із виявленням аномалій, тобто подій, які відрізняються від нормальної поведінки

мережі. Четвертий рівень можна віднести до безпекового аналізу, спрямованого на пошук ознак атак, шкідливої активності або порушень політик безпеки.

У межах цієї роботи доцільно розглядати інтернет-трафік як набір структурованих записів, які можна обробляти програмними засобами. Наприклад, один запис може містити час події, IP-адресу джерела, IP-адресу призначення, порт, протокол, кількість пакетів і обсяг переданих даних. На основі таких записів можна будувати статистику, визначати найактивніші IP-адреси, аналізувати розподіл протоколів, знаходити нетипові порти та оцінювати рівень ризику окремих подій. Такий підхід є достатньо простим для навчальної реалізації, але водночас відображає реальні принципи роботи систем мережевого аналізу.

Значення інтернет-трафіку в комп’ютерних мережах постійно зростає. Це пов’язано зі збільшенням кількості користувачів, підключених пристроїв, онлайн-сервісів і хмарних платформ. У таких умовах підвищуються вимоги до моніторингу мережевої активності, виявлення перевантажень і пошуку потенційно небезпечних подій [3]. Тому системи аналізу трафіку стають важливим елементом сучасної мережевої інфраструктури.

Інтернет-трафік можна розглядати не лише як технічний процес передавання даних, а й як важливе джерело інформації про роботу комп’ютерної мережі. Його аналіз дозволяє краще розуміти поведінку користувачів і пристроїв, контролювати навантаження, виявляти проблеми та знаходити потенційно небезпечні події. Саме тому побудова комп’ютерної системи аналізу інтернет-трафіку є актуальним і практично значущим завданням. Крім того, така система сприяє не тільки оперативному реагуванню на загрози, але й дозволяє прогнозувати майбутнє навантаження, оптимізуючи тим самим використання мережевих ресурсів

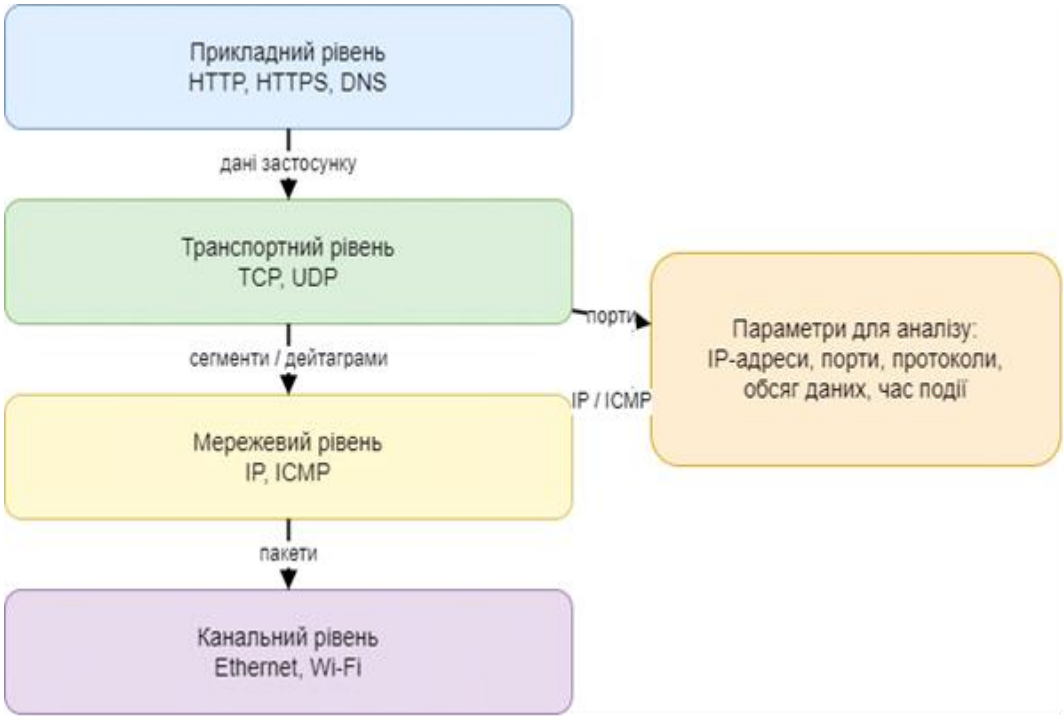


Рисунок 1.2 – Основні мережеві протоколи та їх значення для аналізу трафіку

Інтернет-трафік виникає не хаотично, а відповідно до певних правил, за якими пристрої обмінюються даними в мережі. Такі правила називаються мережевими протоколами. Вони визначають, як саме інформація передається, приймається, розпізнається та обробляється різними пристроями. Без протоколів неможливо було б забезпечити узгоджену роботу комп'ютерів, серверів, маршрутизаторів, мобільних пристроїв і різних онлайн-сервісів. Саме тому під час аналізу інтернет-трафіку протоколи мають важливе значення: вони допомагають зрозуміти, який тип мережевої взаємодії відбувається, які сервіси та чи є в трафіку ознаки нетипової активності.

Основою роботи сучасного інтернету є сімейство протоколів TCP/IP. Воно забезпечує передавання даних між вузлами мережі та визначає, як інформація розбивається на пакети, адресовується, доставляється до отримувача і потім обробляється. Для аналізу трафіку найбільше значення мають протоколи IP, TCP, UDP, ICMP, DNS, HTTP, HTTPS і TLS. Кожен із них виконує свою окрему

функцію, але в реальній мережевій взаємодії вони працюють разом і формують більшу частину щоденного інтернет-трафіку.

Протокол IP відповідає за адресацію та доставлення пакетів між пристроями. Завдяки IP-адресам можна визначити, звідки надійшов пакет і куди він має бути переданий. Для системи аналізу трафіку це один із головних параметрів, оскільки саме за IP-адресами можна побачити активні джерела трафіку, зовнішні та внутрішні вузли, а також нетипові або підозрілі з'єднання. Наприклад, якщо одна IP-адреса за короткий час створює дуже багато з'єднань, це може бути наслідком помилки, автоматизованої роботи програми або потенційно небезпечної активності [4].

На транспортному рівні важливими є протоколи TCP і UDP. Протокол TCP використовується тоді, коли потрібно забезпечити надійне передавання даних. Він встановлює з'єднання між двома вузлами, контролює доставлення пакетів і може повторно передавати дані, якщо вони були втрачені. Через TCP працює багато поширених сервісів: вебсайти, електронна пошта, передавання файлів, віддалений доступ. Для аналізу трафіку TCP корисний тим, що дозволяє досліджувати порти, стан з'єднання, кількість спроб підключення та інші службові ознаки. Наприклад, багато коротких спроб підключення до різних портів може вказувати на можливе сканування.

UDP працює інакше. Він не встановлює попереднього з'єднання і не гарантує, що кожен пакет обов'язково буде доставлений. Через це UDP є швидшим, але менш надійним порівняно з TCP. Його часто використовують там, де важлива швидкість: у DNS-запитах, потоковому відео, онлайн-іграх, голосовому зв'язку. Для аналізу трафіку UDP також є важливим, оскільки різке збільшення кількості UDP-пакетів або велика кількість звернень до одного порту може свідчити про перевантаження сервісу або активність.

Окремо варто згадати ICMP. Цей протокол використовується переважно для службових і діагностичних повідомлень у мережі. Наприклад, саме на ICMP базується робота команди ping, яка дозволяє перевірити, чи доступний певний

вузол. У звичайних умовах ICMP є корисним інструментом для перевірки роботи мережі. Проте надмірна кількість ICMP-повідомлень може потребувати уваги, оскільки іноді така активність пов’язана зі скануванням мережі або перевіркою доступних пристроїв.

Важливим протоколом прикладного рівня є DNS. Його завдання полягає в перетворенні доменних імен на IP-адреси. Коли користувач вводить адресу сайту в браузері, система спочатку має визначити, якій IP-адресі відповідає це доменне ім’я. Для аналізу інтернет-трафіку DNS-запити є досить інформативними, оскільки вони показують, до яких ресурсів звертається пристрій. Наприклад, часті запити до невідомих або підозрілих доменів можуть бути ознакою роботи небажаного програмного забезпечення або автоматизованої активності [5].

Для роботи з вебресурсами використовуються HTTP і HTTPS. HTTP передає дані у відкритому вигляді, тоді як HTTPS забезпечує захищене з’єднання. У сучасному інтернеті більшість вебсайтів працює саме через HTTPS, тому зміст переданих даних зазвичай є зашифрованим. Це ускладнює повний аналіз вмісту трафіку, але не робить аналіз марним. Навіть якщо система не бачить сам текст запиту або відповіді, вона може аналізувати супровідні дані: IP-адреси, порти, час з’єднання, обсяг переданої інформації, кількість запитів і тривалість сеансу.

TLS є протоколом, який забезпечує шифрування з’єднань і використовується, зокрема, у HTTPS. Його значення для сучасних мереж дуже велике, тому що він захищає дані користувачів від перехоплення та несанкціонованого перегляду. Водночас для систем аналізу трафіку поширення TLS створює певні обмеження. Якщо раніше можна було аналізувати більшу частину відкритого HTTP-трафіку, то зараз значна частина даних передається в зашифрованому вигляді. Тому дедалі більшого значення набуває аналіз не змісту пакетів, а їхніх метаданих, статистики, часових характеристик і поведінкових закономірностей [6].

Порти також відіграють важливу роль у дослідженні мережевої активності. Вони допомагають зрозуміти, з яким сервісом або застосунком пов’язане конкретне з’єднання. Наприклад, порт 80 зазвичай використовується для HTTP, порт 443 - для HTTPS, порт 53 - для DNS, порт 22 - для SSH. Якщо система фіксує багато звернень до службових або нетипових портів, це може бути підставою для додаткової перевірки. У межах програмного прототипу аналіз портів можна використати як один із простих способів виділення потенційно підозрілих подій.

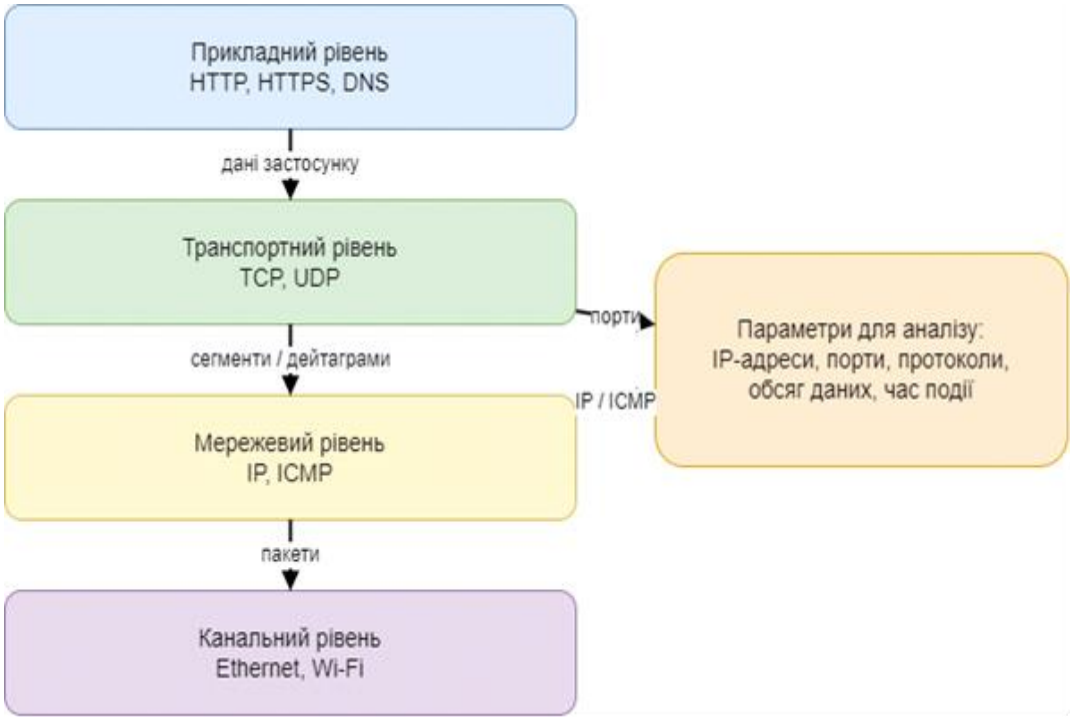


Рисунок 1.2 – Рівні мережевої взаємодії та приклади протоколів

З погляду системи аналізу трафіку кожен протокол дає певний набір ознак. IP дозволяє працювати з адресами джерела та призначення. TCP і UDP дають інформацію про порти, кількість з’єднань і характер обміну. DNS показує, до яких доменів звертається пристрій. HTTP і HTTPS дозволяють оцінити вебактивність за метаданими, обсягом даних і частотою звернень. ICMP дає уявлення про службові повідомлення та діагностичну активність. Усі ці параметри можна використовувати для статистики, фільтрації подій і формування висновків про стан мережі.

У практичній частині роботи протокол доцільно розглядати як один із ключових параметрів запису мережевого трафіку. Наприклад, у тестовому CSV-файлі кожен рядок може містити поле protocol, де зазначається TCP, UDP, ICMP або інший протокол. На основі цього поля система може будувати діаграму розподілу протоколів, визначати найпоширеніший тип трафіку, фільтрувати записи та враховувати протокол під час розрахунку умовного рівня ризику.

Якщо в наборі даних переважає TCP-трафік, це може бути звичайною ситуацією для мережі, де активно використовуються вебсервіси. Якщо ж різко зростає частка ICMP або UDP-трафіку, таку ситуацію варто перевірити додатково. Так само важливо звертати увагу на велику кількість звернень до одного порту або на ситуацію, коли одна IP-адреса звертається до багатьох різних портів. Такі ознаки можуть бути корисними для виявлення аномалій.

Мережеві протоколи є основою формування інтернет-трафіку й одночасно важливим джерелом інформації для його аналізу. Вони дають змогу зрозуміти, які сервіси використовуються в мережі, які пристрої взаємодіють між собою, яким є характер передавання даних і чи присутні ознаки підозрілої активності. Тому в комп’ютерній системі аналізу інтернет-трафіку протоколи, IP-адреси та порти доцільно розглядати як основні параметри для обробки, візуалізації та оцінювання ризику.

1.3 Класифікація інтернет-трафіку

Інтернет-трафік у комп’ютерних мережах має різну природу. Він може створюватися користувачами, програмами, серверами, мобільними пристроями, хмарними сервісами та мережевим обладнанням. Через це трафік відрізняється за призначенням, обсягом, напрямком передавання, частотою появи та рівнем потенційної небезпеки. Щоб такі дані можна було нормально аналізувати, їх потрібно не просто зібрати, а впорядкувати за певними ознаками.

Класифікація інтернет-трафіку - це поділ мережевих даних на окремі групи відповідно до їхніх характеристик. До таких характеристик можна віднести протокол, напрямок передавання, тип сервісу, IP-адреси учасників обміну, порти, обсяг переданих даних, час активності та рівень підозрілості. У системах аналізу трафіку класифікація використовується для побудови статистики, налаштування фільтрів, виявлення нетипових подій і формування підсумкових звітів.

Одним із найзручніших способів є класифікація за протоколами. У такому випадку мережеві записи поділяються залежно від того, який протокол використовується: TCP, UDP, ICMP, DNS, HTTP, HTTPS та інші. Цей підхід дозволяє швидко отримати загальне уявлення про характер мережевої активності. Наприклад, велика частка HTTPS-трафіку зазвичай пов’язана з активним використанням вебресурсів, значна кількість DNS-запитів може свідчити про часті звернення до різних доменів, а збільшення ICMP-повідомлень може бути пов’язане з діагностикою або перевіркою доступності вузлів [7].

Ще одним важливим критерієм є напрямок передавання трафіку. За цією ознакою можна виділити вхідний, вихідний і внутрішній трафік. Вхідний трафік надходить до локальної мережі або окремого пристрою із зовнішніх джерел. Вихідний трафік створюється внутрішніми пристроями та спрямовується до зовнішніх ресурсів. Внутрішній трафік передається між пристроями однієї локальної мережі. Такий поділ є корисним, оскільки допомагає зрозуміти, де саме виникає активність: поза мережею, усередині неї або між локальними вузлами.

За типом сервісу інтернет-трафік можна поділити на вебтрафік, DNS-трафік, поштовий трафік, трафік віддаленого доступу, потокове відео, файлові передавання, службові повідомлення та трафік хмарних сервісів. Така класифікація допомагає оцінити, які саме сервіси найбільше використовуються в мережі. Наприклад, потокове відео зазвичай створює значний обсяг даних, тоді як DNS-запити мають невеликий розмір, але можуть виконуватися дуже часто.

Окремо можна класифікувати трафік за інтенсивністю. Для цього враховують кількість пакетів, обсяг переданих даних, частоту з'єднань і тривалість активності. За цими ознаками трафік можна умовно поділити на звичайний, підвищений і надмірний. Наприклад, велика кількість коротких з'єднань може бути ознакою автоматизованої роботи програми, а дуже великий обсяг переданих даних може свідчити про завантаження файлів, резервне копіювання або потенційно небажане передавання інформації.

Важливе значення має також класифікація за портами. Порт допомагає зрозуміти, з яким сервісом пов'язане конкретне з'єднання. Наприклад, порт 443 зазвичай використовується для HTTPS, порт 53 - для DNS, порт 22 - для SSH. У багатьох випадках використання таких портів є звичайним. Проте якщо система фіксує багато звернень до різних портів одного вузла або бачить активність на нетипових портах, це може бути причиною для додаткової перевірки. Саме тому аналіз портів часто застосовується для виявлення можливого сканування або небажаних сервісів.

За рівнем ризику трафік можна умовно поділити на нормальний, підозрілий і потенційно небезпечний. Нормальний трафік відповідає типовій роботі мережі та не має очевидних ознак відхилення. Підозрілий трафік містить окремі нетипові характеристики, наприклад незвичні порти, різке збільшення кількості з'єднань або надмірний обсяг переданих даних. Потенційно небезпечний трафік може містити ознаки сканування, атаки, несанкціонованої взаємодії або іншої загрози. Такий поділ зручний для програмної системи, оскільки кожній події можна надати умовний рівень ризику.

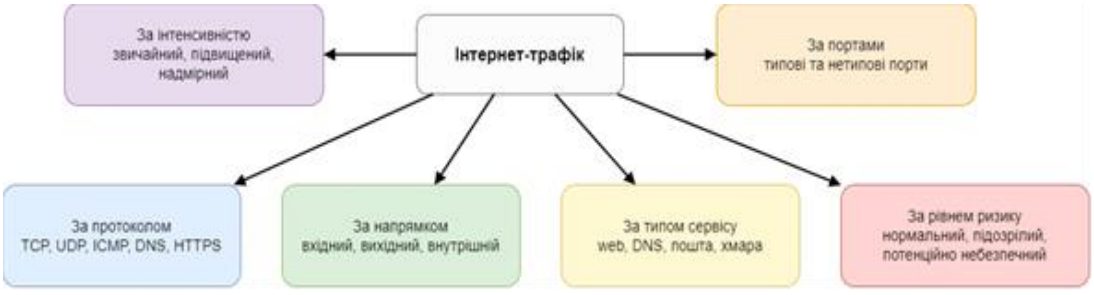


Рисунок 1.3 – Класифікація інтернет-трафіку за основними ознаками

Варто зазначити, що одна мережева подія може одночасно належати до кількох груп. Наприклад, DNS-запит може бути UDP-трафіком, службовим трафіком і вихідним трафіком. Якщо ж пристрій постійно звертається до великої кількості невідомих доменів, така подія може додатково вважатися підозрілою. Аналогічно HTTPS-з'єднання зазвичай є звичайною вебактивністю, але якщо воно пов'язане з нетиповою IP-адресою або дуже великим обсягом переданих даних, його також варто перевірити.

Для комп'ютерної системи аналізу інтернет-трафіку класифікація має практичне значення. Вона дозволяє не просто відобразити всі записи у вигляді великої таблиці, а зробити з них зрозумілу аналітичну картину. Наприклад, система може показати, яку частку займають TCP, UDP та ICMP, які IP-адреси є найактивнішими, які порти використовуються найчастіше та які записи мають підвищений рівень ризику. Без такого поділу дані були б менш наочними й складнішими для інтерпретації.

У практичній реалізації класифікацію можна виконувати за допомогою простих алгоритмів групування та фільтрації. Якщо система працює з CSV-файлом, кожен запис може містити поля protocol, src_ip, dst_ip, src_port, dst_port, bytes, packets і timestamp. На основі цих полів програма може групувати записи за протоколами, IP-адресами, портами та часовими інтервалами. Наприклад, у Python за допомогою бібліотеки pandas можна швидко підрахувати кількість записів для кожного протоколу, знайти найактивніші IP-адреси або визначити кількість унікальних портів для певного джерела.

Класифікація також є основою для побудови dashboard-інтерфейсу. Користувачеві зручніше працювати не з необробленим набором рядків, а з узагальненими блоками: діаграмою протоколів, таблицею активних IP-адрес, графіком активності за часом і списком підозрілих подій. Завдяки цьому технічні мережеві дані стають зрозумілішими та можуть використовуватися для оцінювання стану мережі.

Водночас класифікація трафіку не завжди є простою. У сучасних мережах багато сервісів використовують шифрування, динамічні порти, хмарні платформи та мережі доставки контенту. Через це не завжди можна точно визначити тип сервісу лише за портом або IP-адресою. Наприклад, через порт 443 може передаватися не тільки звичайний вебтрафік, а й трафік різних застосунків, API-запитів або хмарних сервісів. Тому в реальних системах аналізу зазвичай враховують не одну ознаку, а їх поєднання: протокол, порт, напрямок, обсяг, час активності та статистичні характеристики [8].

Для навчальної системи доцільно використати спрощений підхід до класифікації. Він не потребує складних моделей машинного навчання, але дає змогу показати основний принцип аналізу. Наприклад, трафік можна класифікувати за протоколом, портом, IP-адресою та рівнем ризику. Цього достатньо для демонстрації роботи системи, побудови графіків і пояснення результатів тестування. У майбутньому таку систему можна розширити, додавши аналіз DNS-запитів, підтримку PCAP-файлів, складніші правила або методи машинного навчання.

Класифікація інтернет-трафіку є важливим етапом його аналізу. Вона допомагає впорядкувати мережеві дані, виділити основні типи активності, оцінити навантаження на мережу та знайти потенційно підозрілі події. У межах розроблюваної комп’ютерної системи класифікація використовується для побудови статистики, фільтрації записів, оцінювання ризику та візуалізації результатів у dashboard-інтерфейсі.

1.4 Загрози та аномалії, які можна виявити за допомогою аналізу трафіку

Аналіз інтернет-трафіку важливий не лише для оцінювання навантаження на мережу, а й для виявлення потенційно небезпечних подій. У нормальному режимі комп’ютерна мережа має певну типову поведінку: користувачі відкривають вебсторінки, працюють із хмарними сервісами, надсилають DNS-

запити, передають файли, використовують месенджери або корпоративні застосунки. Якщо в такій поведінці з'являються різкі відхилення, це може свідчити про технічну помилку, неправильну роботу програми або спробу несанкціонованого впливу на мережу.

Під аномалією в мережевому трафіку можна розуміти подію або групу подій, які помітно відрізняються від звичайної роботи системи. При цьому не кожна аномалія обов'язково є атакою. Наприклад, різке збільшення обсягу трафіку може бути пов'язане з оновленням операційної системи, резервним копіюванням або завантаженням великих файлів. Однак такі ситуації все одно потребують уваги, оскільки вони можуть впливати на продуктивність мережі або приховувати небажану активність. Тому система аналізу трафіку має не тільки збирати дані, а й допомагати користувачу правильно їх інтерпретувати.

Однією з поширених підозрілих ознак є сканування портів. Суть такої активності полягає в тому, що один вузол намагається звернутися до багатьох портів іншого вузла або кількох пристроїв у мережі. З погляду інформаційної безпеки це важливо, оскільки сканування часто може бути підготовчим етапом перед атакою. За допомогою такого підходу зловмисник або автоматизована програма може перевіряти, які сервіси доступні, а потім намагатися використати знайдені слабкі місця. У трафіку це зазвичай проявляється як велика кількість коротких з'єднань від однієї IP-адреси до різних портів [9].

Ще однією характерною аномалією є надмірна кількість з'єднань за короткий проміжок часу. У звичайних умовах пристрій створює таку кількість з'єднань, яка відповідає роботі користувача або певного сервісу. Якщо ж одна IP-адреса починає генерувати сотні або тисячі звернень за короткий час, це може бути ознакою автоматизованої активності, програмної помилки, зараження пристрою або спроби перевантажити сервіс. Саме такі ознаки можуть бути корисними для виявлення DoS- або DDoS-подібної активності.

DoS-атака спрямована на те, щоб зробити певний сервіс недоступним для звичайних користувачів через надмірне навантаження. У випадку DDoS-атаки

джерелами такого навантаження є багато пристроїв одночасно. Для системи аналізу трафіку важливими ознаками можуть бути різке зростання кількості пакетів, велика кількість однотипних запитів, надмірне використання одного протоколу або значне збільшення трафіку до одного вузла. Сама система аналізу не обов'язково повинна зупиняти атаку, але вона може допомогти швидше помітити нестандартну ситуацію.

Окрему увагу потрібно приділити DNS-трафіку. DNS є необхідною службою для нормальної роботи інтернету, але водночас DNS-запити можуть містити важливі ознаки небажаної активності. Наприклад, заражений пристрій може часто звертатися до невідомих або випадково згенерованих доменів. Також DNS-запити можуть використовуватися для зв'язку зі шкідливими серверами або для прихованого передавання певної інформації. У навчальному прототипі не обов'язково реалізовувати глибокий аналіз доменних імен, але можна враховувати велику кількість DNS-запитів як окрему аналітичну ознаку [10].

Підозрілою може бути й активність на нетипових або ризикових портах. Наприклад, звернення до портів віддаленого адміністрування, службових портів або портів, які рідко використовуються у звичайній роботі користувача, може потребувати додаткової перевірки. Сам факт використання такого порту ще не доводить наявність атаки. Проте якщо ця ознака поєднується з великою кількістю з'єднань, незвичною IP-адресою призначення або великим обсягом даних, рівень ризику такої події зростає.

Ще одним важливим показником є незвично великий обсяг переданих даних. У мережі бувають ситуації, коли передавання великих файлів є цілком нормальним: наприклад, резервне копіювання, оновлення системи, робота з відеофайлами або хмарними сховищами. Однак якщо великий обсяг вихідного трафіку з'являється без очевидної причини, це може бути приводом для аналізу. Така ситуація може свідчити про неефективну роботу застосунку, неправильне використання мережі або потенційне виведення даних за її межі.

Аномалії також можуть проявлятися в часовій поведінці трафіку. Наприклад, якщо пристрій активно передає дані вночі або в інший період, коли користувач зазвичай не працює, це може бути підозрілою ознакою. Так само регулярні короткі звернення до одного зовнішнього вузла можуть свідчити про автоматизовану взаємодію. Тому під час аналізу важливо враховувати не лише обсяг або протокол трафіку, а й час появи подій. Саме для цього в dashboard-системі доцільно передбачити графік активності за часом.

У реальних системах захисту виявлення загроз зазвичай ґрунтується на поєднанні кількох підходів. Один підхід передбачає використання відомих правил або сигнатур. Інший орієнтується на пошук відхилень від звичайної поведінки. Для студентської роботи доцільно використати спрощену модель, у якій кожна подія отримує умовну оцінку ризику. Наприклад, якщо запис містить нетиповий порт, великий обсяг даних і належить до IP-адреси з надмірною кількістю з’єднань, система може позначити його як подію із середнім або високим рівнем ризику.

Водночас потрібно розуміти, що автоматичне виявлення аномалій не може бути абсолютно точним. Система може помилково позначити звичайну активність як підозрілу або, навпаки, не помітити складнішу загрозу. Такі ситуації називають хибними спрацюваннями та пропущеними спрацюваннями. Тому результати аналізу потрібно розглядати як допоміжний інструмент для прийняття рішення, а не як остаточний доказ атаки. Особливо це стосується навчального прототипу, який використовує прості правила, а не повноцінні промислові механізми виявлення вторгнень.

Для розроблюваної комп’ютерної системи аналізу інтернет-трафіку доцільно виділити кілька типів подій, які вона може виявляти. По-перше, це можлива ознака сканування портів, коли одна IP-адреса звертається до великої кількості різних портів. По-друге, це надмірна активність, якщо кількість записів від одного джерела перевищує встановлений поріг. По-третє, це використання

ризикового або нетипового порту. По-четверте, це великий обсяг переданих даних, який може свідчити про інтенсивне передавання інформації.

У практичній частині такі правила можна реалізувати через просту систему оцінювання ризику. Кожна підозріла ознака додає певну кількість балів до загального показника risk_score. Якщо підсумковий бал є низьким, подія вважається звичайною. Якщо бал середній, подія потребує уваги. Якщо бал високий, система позначає її як потенційно небезпечну. Такий підхід є зрозумілим для користувача, порівняно легко реалізується програмно та добре підходить для демонстраційного dashboard-інтерфейсу.

З погляду візуалізації важливо не лише знайти підозрілі події, а й подати їх у зручній формі. Наприклад, dashboard може показувати загальну кількість подій, кількість підозрілих записів, список IP-адрес із найбільшим ризиком, розподіл подій за рівнями ризику та таблицю з фільтрами. Це дозволяє користувачу швидко перейти від загальної картини до конкретних записів, які потребують додаткової уваги.

Аналіз інтернет-трафіку дає змогу виявляти різні типи аномалій і потенційних загроз. До них належать сканування портів, надмірна кількість з’єднань, підозрілі DNS-запити, активність на нетипових портах, різкі сплески трафіку та незвично великі обсяги передавання даних. У межах цієї роботи такі ознаки розглядаються з позиції моніторингу та захисту мережі. Це дозволяє побудувати навчальну систему, яка демонструє основні принципи виявлення підозрілої мережевої активності.

1.5 Постановка задачі

Після розгляду теоретичних основ інтернет-трафіку, основних мережевих протоколів, класифікації трафіку та типових аномалій доцільно сформулювати постановку задачі дослідження. Це потрібно для того, щоб чітко визначити, що

саме розглядається в роботі, яку проблему необхідно вирішити та який результат має бути отриманий у практичній частині.

У сучасних комп'ютерних мережах щоденно передається велика кількість даних. Ці дані можуть бути пов'язані з роботою вебсайтів, хмарних сервісів, месенджерів, DNS-запитів, передаванням файлів, службовими повідомленнями та іншими видами мережевої активності. Якщо такий трафік не аналізувати, адміністратору або користувачу складно зрозуміти, які пристрої створюють найбільше навантаження, які протоколи переважають, які порти використовуються та чи є в мережі потенційно підозрілі події.

Основна проблема полягає в тому, що необроблені мережеві дані у вигляді великої таблиці або журналу подій є складними для швидкого сприйняття. Користувачеві незручно вручну переглядати сотні або тисячі записів, порівнювати IP-адреси, порти, протоколи й обсяги переданих даних. Тому виникає потреба в комп'ютерній системі, яка може автоматизувати обробку таких даних, виконувати їх класифікацію, розраховувати статистичні показники, виявляти підозрілі ознаки та подавати результати у зрозумілому вигляді.

Об'єктом дослідження є процес передавання та обробки інтернет-трафіку в комп'ютерних мережах.

Предметом дослідження є методи, алгоритми та програмні засоби аналізу інтернет-трафіку, зокрема способи статистичної обробки мережевих записів, виявлення підозрілих подій і візуалізації результатів.

Метою роботи є проєктування та реалізація комп'ютерної системи аналізу інтернет-трафіку, яка забезпечує завантаження мережевих даних, їх попередню обробку, статистичний аналіз, виявлення потенційно підозрілих подій і подання результатів у dashboard-інтерфейсі.

Для досягнення поставленої мети необхідно виконати такі завдання:

1. Проаналізувати поняття інтернет-трафіку та його роль у комп'ютерних мережах.

2. Розглянути основні мережеві протоколи та визначити їх значення для аналізу трафіку.

3. Описати основні підходи до класифікації інтернет-трафіку.

4. Визначити типові загрози й аномалії, які можна виявити за допомогою аналізу мережевих даних.

5. Дослідити методи збору та статистичного аналізу інтернет-трафіку.

6. Спроєктувати архітектуру програмної системи та визначити структуру її основних модулів.

7. Розробити алгоритм аналізу інтернет-трафіку на основі статистичних показників і простих правил оцінювання ризику.

8. Реалізувати програмний прототип системи мовою Python із використанням бібліотек для обробки даних і побудови dashboard-інтерфейсу.

9. Провести тестування системи на підготовленому наборі мережевих даних і проаналізувати отримані результати.

У межах цієї роботи розроблювана система має навчально-практичний характер. Вона не призначена для перехоплення реального трафіку в мережі та не замінює професійні системи кіберзахисту. Основний акцент зроблено на обробці підготовлених мережевих даних у форматі CSV, розрахунку статистики, оцінюванні умовного рівня ризику та візуалізації результатів.

Очікуваним результатом роботи є програмний прототип комп'ютерної системи аналізу інтернет-трафіку. Така система повинна завантажувати файл із мережевими записами, перевіряти структуру даних, виконувати попередню обробку, будувати статистику за IP-адресами, портами й протоколами, визначати потенційно підозрілі події та відображати результати у вигляді таблиць, графіків і короткого аналітичного висновку.

1.6 Висновок до 1 розділу

У першому розділі було розглянуто теоретичні основи аналізу інтернет-трафіку. Було визначено, що інтернет-трафік є сукупністю даних, які передаються між пристроями в комп’ютерній мережі. Такий трафік формується під час звичайної роботи користувачів із вебсайтами, хмарними сервісами, файлами, месенджерами та іншими інтернет-ресурсами.

У розділі було показано, що інтернет-трафік має складну структуру й передається у вигляді пакетів або потоків даних. Для його аналізу важливими є IP-адреси, порти, протоколи, часові мітки, кількість пакетів і обсяг переданих даних. Саме ці параметри дозволяють оцінити активність пристроїв, визначити характер мережевої взаємодії та виявити можливі відхилення від нормальної роботи мережі.

Також було розглянуто основні мережеві протоколи, зокрема IP, TCP, UDP, ICMP, DNS, HTTP, HTTPS і TLS. Встановлено, що кожен із них має власне призначення та може надавати корисну інформацію для аналізу трафіку. Наприклад, IP-адреси допомагають визначати джерела й отримувачів даних, порти вказують на використані сервіси, а протоколи дозволяють зрозуміти тип мережевої активності.

2 МЕТОДИ ТА ЗАСОБИ АНАЛІЗУ ІНТЕРНЕТ-ТРАФІКУ

2.1. Методи збору мережевого трафіку

Збір мережевого трафіку є першим етапом роботи будь-якої системи аналізу. Перед тим як виконувати статистичну обробку, виявляти аномалії або будувати dashboard, потрібно отримати дані про мережеву активність. Від способу збору залежить, наскільки детальним буде подальший аналіз, які характеристики трафіку будуть доступні та які обмеження матиме система. У загальному вигляді можна виділити кілька основних підходів: захоплення пакетів, аналіз потоків, використання журналів подій, застосування мережевих сенсорів і робота з підготовленими файлами даних.

Найдетальнішим способом збору мережевого трафіку є захоплення пакетів. У цьому випадку система отримує окремі мережеві пакети, які проходять через певний мережевий інтерфейс або збережені у спеціальному файлі. Такий підхід дозволяє аналізувати заголовки пакетів, IP-адреси, порти, протоколи, часові мітки, розмір пакетів і деякі службові параметри. Для глибокого технічного аналізу це дуже корисно, оскільки дослідник отримує максимально деталізовану інформацію про мережеву взаємодію [12].

Однак пакетний аналіз має й певні обмеження. По-перше, захоплення реального трафіку може потребувати прав адміністратора. По-друге, такий підхід може створювати питання приватності, оскільки в мережевих пакетах можуть міститися службові або користувацькі дані. По-третє, за великого обсягу трафіку зберігання та обробка пакетів може вимагати значних ресурсів. Тому для навчальної роботи доцільніше не виконувати постійне перехоплення реального трафіку, а використовувати підготовлені файли або спрощені набори даних.

Поширеним форматом для збереження захопленого мережевого трафіку є PCAP. Такі файли можна створювати за допомогою спеціальних інструментів аналізу мережі, наприклад Wireshark. PCAP-файл містить збережені пакети, які потім можна повторно аналізувати без підключення до реальної мережі. Це

зручно для навчальних і дослідницьких завдань, оскільки дозволяє працювати з однаковим набором даних, порівнювати результати аналізу та не втручатися в роботу мережі.

Іншим підходом є аналіз потоків. Потік можна розглядати як узагальнений запис про взаємодію між двома вузлами мережі. Замість того щоб зберігати кожен пакет окремо, система зберігає агреговану інформацію: IP-адресу джерела, IP-адресу призначення, порти, протокол, час початку та завершення з'єднання, кількість пакетів і обсяг переданих даних. Такий підхід є менш детальним, ніж пакетний аналіз, але він значно зручніший для статистичної обробки та dashboard-аналітики.

Перевага flow-based підходу полягає в тому, що він зменшує обсяг даних і дозволяє швидше виконувати аналіз. Для виявлення багатьох аномалій не завжди потрібно бачити кожен пакет. Наприклад, для визначення найактивніших IP-адрес, найпоширеніших портів або сплесків активності достатньо мати агреговану інформацію. Саме тому в практичній частині роботи доцільно використовувати структуру даних, подібну до потокового запису, де кожен рядок описує окрему мережеву подію або з'єднання.

Ще одним джерелом даних є журнали подій. Багато мережевих систем, серверів, маршрутизаторів, міжмережевих екранів і систем виявлення вторгнень формують журнали, у яких записуються події мережевої активності. Такі журнали можуть містити інформацію про дозволені й заблоковані з'єднання, спроби доступу, DNS-запити, попередження безпеки або помилки роботи сервісів. Журнали не завжди дають повну картину трафіку, але вони добре підходять для аналізу подій і пошуку підозрілої активності [13].

У практичних системах часто використовуються мережеві сенсори. Сенсор - це програмний або апаратний компонент, який розміщується в певній точці мережі й збирає дані для подальшого аналізу. Наприклад, сенсор може бути встановлений на сервері, шлюзі, маршрутизаторі або окремому пристрої моніторингу. Він може збирати пакети, формувати потоки, створювати журнали

або передавати дані до центральної системи аналізу. У великих мережах сенсори дозволяють охопити різні сегменти інфраструктури.

Для збору трафіку також може використовуватися дзеркалювання портів. У цьому випадку мережевий комутатор копіює трафік з одного або кількох портів на окремий порт, до якого підключено систему моніторингу. Це дозволяє аналізувати трафік без прямого втручання в роботу основного каналу. Такий метод часто використовується в корпоративних мережах, але для студентської роботи його достатньо описати теоретично, оскільки налаштування мережевого обладнання не є основною метою цієї теми.

Збір трафіку може бути активним або пасивним. Активний підхід передбачає, що система сама створює певні запити або перевірки, наприклад тестує доступність вузлів. Пасивний підхід означає, що система лише спостерігає за вже наявним трафіком і не впливає на його формування. Для аналізу безпеки та моніторингу мережі пасивний підхід є більш прийнятним, оскільки він не змінює поведінку мережі та дозволяє отримувати реальні дані про її роботу.

У межах цієї роботи найбільш доречним є використання підготовленого набору мережевих даних. Такий підхід має кілька переваг. По-перше, він не потребує доступу до реальної мережевої інфраструктури. По-друге, він не створює ризиків для приватності користувачів. По-третє, тестовий набір можна сформувати так, щоб у ньому були як нормальні, так і підозрілі події. Це дозволяє перевірити роботу алгоритмів аналізу та продемонструвати можливості системи.

Для програмного прототипу на Python доцільно використовувати CSV або JSON-файл. У такому файлі кожен запис може містити час події, IP-адресу джерела, IP-адресу призначення, порти, протокол, кількість пакетів і обсяг переданих даних. Це не є повною заміною реального PCAP-аналізу, але для демонстрації принципів роботи системи такого формату достатньо. Крім того, CSV-файл легко обробляється бібліотекою pandas і добре підходить для побудови графіків у dashboard-інтерфейсі.

У подальшому систему можна розширити підтримкою PCAP-файлів. Наприклад, за допомогою бібліотек Scapy або PyShark можна зчитувати пакети, витягувати з них основні поля й перетворювати їх у табличну структуру. Після цього система могла б працювати з PCAP так само, як із CSV: виконувати групування, розраховувати статистику, визначати ризикові події та будувати графіки. Проте для базової реалізації достатньо стабільного й контрольованого CSV-формату.

Метод збору трафіку визначає можливості всієї системи аналізу. Пакетний підхід забезпечує максимальну деталізацію, але потребує більше ресурсів і має обмеження щодо приватності. Потоковий підхід є зручним для статистики й візуалізації. Журнали подій добре підходять для аналізу безпеки. Для студентського програмного прототипу найдоцільніше використати підготовлений файл мережевих даних, оскільки він дозволяє безпечно, стабільно й наочно продемонструвати основні принципи аналізу інтернет-трафіку.

2.2 Статистичні методи аналізу трафіку

Статистичні методи є одним із найпростіших і водночас найкорисніших підходів до аналізу інтернет-трафіку. Їхня основна ідея полягає в тому, щоб не розглядати кожен мережевий запис окремо, а узагальнювати дані за певними показниками. Такий підхід дозволяє швидко оцінити загальний стан мережі, визначити найбільш активні вузли, знайти пікові періоди навантаження та помітити події, які відрізняються від звичайної поведінки.

У практичних системах аналізу статистичні методи часто використовуються як перший рівень обробки трафіку. Вони не завжди дозволяють точно визначити тип загрози, але добре показують загальну картину мережевої активності. Наприклад, за допомогою статистики можна побачити, що певна IP-адреса створює набагато більше з’єднань, ніж інші, або що за короткий

час різко збільшилася кількість UDP-пакетів. Такі спостереження можуть бути підставою для подальшого аналізу [14].

До базових статистичних показників інтернет-трафіку належать загальна кількість записів, кількість пакетів, обсяг переданих даних, кількість унікальних IP-адрес, кількість використаних портів, розподіл протоколів і кількість подій за певний проміжок часу. Ці показники є зрозумілими для користувача та добре підходять для відображення в dashboard-інтерфейсі. Наприклад, користувач може одразу побачити, скільки записів було оброблено, який протокол переважає та сисистема позначила як підозрілі.

Одним із найважливіших показників є загальний обсяг трафіку. Він може вимірюватися в байтах, кілобайтах, мегабайтах або гігабайтах. За цим показником можна оцінити, наскільки інтенсивно використовується мережа. Якщо обсяг трафіку раптово зростає, це може бути пов’язано з нормальною активністю, наприклад завантаженням великих файлів, але також може свідчити про аномалію. Тому обсяг трафіку доцільно аналізувати не сам по собі, а разом з іншими характеристиками: джерелом, призначенням, протоколом і часом події.

Іншим важливим показником є кількість пакетів або записів. Якщо система працює з потоковими або підготовленими даними, кожен рядок може розглядатися як окрема подія або з’єднання. Підрахунок кількості таких записів дозволяє оцінити загальну активність мережі. Якщо кількість подій за певний проміжок часу значно перевищує звичайний рівень, це може бути ознакою автоматизованої активності, помилки сервісу або потенційної атаки.

Розподіл трафіку за протоколами допомагає зрозуміти, які типи мережевої взаємодії переважають. У типовій користувацькій мережі значну частину може становити TCP- або HTTPS-трафік. DNS-запити зазвичай мають невеликий обсяг, але можуть бути частими. ICMP-трафік у багатьох випадках використовується для діагностики й не повинен домінувати в загальному потоці. Якщо частка певного протоколу раптово збільшується, це може бути підставою для перевірки.

Аналіз IP-адрес є ще одним важливим статистичним напрямом. Система може визначати найактивніші IP-адреси джерела та призначення. Це дозволяє зрозуміти, які пристрої створюють найбільше трафіку або з якими зовнішніми ресурсами найчастіше взаємодіє мережа. Наприклад, якщо одна внутрішня IP-адреса надсилає велику кількість запитів до різних вузлів, це може бути нормальною роботою сервера, але також може потребувати додаткового аналізу. Саме тому статистика за IP-адресами є корисною для виявлення активних або нетипових вузлів.

Важливим є також аналіз портів. Порти дозволяють визначити, які сервіси використовуються в мережі. Наприклад, порти 80 і 443 зазвичай пов'язані з вебтрафіком, порт 53 - з DNS, порт 22 - з SSH. Якщо певна IP-адреса звертається до великої кількості різних портів, це може бути ознакою сканування. Якщо ж трафік концентрується на одному нетиповому порту, це також може бути причиною для перевірки. Статистичний аналіз портів є простим, але корисним механізмом для первинного виявлення підозрілої поведінки [15].

Окреме значення має часовий аналіз. Він передбачає групування подій за хвилинами, годинами або іншими проміжками часу. Завдяки цьому можна побачити, коли мережа була найбільш активною, чи були різкі сплески трафіку та чи повторюються певні події регулярно. Наприклад, якщо вночі з'являється велика кількість вихідних з'єднань, це може бути нормальною роботою автоматичного резервного копіювання, але також може бути підозрілою ознакою. Тому часові графіки є важливим елементом dashboard-системи.

Статистичні методи можуть бути як простими, так і складнішими. До простих належать підрахунок кількості, суми, середнього значення, максимуму й мінімуму. Наприклад, система може визначити середній розмір переданих даних для одного запису або максимальний обсяг трафіку між двома IP-адресами. До складніших методів можна віднести аналіз відхилень від середнього рівня, порівняння активності за різні періоди часу або пошук значень, які значно перевищують типові показники.

Для студентського програмного прототипу достатньо використати базові статистичні методи, оскільки вони добре демонструють основний принцип аналізу трафіку. Наприклад, система може виконувати такі операції: підрахунок загальної кількості записів, визначення кількості унікальних IP-адрес, побудова розподілу протоколів, визначення топ-10 активних IP-адрес, підрахунок найпопулярніших портів і формування списку подій з найбільшим обсягом переданих даних.

У мові Python такі завдання зручно реалізовувати за допомогою бібліотеки pandas. Вона дозволяє працювати з табличними даними, виконувати групування, фільтрацію, сортування та обчислення статистичних показників. Наприклад, для визначення кількості подій за протоколами можна згрупувати дані за полем protocol, а для пошуку найактивніших IP-адрес - за полем src_ip. Отримані результати можна передати до Plotly для побудови графіків у Streamlit-dashboard.

Візуалізація статистики має велике значення, оскільки таблиці з великою кількістю записів складно сприймати без додаткової обробки. Кругова або стовпчикова діаграма розподілу протоколів дозволяє швидко побачити структуру трафіку. Лінійний графік активності за часом показує сплески навантаження. Таблиця найактивніших IP-адрес допомагає знайти вузли, які створюють найбільше подій. Завдяки цьому dashboard перетворює технічні дані на зрозумілу аналітичну інформацію.

Разом з тим статистичні методи мають певні обмеження. Вони показують, що саме відбулося з точки зору кількісних показників, але не завжди пояснюють причину події. Наприклад, великий обсяг трафіку може бути як нормальною передачею файлів, так і потенційно небажаною активністю. Велика кількість DNS-запитів може бути пов’язана з активною роботою браузера або з підозрілим програмним забезпеченням. Тому статистику потрібно поєднувати з правилами оцінювання ризику, аналізом портів, IP-адрес і поведінкових ознак.

У межах розроблюваної системи статистичний аналіз буде виконувати роль базового аналітичного шару. Спочатку система зчитуватиме дані з файлу,

потім обчислюватиме основні показники, після чого результати передаватимуться до dashboard-інтерфейсу. На основі цих же даних працюватиме модуль виявлення підозрілих подій. Таким чином, статистичні методи стануть основою для подальшого оцінювання ризиків і формування висновків.

Статистичні методи аналізу інтернет-трафіку є важливим інструментом для первинної оцінки стану мережі. Вони дозволяють визначати обсяг трафіку, активність IP-адрес, розподіл протоколів, використання портів і часові закономірності. Хоча самі по собі статистичні показники не завжди дозволяють точно встановити наявність загрози, вони створюють основу для подальшого аналізу, виявлення аномалій і побудови зручної системи візуалізації.

2.3 Сигнатурний та евристичний підходи до виявлення загроз

Виявлення загроз у мережевому трафіку є одним із ключових завдань систем аналізу. Якщо статистичні методи дозволяють побачити загальну картину активності, то методи виявлення загроз допомагають зрозуміти, які події можуть бути небезпечними або потребують додаткової уваги. У практиці мережевої безпеки часто використовуються два базові підходи: сигнатурний та евристичний. Вони відрізняються принципом роботи, але можуть доповнювати один одного в межах однієї системи.

Сигнатурний підхід базується на використанні заздалегідь визначених правил або шаблонів. Сигнатура описує певну ознаку відомої загрози або небажаної активності. Якщо мережевий трафік відповідає такому шаблону, система формує попередження. Наприклад, сигнатура може враховувати конкретний порт, тип протоколу, характер запиту, послідовність пакетів або іншу ознаку, яка вже відома як потенційно небезпечна [16].

Основною перевагою сигнатурного підходу є його точність під час виявлення відомих загроз. Якщо правило сформульоване правильно, система

може швидко знаходити події, які збігаються з описаною сигнатурою. Саме тому цей підхід широко використовується в системах виявлення та запобігання вторгненням. Наприклад, правила можуть описувати спроби використання відомих вразливостей, підозрілі мережеві запити, звернення до небезпечних служб або характерні ознаки певних типів атак.

Водночас сигнатурний підхід має важливе обмеження: він добре працює лише з тими загрозами, які вже відомі та описані у правилах. Якщо з'являється нова атака або змінений варіант уже відомої загрози, система може її не виявити. Це означає, що сигнатурні бази потрібно регулярно оновлювати, а правила - перевіряти та вдосконалювати. Без цього ефективність такого підходу поступово зменшується.

Ще однією проблемою сигнатурного аналізу є можливість хибних спрацювань. Якщо правило сформульоване занадто широко, система може позначати як підозрілі навіть звичайні події. Якщо правило занадто вузьке, воно може пропускати реальні загрози. Тому якість сигнатур значною мірою визначає якість роботи всієї системи. У професійних інструментах правила створюються й підтримуються спеціалістами, а в навчальному прототипі можна використовувати спрощені правила, які демонструють сам принцип.

Евристичний підхід працює інакше. Він не обов'язково шукає точний збіг із відомою сигнатурою, а оцінює поведінку мережевих подій за певними логічними ознаками. Евристика - це практичне правило, яке допомагає зробити припущення про підозрілість події. Наприклад, якщо одна IP-адреса за короткий час звертається до великої кількості різних портів, це може бути ознакою сканування. Якщо один вузол передає незвично великий обсяг даних, це може бути приводом для додаткової перевірки.

Перевага евристичного підходу полягає в тому, що він може виявляти не лише відомі, а й нетипові або нові прояви підозрілої активності. Система не потребує точного опису конкретної атаки, а аналізує загальну поведінку трафіку. Це особливо корисно тоді, коли загроза не має чіткої сигнатури або коли її ознаки

змінюються. Евристичні правила також добре підходять для навчальних систем, оскільки їх можна зрозуміло пояснити й реалізувати програмно.

Проте евристичний підхід також не є ідеальним. Його головний недолік - більша ймовірність хибних спрацювань. Наприклад, звернення до багатьох портів може бути не атакою, а результатом роботи адміністративного інструмента. Великий обсяг трафіку може бути звичайним резервним копіюванням. Часті DNS-запити можуть бути пов’язані з нормальною роботою браузера або корпоративного застосунку. Тому евристичні результати потрібно розглядати як попередження, а не як остаточний доказ загрози [17].

У практичній системі аналізу інтернет-трафіку доцільно поєднувати сигнатурний та евристичний підходи. Наприклад, сигнатурні правила можуть використовуватися для виявлення чітко визначених подій: звернення до ризикових портів, використання небажаних протоколів або збіг із відомим шаблоном. Евристичні правила можуть оцінювати ширшу поведінку: кількість з’єднань, кількість унікальних портів, обсяг трафіку, частоту запитів і часові сплески активності. Такий комбінований підхід робить систему гнучкою.

Для студентського програмного прототипу можна реалізувати спрощену модель такого поєднання. Наприклад, до сигнатурних правил можна віднести перевірку на використання певних портів: 22, 23, 3389, 4444 або інших портів, які в конкретному контексті вважаються ризиковими. До евристичних правил можна віднести перевірку кількості унікальних портів для однієї IP-адреси, перевищення порогу обсягу даних або надмірну кількість записів від джерела.

На основі цих правил система може формувати умовну оцінку ризику. Якщо подія відповідає одному правилу, вона отримує невелику кількість балів. Якщо одночасно спрацьовує кілька правил, загальний ризик зростає. Наприклад, якщо IP-адреса звертається до ризикового порту, передає великий обсяг даних і має багато з’єднань, така подія може отримати високий рівень ризику. Такий підхід легко пояснити й показати в dashboard-інтерфейсі.

Сигнатурний і евристичний підходи також відрізняються за складністю впровадження. Сигнатурний аналіз потребує набору правил, які потрібно зберігати, оновлювати та коректно застосовувати. Евристичний аналіз може бути простішим на початковому етапі, оскільки базується на загальних умовах. Наприклад, у Python-програмі можна реалізувати кілька функцій, які перевіряють окремі ознаки й додають бали до risk_score.

У межах навчальної системи важливо не створювати ілюзію повної безпеки. Прототип не буде професійною IDS/IPS-системою, але він зможе продемонструвати основні принципи аналізу. Його завдання - показати, як мережеві записи можна обробляти, як на основі правил визначати підозрілі події та як подавати результати користувачу. Такий підхід є достатнім для практичної частини студентської роботи.

Важливо також правильно інтерпретувати результати. Якщо система позначає подію як підозрілу, це означає, що вона має певні ознаки ризику, але не обов’язково є реальною атакою. Остаточна оцінка повинна враховувати контекст: тип мережі, призначення пристрою, час події, характер сервісу й інші фактори. Наприклад, активність на порту 22 може бути нормальною для адміністративного сервера, але підозрілою для звичайного комп’ютера.

Сигнатурний та евристичний підходи є важливими методами виявлення загроз у мережевому трафіку. Сигнатурний підхід ефективний для відомих загроз, але залежить від якості та актуальності правил. Евристичний підхід є гнучкішим і дозволяє знаходити нетипову поведінку, але може частіше давати хибні спрацювання. Для розроблюваного програмного прототипу доцільно використати їх спрощене поєднання у вигляді правил оцінювання ризику, що дозволить наочно продемонструвати принципи виявлення підозрілих подій у трафіку.

2.4 Інструменти аналізу трафіку

Для аналізу інтернет-трафіку використовуються різні програмні засоби. Вони відрізняються призначенням, рівнем деталізації, способом обробки даних і форматом результатів. Одні інструменти орієнтовані на детальний перегляд пакетів, інші - на моніторинг мережевих подій, виявлення вторгнень або формування журналів безпеки. У межах цієї роботи доцільно розглянути чотири відомі інструменти: Wireshark, Zeek, Suricata та Snort. Вони часто використовуються в навчанні, адмініструванні мереж і практиці кібербезпеки.

Wireshark є одним із найпоширеніших інструментів для аналізу мережевих пакетів. Його основне призначення - захоплення, перегляд і детальне дослідження пакетів, які проходять через мережевий інтерфейс або збережені у файлі. Wireshark дозволяє бачити структуру пакетів, протоколи, IP-адреси, порти, службові поля та інші технічні характеристики. Для навчальних цілей цей інструмент є особливо корисним, оскільки він наочно показує, як виглядає мережевий обмін на рівні окремих пакетів [18].

Перевагою Wireshark є висока деталізація. За його допомогою можна аналізувати окремі пакети, фільтрувати трафік за протоколами, адресами або портами, переглядати послідовність з’єднань і досліджувати особливості роботи різних мережевих сервісів. Наприклад, за допомогою Wireshark можна побачити DNS-запити, TCP-з’єднання, ICMP-повідомлення або HTTPS-сеанси. Це дає змогу краще зрозуміти, як саме формується інтернет-трафік.

Водночас Wireshark не завжди зручний для автоматизованого аналізу великих обсягів даних. Він більше підходить для ручного дослідження окремих ситуацій або навчального перегляду пакетів. Якщо потрібно постійно моніторити мережу, автоматично виявляти події або будувати великі dashboard-звіти, одного Wireshark може бути недостатньо. У такому випадку доцільніше використовувати інструменти, які формують журнали або автоматично аналізують мережеву активність.

Zeek є системою мережевого моніторингу, яка орієнтована не стільки на перегляд окремих пакетів, скільки на створення структурованих журналів подій. Zeek аналізує мережевий трафік і формує записи про з’єднання, DNS-запити, HTTP-активність, SSL/TLS-сеанси, файли та інші події. Такий підхід є зручним для подальшої обробки, оскільки результати можна аналізувати як таблиці або журнали [19].

Важливою перевагою Zeek є те, що він перетворює складний мережевий трафік на більш зрозумілі події. Наприклад, замість перегляду сотень пакетів можна отримати один запис про з’єднання: хто з ким взаємодіяв, який протокол використовувався, скільки байтів було передано, коли почалося й завершилося з’єднання. Такий формат добре підходить для статистичного аналізу, побудови графіків і виявлення аномалій. Саме тому ідея практичного прототипу цієї роботи близька до підходу Zeek: система також працює зі структурованими записами трафіку, а не з кожним пакетом окремо.

Suricata є системою виявлення та запобігання вторгненням, яка може аналізувати мережевий трафік у режимі IDS або IPS. Вона використовує правила для пошуку підозрілих подій, формує сповіщення та може створювати журнали у форматі EVE JSON. Suricata застосовується для виявлення відомих загроз, аналізу протоколів і контролю мережевої активності. Її особливістю є поєднання сигнатурного аналізу, підтримки сучасних протоколів і зручного формату подій для подальшої обробки [20].

Для цієї роботи Suricata є важливою як приклад професійного інструмента, який демонструє роль правил і сповіщень у системах аналізу трафіку. У студентському прототипі не потрібно реалізовувати повну IDS-систему, але можна використати спрощену ідею: якщо подія відповідає певним умовам, їй надається підвищений рівень ризику. Наприклад, якщо одна IP-адреса звертається до багатьох портів або використовує підозрілий порт, система може позначити таку подію як потенційно небезпечну.

Snort також є відомою системою виявлення вторгнень. Він використовує набір правил, які описують ознаки підозрілої або небажаної мережевої активності. Snort може працювати в різних режимах, зокрема як аналізатор пакетів, реєстратор трафіку або IDS-система. Його часто використовують для навчання принципам сигнатурного аналізу, оскільки правила Snort демонструють, як можна формально описати певні мережеві події [21].

Порівняно з Wireshark, Snort і Suricata більше орієнтовані на виявлення загроз, а не на ручне дослідження кожного пакета. Вони автоматично аналізують трафік і формують попередження, якщо знаходять збіг із правилом. Це робить їх корисними для безпекового моніторингу. Проте для новачка такі системи можуть бути складнішими, оскільки потребують налаштування правил, розуміння форматів журналів і правильної інтерпретації сповіщень.

Якщо порівнювати ці інструменти між собою, можна зазначити, що кожен із них має свою роль. Wireshark найкраще підходить для детального ручного аналізу пакетів. Zeek зручний для формування структурованих журналів і дослідження мережевої поведінки. Suricata та Snort орієнтовані на виявлення вторгнень і роботу з правилами.

Таблиця 2.1 – Порівняння інструментів аналізу інтернет-трафіку

Інструмент	Основне призначення	Тип аналізу	Можливе використання в роботі
Wireshark	Детальний перегляд і аналіз мережевих пакетів	Пакетний аналіз	Використовується як приклад інструмента для ручного дослідження пакетів і протоколів

Zeek	Моніторинг мережевої активності	Аналіз мережевих подій	Використовується як приклад підходу, де трафік перетворюється на структуровані записи
Suricata	Виявлення та запобігання мережевим вторгненням	Сигнатурний і поведінковий аналіз	Використовується як приклад системи, що застосовує правила для виявлення підозрілих подій
Snort	Виявлення вторгнень і аналіз трафіку за правилами	Сигнатурний аналіз	Використовується як приклад IDS-системи з правиловим підходом до аналізу трафіку

У контексті розроблюваної системи ці інструменти можна розглядати як джерело ідей для архітектури. Від Wireshark можна взяти ідею аналізу базових параметрів пакетів: адрес, портів, протоколів і розміру. Від Zeek - ідею перетворення трафіку на структуровані записи, зручні для обробки. Від Suricata та Snort - ідею правил, які дозволяють позначати підозрілі події. Таким чином, студентський прототип не повторює повністю жоден із цих інструментів, але використовує їхні загальні принципи.

Wireshark, Zeek, Suricata та Snort є важливими прикладами інструментів, які використовуються для аналізу інтернет-трафіку та виявлення підозрілої активності. Вони мають різне призначення: від детального перегляду пакетів до автоматичного формування сповіщень про загрози. Для цієї роботи вони є не лише джерелом теоретичного аналізу, а й основою для розуміння того, які функції варто реалізувати у власному програмному прототипі.

2.5 Висновок до 2 розділу

У другому розділі було розглянуто методи та засоби аналізу інтернет-трафіку. Було встановлено, що перед виконанням аналізу необхідно отримати дані про мережеву активність. Для цього можуть використовуватися різні підходи: захоплення пакетів, аналіз потоків, журнали подій, мережеві сенсори або підготовлені файли з даними.

Було визначено, що пакетний аналіз забезпечує найбільш детальну інформацію про трафік, оскільки дозволяє досліджувати окремі пакети. Проте такий підхід може бути складним для навчальної реалізації, потребує додаткових прав доступу та може створювати питання конфіденційності. Саме тому для практичної частини роботи доцільно використовувати підготовлений CSV-файл із мережевими записами.

У розділі також було розглянуто статистичні методи аналізу трафіку. Вони дозволяють визначати загальну кількість записів, обсяг переданих даних, кількість унікальних IP-адрес, розподіл протоколів, найактивніші джерела трафіку, популярні порти та активність за часом. Такі методи є достатньо простими для реалізації.

Окремо було проаналізовано сигнатурний та евристичний підходи до виявлення загроз. Сигнатурний підхід базується на пошуку збігів із відомими правилами або шаблонами, а евристичний - на оцінюванні поведінки трафіку за певними ознаками. Для розроблюваної системи доцільним є використання їх спрощеного поєднання: перевірки ризикових портів, великих обсягів даних, великої кількості пакетів і можливого сканування портів.

Також у розділі було розглянуто інструменти Wireshark, Zeek, Suricata та Snort. Було визначено, що Wireshark підходить для детального ручного аналізу пакетів, Zeek - для формування структурованих журналів мережевої активності, а Suricata та Snort - для виявлення підозрілих подій за допомогою правил.

3 ПРОЄКТУВАННЯ КОМП'ЮТЕРНОЇ СИСТЕМИ АНАЛІЗУ ІНТЕРНЕТ-ТРАФІКУ

3.1 Вимоги до комп'ютерної системи аналізу трафіку

Проєктування комп'ютерної системи аналізу інтернет-трафіку доцільно починати з визначення вимог. Вимоги описують, які функції має виконувати система, з якими даними вона працюватиме, як користувач буде взаємодіяти з програмою та які обмеження потрібно врахувати під час реалізації. Чітке формулювання вимог дозволяє уникнути зайвої складності й зосередитися на тих можливостях, які справді потрібні для досягнення мети роботи.

У межах цієї роботи система розглядається як навчальний програмний прототип. Вона не призначена для заміни професійних засобів моніторингу, систем виявлення вторгнень або комплексних платформ кібербезпеки. Її головне завдання - продемонструвати базові принципи аналізу інтернет-трафіку: завантаження мережевих даних, їх перевірку, попередню обробку, статистичний аналіз, виявлення підозрілих подій і візуалізацію результатів у зручному dashboard-інтерфейсі.

Основною функціональною вимогою є можливість завантаження файлу з мережевими даними. Для базової реалізації доцільно використати формат CSV, оскільки він має просту табличну структуру, легко створюється, редагується та обробляється засобами Python. Кожен рядок такого файлу може описувати окрему мережеву подію або узагальнений запис про з'єднання. У майбутньому систему можна розширити підтримкою JSON або PCAP, однак для навчального прототипу CSV є найбільш зручним [22].

Наступною важливою вимогою є перевірка структури вхідних даних. Система повинна визначати, чи містить завантажений файл необхідні поля: час події, IP-адресу джерела, IP-адресу призначення, порт джерела, порт призначення, протокол, кількість пакетів і обсяг переданих даних. Якщо частина обов'язкових полів відсутня, програма має повідомити користувача про

помилку. Це потрібно для того, щоб уникнути некоректного аналізу та неправильних результатів.

Ще однією функціональною вимогою є попередня обробка даних. На цьому етапі система має приводити дані до єдиного формату: перетворювати часові значення, перевіряти числові поля, видаляти або ігнорувати некоректні записи, уніфікувати назви протоколів. Наприклад, значення tcp, Tcp і TCP потрібно приводити до одного вигляду - TCP. Така обробка спрощує подальше групування, побудову статистики та роботу алгоритмів оцінювання ризику.

Система повинна виконувати статистичний аналіз трафіку. До основних показників належать загальна кількість записів, кількість унікальних IP-адрес, загальний обсяг переданих даних, кількість використаних портів, найпоширеніший протокол, розподіл трафіку за протоколами, активність за часом, топ IP-адрес і топ портів. Ці показники дозволяють користувачу швидко оцінити загальну картину мережевої активності та зрозуміти, які вузли або сервіси створюють найбільше навантаження.

Окремою вимогою є виявлення підозрілих подій. Для навчального прототипу достатньо використати прості правила. Наприклад, система може позначати подію як підозрілу, якщо використовується ризиковий порт, якщо обсяг переданих даних перевищує встановлений поріг або якщо одна IP-адреса звертається до великої кількості різних портів. Такі правила не є повноцінною заміною професійних механізмів виявлення вторгнень, але вони добре демонструють базову логіку аналізу ризиків [23].

Також система повинна розраховувати умовну оцінку ризику. Для кожного запису може визначатися показник risk_score у межах від 0 до 100 балів. На основі цього показника події поділяються на три категорії: низький, середній і високий рівень ризику. Такий підхід є зручним для користувача, оскільки дозволяє швидко відокремити звичайні записи від тих, які потребують додаткової уваги.

Важливою вимогою є наявність dashboard-інтерфейсу. Результати аналізу мають подаватися не лише у вигляді таблиці, а й через графіки, картки показників і короткий текстовий висновок. Доцільно передбачити картки з основними метриками, діаграму розподілу протоколів, графік активності за часом, таблицю мережевих подій, окремий список підозрілих записів і фільтри. Такий інтерфейс робить систему зручною для сприйняття та демонстрації.

Фільтрація є ще однією важливою функцією. Користувач повинен мати змогу відбирати записи за протоколом, IP-адресою, портом або рівнем ризику. Наприклад, можна переглянути лише TCP-трафік, лише події з високим ризиком або записи, пов’язані з конкретною IP-адресою. Це дозволяє не переглядати вручну весь набір даних, а швидко знаходити потрібні події.

Крім функціональних вимог, потрібно врахувати нефункціональні. Першою з них є простота використання. Система має запускатися локально та не вимагати складного налаштування. Саме тому для реалізації доцільно використати Python і Streamlit, оскільки такий підхід дозволяє створити зрозумілий dashboard без окремої розробки frontend і backend.

Другою нефункціональною вимогою є достатня швидкодія для невеликих і середніх наборів даних. Оскільки система має навчальний характер, вона не розрахована на обробку великих потоків трафіку в режимі реального часу. Проте вона повинна стабільно працювати з тестовими файлами, які містять сотні або кілька тисяч записів. Для такого обсягу можливостей Python, pandas і Streamlit достатньо.

Третьою вимогою є безпечність роботи з даними. У межах цієї роботи система не повинна виконувати несанкціоноване перехоплення реального мережевого трафіку. Вона працює з підготовленими файлами, що дозволяє уникнути обробки конфіденційної інформації користувачів. Такий підхід є більш коректним для навчального проєкту та спрощує тестування системи.

Четвертою нефункціональною вимогою є модульність. Програму бажано будувати так, щоб окремі частини відповідали за конкретні завдання:

завантаження даних, попередню обробку, статистичний аналіз, виявлення аномалій, оцінювання ризику та формування звіту. Модульна структура спрощує розробку, тестування та подальше розширення системи.

П’ятою вимогою є наочність результатів. Користувач повинен отримувати не лише числові значення, а й візуальне подання даних. Графіки, таблиці, картки показників і короткий автоматичний висновок допомагають швидше зрозуміти результати аналізу. Це особливо важливо для навчальної роботи, де потрібно не тільки реалізувати алгоритми, а й показати їхню практичну дію.

Узагальнено вимоги до системи можна поділити на чотири групи: вимоги до вхідних даних, функціональні вимоги, нефункціональні вимоги та вимоги до результатів. Вхідні дані мають бути структурованими й містити основні параметри мережевих подій. Функціональні вимоги визначають, що саме система повинна робити. Нефункціональні вимоги описують якість роботи системи. Вимоги до результатів визначають, у якому вигляді користувач має отримувати підсумки аналізу.

Сформульовані вимоги дозволяють визначити межі майбутньої програмної системи. Вона має бути простою у використанні, працювати з підготовленими даними трафіку, виконувати статистичний аналіз, оцінювати ризик подій і відображати результати у вигляді dashboard-інтерфейсу. Такий набір можливостей є достатнім для студентської роботи й водночас логічно відповідає темі комп’ютерної системи аналізу інтернет-трафіку.

3.2 Алгоритм аналізу інтернет-трафіку

Після визначення вимог до комп’ютерної системи аналізу інтернет-трафіку необхідно описати її архітектуру. Архітектура показує, з яких основних частин складається система, як ці частини взаємодіють між собою та яким шляхом проходять дані від моменту завантаження файлу до отримання результатів аналізу. Для цієї роботи доцільно використати модульну архітектуру, оскільки

вона є зрозумілою, зручною для реалізації та легко розширюється в майбутньому.

Запропонована система розглядається як програмний прототип, реалізований мовою Python. Основна ідея полягає в тому, що користувач завантажує файл із мережевими даними, після чого програма виконує їх перевірку, обробку, статистичний аналіз, оцінювання ризику та візуалізацію результатів. Інтерфейс системи реалізується у вигляді dashboard-сторінки, де користувач може бачити основні показники, графіки, таблиці та список підозрілих подій.

Рисунок 3.1 – Архітектура програмної системи аналізу інтернет-трафіку

Першим компонентом системи є модуль завантаження даних. Його завдання полягає в тому, щоб прийняти файл від користувача та передати його до програми для подальшої обробки. У базовій версії системи використовується CSV-файл, оскільки цей формат є простим і добре підходить для табличних мережевих записів. Користувач завантажує файл через інтерфейс Streamlit, після чого система зчитує його за допомогою бібліотеки pandas. Якщо файл не завантажено, система може використовувати тестовий набір даних із папки проєкту [24].

Другим компонентом є модуль перевірки структури даних. Він аналізує, чи містить файл необхідні поля. Для запропонованої системи такими полями є timestamp, src_ip, dst_ip, src_port, dst_port, protocol, bytes і packets. Якщо хоча б одне з обов’язкових полів відсутнє, система повинна повідомити користувача

про помилку. Це дозволяє уникнути неправильних обчислень і підвищує надійність роботи програми.

Третім компонентом є модуль попередньої обробки даних. На цьому етапі система приводить інформацію до єдиного формату. Наприклад, часові значення перетворюються у формат дати й часу, числові поля - у числовий тип, а назви протоколів - до одного регістру. Також система може обробляти пропущені значення або відкидати некоректні записи. Попередня обробка є важливим етапом, тому що якість початкових даних безпосередньо впливає на правильність результатів.

Четвертим компонентом є модуль статистичного аналізу. Він обчислює основні показники мережевого трафіку: загальну кількість записів, кількість унікальних IP-адрес, загальний обсяг переданих даних, розподіл протоколів, найактивніші IP-адреси, найпоширеніші порти та активність за часом. Саме цей модуль формує основу для dashboard-аналітики, оскільки більшість графіків і карток показників будуються на його результатах.

П’ятим компонентом є модуль виявлення аномалій. Його завдання - знайти події, які можуть відрізнятися від звичайної поведінки. У межах цієї роботи доцільно реалізувати кілька простих правил: виявлення IP-адреси, яка звертається до великої кількості портів; визначення подій із надмірним обсягом даних; позначення з’єднань із ризиковими портами; виявлення надмірної кількості записів від одного джерела. Такий підхід не є складною системою машинного навчання, але достатньо добре демонструє принципи аналізу мережевих аномалій [25].

Шостим компонентом є модуль оцінювання ризику. Він використовує результати статистичного аналізу та правила виявлення аномалій для формування умовного показника risk_score. Кожній події нараховується певна кількість балів залежно від її ознак. Наприклад, використання ризикового порту може додати 25 балів, великий обсяг переданих даних - ще 25 балів, велика

кількість пакетів - 20 балів, а ознака можливого сканування портів - 30 балів. Після цього подія отримує рівень ризику: низький, середній або високий.

Сьомим компонентом є модуль візуалізації. Він відповідає за відображення результатів у dashboard-інтерфейсі. Для цього доцільно використати Streamlit і Plotly. Streamlit забезпечує швидке створення вебподібного інтерфейсу, а Plotly дозволяє будувати інтерактивні графіки. У dashboard можна розмістити картки з основними показниками, діаграму розподілу протоколів, графік активності за часом, таблицю найактивніших IP-адрес, список підозрілих подій і фільтри.

Восьмим компонентом є модуль формування звіту. Його завдання полягає у створенні короткого текстового висновку за результатами аналізу. Наприклад, система може автоматично повідомити, скільки записів було оброблено, який протокол переважає, скільки подій мають високий рівень ризику та яка IP-адреса є найактивнішою. Такий звіт не замінює повний аналіз, але допомагає користувачу швидко зрозуміти основні результати.

З погляду реалізації систему доцільно поділити на окремі файли або модулі. Головний файл app.py відповідатиме за запуск Streamlit-додатка та відображення інтерфейсу. Модуль loader.py може відповідати за завантаження даних, preprocessing.py - за очищення й підготовку, statistics.py - за розрахунок показників, anomaly.py - за виявлення аномалій, risk.py - за оцінювання ризику, а report.py - за формування текстового висновку. Така структура є простою й зрозумілою для студентського проєкту.

Архітектура системи не передбачає обов’язкового використання окремого backend-сервера або бази даних. Усі обчислення можуть виконуватися локально в межах Python-додатка. Це спрощує розробку, зменшує кількість залежностей і робить систему зручною для демонстрації. Дані не потрібно зберігати на сервері: користувач завантажує файл, система обробляє його під час поточної сесії та відображає результат.

Відсутність backend не є недоліком для навчального прототипу. Навпаки, такий підхід дозволяє зосередитися на головному завданні - аналізі трафіку.

Якщо в майбутньому систему потрібно буде використовувати для постійного моніторингу або збереження історії результатів, тоді можна додати серверну частину, базу даних і механізм авторизації. Проте в межах цієї роботи це не є необхідним.

Важливою перевагою запропонованої архітектури є її розширюваність. Наприклад, у майбутньому можна додати підтримку PCAP-файлів, імпорт журналів Zeek або Suricata, складніші правила ризику, машинне навчання для виявлення аномалій або експорт результатів у PDF-звіт. При цьому базова структура системи залишиться такою самою: дані завантажуються, обробляються, аналізуються та відображаються користувачу.

Отже, архітектура запропонованої системи базується на модульному підході. Вона включає модулі завантаження, перевірки, попередньої обробки, статистичного аналізу, виявлення аномалій, оцінювання ризику, візуалізації та формування звіту. Така структура відповідає поставленим вимогам, є зрозумілою для реалізації мовою Python і дозволяє створити практичний dashboard-прототип системи аналізу інтернет-трафіку.

3.3 Алгоритм аналізу інтернет-трафіку

Алгоритм аналізу інтернет-трафіку описує послідовність дій, які виконує система після отримання вхідних даних. Якщо архітектура показує загальну будову системи та її модулі, то алгоритм пояснює логіку роботи цих модулів у певному порядку. Для розроблюваного програмного прототипу алгоритм має бути достатньо простим, але водночас повинен охоплювати всі основні етапи: завантаження файлу, перевірку даних, статистичний аналіз, виявлення підозрілих ознак, оцінювання ризику та візуалізацію результатів.

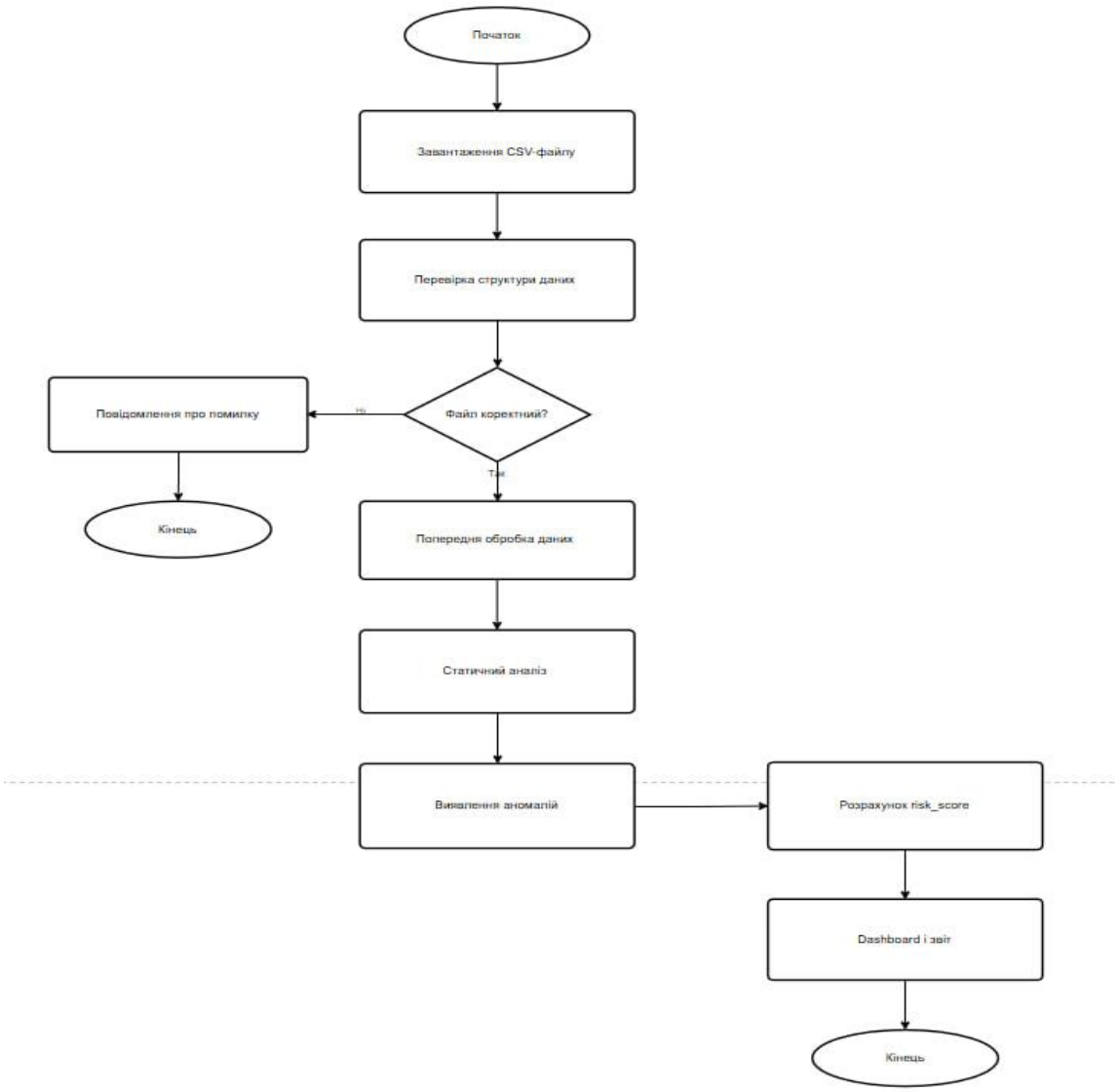


Рисунок 3.2 – Блок-схема алгоритму аналізу трафіку

Першим етапом роботи системи є запуск програмного прототипу. Користувач відкриває dashboard-інтерфейс, створений за допомогою Streamlit. На головній сторінці системи розміщується блок завантаження файлу, короткий опис призначення програми та вимоги до структури вхідних даних. Це потрібно для того, щоб користувач одразу розумів, який файл можна використовувати для аналізу.

Другим етапом є завантаження файлу з мережевими даними. У базовій реалізації використовується CSV-файл, де кожен рядок описує окрему мережеву

подію або з’єднання. Файл має містити такі поля: timestamp, src_ip, dst_ip, src_port, dst_port, protocol, bytes, packets. Після завантаження система зчитує файл у табличну структуру, з якою надалі працюють модулі аналізу.

Третім етапом є перевірка структури даних. Система порівнює фактичні назви колонок із переліком обов’язкових полів. Якщо всі необхідні поля присутні, обробка продовжується. Якщо певного поля немає, користувачу виводиться повідомлення про помилку. Така перевірка є важливою, оскільки подальший аналіз залежить від коректності вхідної структури. Наприклад, без поля protocol неможливо побудувати розподіл трафіку за протоколами, а без поля bytes - оцінити обсяг переданих даних.

Четвертий етап - попередня обробка даних. На цьому кроці система приводить значення до потрібних типів. Часові мітки перетворюються у формат дати й часу, порти, кількість пакетів і байти - у числовий формат, а назви протоколів уніфікуються. Наприклад, значення tcp, Tcp і TCP доцільно привести до одного вигляду - TCP. Також система може видаляти некоректні або порожні записи, якщо вони заважають аналізу.

П’ятий етап - розрахунок загальних статистичних показників. Система визначає загальну кількість записів, кількість унікальних IP-адрес, загальний обсяг переданих даних, кількість використаних портів і найпоширеніший протокол. Ці показники виводяться у вигляді інформаційних карток на dashboard-сторінці. Вони дозволяють користувачу швидко оцінити масштаб і загальний характер проаналізованого трафіку.

Шостий етап - групування даних. Система групує записи за протоколами, IP-адресами, портами та часовими інтервалами. Групування за протоколами дозволяє побудувати діаграму розподілу TCP, UDP, ICMP та інших типів трафіку. Групування за IP-адресами допомагає знайти найактивніші джерела та призначення. Групування за портами дозволяє визначити, які сервіси використовуються найчастіше. Часове групування дає змогу побудувати графік активності.

Сьомий етап - виявлення потенційних аномалій. У межах цієї роботи доцільно використовувати набір простих правил. Наприклад, система перевіряє, чи звертається одна IP-адреса до великої кількості різних портів. Якщо кількість унікальних портів перевищує встановлений поріг, така активність може бути позначена як можлива ознака сканування. Також система може перевіряти, чи перевищує обсяг переданих даних певне значення, чи використовується ризиковий порт, чи створює одна IP-адреса надмірну кількість записів.

Восьмий етап - розрахунок умовної оцінки ризику. Для кожної події система обчислює показник risk_score. Він формується на основі окремих ознак. Наприклад, використання ризикового порту додає певну кількість балів, великий обсяг даних - додаткові бали, можлива ознака сканування портів - ще більшу кількість балів. Після цього значення обмежується діапазоном від 0 до 100. Такий підхід дозволяє формалізувати оцінювання подій і поділити їх за рівнем небезпеки.

Дев'ятий етап - визначення рівня ризику. На основі risk_score система присвоює кожній події одну з категорій: низький, середній або високий рівень ризику. Наприклад, події з оцінкою від 0 до 30 балів можна вважати низькоризиковими, від 31 до 70 - середніми, а від 71 до 100 - високоризиковими. Такий поділ є умовним, але він зрозумілий для користувача та добре підходить для демонстраційної системи.

Десятий етап - фільтрація та підготовка результатів до виведення. Після обчислення всіх показників система формує таблицю записів, де додаються нові поля: risk_score, risk_level, можлива ознака сканування або інша позначка аномалії. Користувач може застосовувати фільтри за протоколом, IP-адресою, портом або рівнем ризику. Це дозволяє швидко перейти від загального огляду до конкретних подій.

Одинадцятий етап - побудова dashboard-інтерфейсу. Система виводить основні результати у вигляді карток, графіків і таблиць. Наприклад, на сторінці можна показати загальну кількість записів, кількість підозрілих подій, діаграму

протоколів, графік активності за часом, топ IP-адрес, топ портів і таблицю подій з оцінкою ризику. Завдяки цьому користувач отримує не просто необроблені дані, а зрозумілу аналітичну картину.

Дванадцятий етап - формування короткого висновку. На основі результатів аналізу система може автоматично створити текстовий підсумок. У ньому зазначається, скільки записів було оброблено, який протокол переважає, скільки подій отримали середній або високий рівень ризику, яка IP-адреса є найактивнішою. Такий висновок корисний для швидкого сприйняття результатів і може бути використаний у звітній частині практичного розділу.

Важливо, що запропонований алгоритм є послідовним і зрозумілим. Він не потребує складної серверної інфраструктури або машинного навчання, але дозволяє реалізувати основні принципи системи аналізу трафіку. Саме така логіка добре підходить для студентської роботи, оскільки її можна пояснити теоретично, показати у вигляді блок-схеми та реалізувати програмно.

Окремо потрібно зазначити, що алгоритм працює з підготовленим набором даних. Це означає, що система не виконує постійного перехоплення реального трафіку в мережі. Такий підхід є безпечнішим і зручнішим для тестування. Користувач може багато разів запускати аналіз на одному й тому самому файлі, перевіряти результати, змінювати порогові значення та порівнювати поведінку системи.

Влгоритм аналізу інтернет-трафіку в запропонованій системі складається з послідовних етапів завантаження, перевірки, обробки, статистичного аналізу, виявлення аномалій, оцінювання ризику та візуалізації результатів. Він забезпечує логічний зв’язок між теоретичними методами аналізу трафіку та практичною реалізацією програмного прототипу на Python.

3.4 Структура даних для демонстраційної системи

Для роботи комп’ютерної системи аналізу інтернет-трафіку важливо правильно визначити структуру вхідних даних. Оскільки система має не лише зчитувати записи, а й виконувати статистичний аналіз, виявляти підозрілі події та будувати dashboard, кожен запис повинен містити набір основних параметрів мережевої активності. Від якості та повноти цих даних залежить коректність подальшої обробки.

У реальних умовах мережевий трафік може зберігатися в різних форматах. Наприклад, для детального аналізу пакетів часто використовуються PCAP-файли, журнали Zeek можуть містити структуровані записи про з’єднання, а Suricata може формувати події у форматі JSON. Проте для навчального програмного прототипу доцільно використати спрощений табличний формат CSV. Він є зручним для обробки в Python, легко створюється вручну або експортується з інших інструментів, а також добре підходить для демонстрації роботи алгоритмів аналізу [28].

У запропонованій системі кожен рядок CSV-файлу розглядається як окрема мережева подія або узагальнений запис про з’єднання. Такий підхід не відображає всієї складності реального пакетного аналізу, але дозволяє зберегти головні параметри, потрібні для статистики та оцінювання ризику. Для студентської роботи це є доцільним компромісом між реалістичністю і простотою реалізації.

Основними полями структури даних є: timestamp, src_ip, dst_ip, src_port, dst_port, protocol, bytes і packets. Ці поля дозволяють описати, коли відбулася подія, між якими вузлами відбувався обмін, який протокол використовувався, через які порти передавалися дані та який обсяг трафіку було зафіксовано.

Поле timestamp містить час події. Воно потрібне для аналізу активності за часовими інтервалами. Наприклад, за допомогою цього поля система може побудувати графік кількості подій за хвилинами або годинами, визначити

періоди пікового навантаження та виявити різкі сплески активності. Для коректної роботи це поле бажано зберігати у форматі, який Python може перетворити на тип дати й часу.

Поля src_ip і dst_ip описують IP-адресу джерела та IP-адресу призначення. Вони є одними з найважливіших параметрів для аналізу трафіку. За IP-адресою джерела можна визначити, який пристрій створює активність. За IP-адресою призначення можна встановити, до якого вузла або сервісу спрямовується трафік. Система може використовувати ці поля для побудови списку найактивніших IP-адрес і виявлення джерел із підозрілою поведінкою.

Поля src_port і dst_port містять порти джерела та призначення. У багатьох випадках саме порт призначення дозволяє зробити припущення про тип сервісу, до якого відбувається звернення. Наприклад, порт 443 часто використовується для HTTPS, порт 53 - для DNS, порт 22 - для SSH. У системі аналізу ці поля можна використовувати для статистики за портами, фільтрації записів і виявлення активності на ризикових або нетипових портах.

Поле protocol містить назву мережевого протоколу. У спрощеному варіанті воно може приймати значення TCP, UDP, ICMP або інші. Це поле потрібне для побудови розподілу трафіку за протоколами. Наприклад, dashboard може показувати, яку частку становить TCP-трафік, скільки подій пов’язано з UDP і чи присутні ICMP-повідомлення. Така інформація допомагає швидко оцінити загальний характер мережевої активності.

Поле bytes показує обсяг переданих даних у межах певної події або з’єднання. Воно використовується для обчислення загального обсягу трафіку, пошуку записів із найбільшим навантаженням і виявлення потенційно підозрілих передавань великих обсягів даних. Наприклад, якщо певна подія має значення bytes, яке значно перевищує типовий рівень, система може додати до неї бали ризику.

Поле packets містить кількість пакетів, пов’язаних із записом. Воно дозволяє оцінити інтенсивність обміну. Наприклад, невеликий обсяг даних при

великій кількості пакетів може свідчити про значну кількість коротких повідомлень, тоді як велике значення bytes і велике значення packets можуть вказувати на інтенсивне передавання даних. У навчальному прототипі це поле можна використовувати як додаткову ознаку для статистики та оцінювання ризику.

Крім основних полів, система може створювати додаткові службові поля вже під час обробки. До них належать risk_score, risk_level, possible_port_scan і is_suspicious. Поле risk_score містить числову оцінку ризику від 0 до 100. Поле risk_level відображає текстову категорію ризику: низький, середній або високий. Поле possible_port_scan може показувати, чи є IP-адреса джерела пов'язаною з можливою ознакою сканування портів. Поле is_suspicious може використовуватися для швидкої фільтрації підозрілих подій.

Таблиця 3.1 – Основні поля системи

Поле	**Приклад значення**	**Пояснення**
timestamp	2026-05-01 10:05:12	Час мережевої події
src_ip	192.168.1.15	IP-адреса джерела
dst_ip	8.8.8.8	IP-адреса призначення
src_port	52344	Порт джерела
dst_port	53	Порт призначення
protocol	UDP	Протокол
bytes	420	Обсяг переданих даних
packets	3	Кількість пакетів

Така структура є достатньою для реалізації основних функцій системи. Наприклад, для побудови графіка протоколів використовується поле protocol, для визначення найактивніших джерел - src_ip, для пошуку ризикових сервісів - dst_port, для оцінки навантаження - bytes і packets, а для часової аналітики - timestamp.

У процесі попередньої обробки система повинна перевіряти не лише наявність полів, а й коректність їхніх значень. Наприклад, порти мають бути числовими значеннями, обсяг даних не повинен бути від'ємним, IP-адреси мають відповідати очікуваному формату, а часові мітки повинні коректно перетворюватися на дату й час. Така перевірка дозволяє зменшити кількість помилок під час аналізу.

Для демонстраційного набору даних доцільно передбачити як звичайні, так і підозрілі записи. До звичайних можна віднести HTTPS-з'єднання, DNS-запити, невеликі UDP-події та типову взаємодію з вебсервісами. До підозрілих - звернення однієї IP-адреси до багатьох портів, події з великим обсягом переданих даних, використання нетипових портів або велику кількість записів від одного джерела. Завдяки цьому під час тестування можна показати, що система здатна відрізняти звичайні події від потенційно ризикових.

Важливо зазначити, що запропонована структура даних є спрощеною. Вона не містить усіх полів, які можуть бути присутні в реальному PCAP-файлі або професійному журналі мережевого моніторингу. Наприклад, у ній не враховуються TCP-прапорці, доменні імена, повні HTTP-запити, TLS-параметри або детальні дані про сесію. Проте для навчальної системи така структура є достатньою, оскільки вона дозволяє реалізувати статистику, фільтрацію, оцінювання ризику й dashboard-візуалізацію.

У майбутньому структуру даних можна розширити. Наприклад, можна додати поля domain, country, service, duration, status, alert_type або direction. Поле domain дозволило б аналізувати DNS або вебактивність, duration - тривалість з'єднань, direction - напрямок трафіку, а alert_type - тип виявленої підозрілої події. Проте в межах цієї роботи такі поля не є обов'язковими, щоб не ускладнювати реалізацію.

Отже, структура даних для демонстраційної системи повинна бути простою, зрозумілою та достатньою для виконання основних функцій аналізу. Використання CSV-файлу з полями часу, IP-адрес, портів, протоколу, обсягу

даних і кількості пакетів дозволяє реалізувати статистичний аналіз, виявлення підозрілих подій і побудову dashboard-інтерфейсу. Такий підхід добре відповідає практичній меті роботи та забезпечує зрозумілу основу для програмної реалізації.

3.5 Вибір технологій для реалізації програмного прототип

Для реалізації програмного прототипу системи аналізу інтернет-трафіку необхідно обрати технології, які дозволяють швидко створити зрозумілу, наочну та достатньо функціональну систему. Оскільки метою практичної частини є не створення промислового засобу кіберзахисту, а демонстрація принципів аналізу мережевих даних, доцільно використовувати інструменти, які добре підходять для обробки таблиць, побудови графіків і створення dashboard-інтерфейсів.

Основною мовою реалізації обрано Python. Ця мова широко використовується для обробки даних, автоматизації, аналізу журналів, створення прототипів і побудови аналітичних систем. Її перевагою є зрозумілий синтаксис, велика кількість бібліотек і зручність роботи з табличними структурами. Для цієї роботи Python підходить тому, що дозволяє реалізувати завантаження файлів, обробку мережевих записів, статистичні розрахунки, правила оцінювання ризику та побудову інтерфейсу [30].

Для створення dashboard-інтерфейсу доцільно використати Streamlit. Це Python-фреймворк, який дозволяє створювати вебподібні аналітичні застосунки без окремої розробки frontend і backend. У межах цієї роботи це є важливою перевагою, оскільки основна увага має бути зосереджена на логіці аналізу трафіку, а не на складній вебархітектурі. Streamlit дозволяє додавати блок завантаження файлу, картки показників, таблиці, фільтри, графіки та текстові висновки без значного обсягу додаткового коду.

Для обробки даних використовується бібліотека pandas. Вона є одним із найзручніших інструментів Python для роботи з табличними даними. У системі

аналізу трафіку pandas може використовуватися для зчитування CSV-файлів, перевірки колонок, очищення значень, групування записів, підрахунку статистичних показників і сортування результатів. Наприклад, за допомогою pandas можна швидко визначити кількість подій за кожним протоколом, знайти найактивніші IP-адреси або обчислити загальний обсяг переданих даних [31].

Для побудови графіків доцільно використати Plotly. Ця бібліотека дозволяє створювати інтерактивні діаграми, які добре інтегруються зі Streamlit. У межах системи можна побудувати стовпчикову діаграму розподілу протоколів, лінійний графік активності за часом, графік топ IP-адрес або діаграму подій за рівнями ризику. Інтерактивність графіків робить dashboard більш зручним, оскільки користувач може швидше аналізувати результати.

Окремо можна передбачити можливість подальшого використання бібліотек Scapy або PyShark для роботи з PCAP-файлами. Вони дозволяють зчитувати мережеві пакети та витягувати з них потрібні поля. Проте в базовій реалізації практичної частини доцільніше використати CSV-файл. Це робить систему стабільнішою, простішою для тестування та зрозумілішою для опису в роботі. Підтримку PCAP можна зазначити як напрям подальшого розвитку системи.

Вибір CSV як основного формату вхідних даних також має практичне обґрунтування. CSV-файл легко створити вручну, отримати з іншого інструмента або сформувати як тестовий набір. Він має просту табличну структуру, де кожен рядок відповідає окремій події, а кожна колонка - певному параметру трафіку. Для студентської роботи це дозволяє уникнути надмірної складності, але зберегти логіку реального аналізу мережевих записів.

З технічного погляду система може працювати локально на комп’ютері користувача. Для її запуску достатньо встановити Python, необхідні бібліотеки та виконати команду запуску Streamlit-додатка. Після цього в браузері відкривається dashboard, де користувач завантажує файл і переглядає результати.

Такий підхід не потребує серверної інфраструктури, бази даних або складного розгортання.

Важливою перевагою обраного технологічного стеку є його відповідність завданням роботи. Python забезпечує логіку аналізу, pandas - обробку табличних даних, Plotly - візуалізацію, Streamlit - інтерфейс користувача. Усі ці компоненти добре поєднуються між собою та дозволяють створити цілісний програмний прототип. При цьому система залишається зрозумілою для пояснення в тексті роботи та достатньо простою для практичної реалізації.

Запропонований стек також дозволяє дотриматися принципу модульності. Наприклад, окремі функції можна винести в модулі для завантаження, попередньої обробки, статистики, оцінювання ризику та формування звіту. Це робить код більш структурованим і полегшує його опис у практичному розділі. Крім того, у майбутньому до системи можна додати нові функції без повного переписування програми.

Отже, для реалізації програмного прототипу системи аналізу інтернет-трафіку обрано технології Python, Streamlit, pandas і Plotly. Такий вибір є доцільним, оскільки ці інструменти дозволяють швидко створити навчальну аналітичну систему, яка завантажує мережеві дані, виконує їх обробку, розраховує статистичні показники, виявляє підозрілі події та подає результати у вигляді dashboard-інтерфейсу.

3.6 Структура Python-проєкту та основні модулі системи

Після вибору технологій необхідно визначити структуру програмного проєкту. Правильна структура файлів і модулів допомагає зробити систему зрозумілою, зручною для розробки та простою для подальшого розширення. Оскільки розроблювана система аналізу інтернет-трафіку має кілька окремих функцій, доцільно не розміщувати весь код в одному файлі, а поділити його на логічні частини.

Програмний прототип можна організувати як невеликий Python-проєкт. Головним файлом є app.py, який відповідає за запуск dashboard-інтерфейсу на Streamlit. Саме в цьому файлі користувач бачить сторінку системи, завантажує CSV-файл, переглядає картки показників, графіки, таблиці та короткий висновок. Інші функції доцільно винести в окремі модулі, щоб код був більш упорядкованим.

Файл requirements.txt містить перелік бібліотек, необхідних для запуску системи. Наприклад, до нього можна включити streamlit, pandas, plotly та за потреби інші допоміжні бібліотеки. Наявність такого файлу спрощує встановлення залежностей, оскільки користувач може встановити всі потрібні пакети однією командою. Це важливо для практичної частини, бо система має бути відтворюваною й зрозумілою для запуску на іншому комп'ютері.

Файл README.md може містити короткий опис системи, інструкцію із запуску, вимоги до вхідного CSV-файлу та приклад використання. Для студентської роботи цей файл не є основним елементом, але його наявність показує, що проєкт оформлено акуратно. У ньому можна зазначити, що система призначена для навчального аналізу підготовлених мережевих даних і не виконує несанкціоноване перехоплення трафіку.

Папка data містить тестовий набір даних, наприклад sample_traffic.csv. Цей файл використовується для демонстрації роботи системи. У ньому мають бути як звичайні мережеві події, так і кілька підозрілих записів. Це дозволяє під час тестування показати, що система не лише рахує статистику, а й може виділяти події з підвищеним рівнем ризику.

Папка modules містить окремі програмні модулі. Такий поділ потрібний для того, щоб кожна частина системи відповідала за конкретне завдання. Наприклад, один модуль зчитує файл, другий очищує дані, третій рахує статистику, четвертий шукає аномалії, п'ятий оцінює ризик, а шостий формує текстовий звіт. Завдяки цьому код стає більш зрозумілим і легшим для пояснення в роботі.

Модуль loader.py відповідає за завантаження даних. У ньому може бути функція, яка приймає файл від користувача та зчитує його у форматі таблиці pandas DataFrame. Якщо файл має неправильний формат або не може бути прочитаний, система повинна вивести повідомлення про помилку. Цей модуль є першим етапом програмної обробки, тому від нього залежить подальша робота системи.

Модуль preprocessing.py виконує попередню обробку даних. Його завдання - перевірити наявність обов'язкових колонок, привести значення до правильних типів, уніфікувати назви протоколів і підготувати дані для аналізу. Наприклад, поле timestamp потрібно перетворити на формат дати й часу, а поля src_port, dst_port, bytes і packets - на числові значення. Якщо в даних є порожні або некоректні рядки, їх можна видалити або позначити як помилкові.

Модуль statistics.py відповідає за розрахунок основних статистичних показників. У ньому можна реалізувати функції для підрахунку загальної кількості записів, кількості унікальних IP-адрес, загального обсягу трафіку, розподілу протоколів, топ IP-адрес і топ портів. Саме результати цього модуля виводяться на dashboard у вигляді карток, таблиць і графіків.

Модуль anomaly.py призначений для виявлення підозрілих ознак у трафіку. Наприклад, у ньому можна реалізувати функцію виявлення можливого сканування портів. Така функція групує дані за IP-адресою джерела та підраховує кількість унікальних портів призначення. Якщо ця кількість перевищує встановлений поріг, IP-адреса позначається як підозріла. Також у цьому модулі можна реалізувати перевірку надмірної кількості з'єднань або великого обсягу переданих даних.

Модуль risk.py відповідає за розрахунок умовного рівня ризику. У ньому можна створити функцію, яка для кожного запису нараховує бали ризику залежно від певних ознак. Наприклад, використання ризикового порту додає певну кількість балів, великий обсяг даних - додаткові бали, а ознака можливого

сканування портів - ще більше балів. Після цього система визначає рівень ризику: низький, середній або високий.

Модуль report.py формує короткий текстовий висновок. Він може використовувати результати статистичного аналізу та оцінювання ризику. Наприклад, система може автоматично створити повідомлення про те, скільки записів було оброблено, який протокол є найпоширенішим, скільки подій мають високий рівень ризику та яка IP-адреса була найактивнішою. Такий висновок корисний для швидкого розуміння результатів аналізу.

Головний файл app.py об’єднує всі модулі. У ньому створюється інтерфейс Streamlit, реалізується завантаження файлу, викликаються функції попередньої обробки, статистики, виявлення аномалій і розрахунку ризику. Після цього результати виводяться на сторінку у вигляді графіків, таблиць і текстового звіту. Таким чином, app.py виконує роль центрального керуючого компонента.

З погляду логіки виконання програми взаємодія модулів може бути описана так: користувач завантажує файл через app.py, після чого loader.py зчитує дані, preprocessing.py перевіряє та очищує їх, statistics.py рахує основні показники, anomaly.py шукає підозрілі ознаки, risk.py обчислює рівень ризику, а report.py формує короткий висновок. Після цього всі результати повертаються до app.py і відображаються в dashboard.

Такий поділ має кілька переваг. По-перше, система стає простішою для розуміння. По-друге, кожен модуль можна окремо тестувати й змінювати. По-третє, у майбутньому можна додавати нові можливості без порушення всієї структури. Наприклад, якщо потрібно додати підтримку PCAP-файлів, можна розширити loader.py, не змінюючи логіку dashboard-інтерфейсу.

Модульна структура також зручна для опису в студентській роботі. У практичному розділі можна послідовно пояснити, за що відповідає кожен файл, які дані він отримує та який результат повертає. Це робить реалізацію не просто набором коду, а повноцінною програмною системою з визначеною архітектурою.

3.7. Реалізація алгоритмів аналізу інтернет-трафіку

Реалізація алгоритмів аналізу інтернет-трафіку є основною частиною програмного прототипу. Саме на цьому етапі система переходить від завантаження даних до отримання практичних результатів: статистичних показників, рівня ризику, списку підозрілих подій і візуального подання результатів у dashboard-інтерфейсі.

У запропонованій системі алгоритми реалізуються мовою Python. Для роботи з табличними даними використовується бібліотека pandas, а для відображення результатів - Streamlit і Plotly. Вхідним форматом є CSV-файл, у якому кожен рядок описує окрему мережеву подію або з'єднання. Основними полями є час події, IP-адреси джерела та призначення, порти, протокол, кількість пакетів і обсяг переданих даних.

Рисунок 3.3 – Блок-схема реалізації аналізу інтернет-трафіку

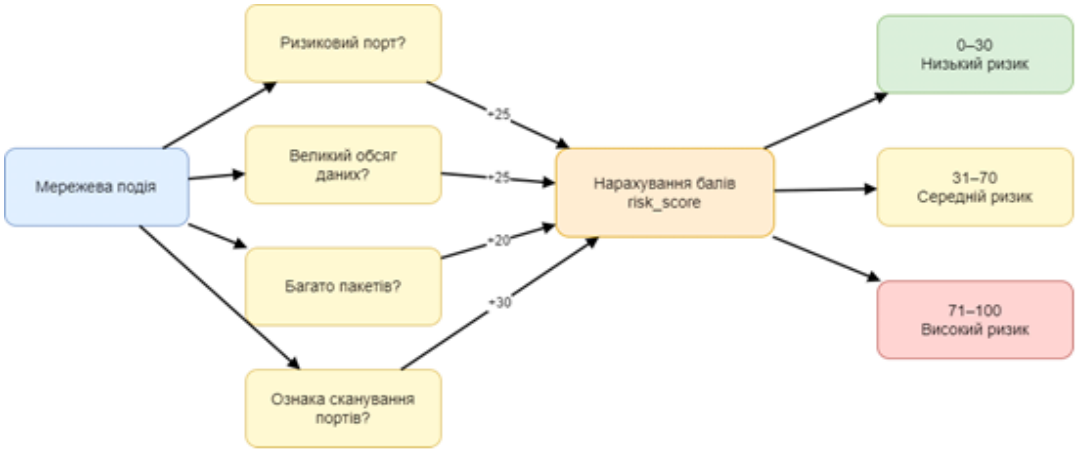


Рисунок 3.4 – Блок-схема оцінювання рівня ризику мережевої події

Першим етапом реалізації є завантаження CSV-файлу. Користувач передає файл через інтерфейс Streamlit, після чого система зчитує його у форматі таблиці. Якщо користувач не завантажив власний файл, може використовуватися тестовий набір даних, розміщений у папці проєкту. Після цього програма перевіряє, чи містить файл усі обов'язкові поля.

Другим етапом є попередня обробка даних. Система приводить часові значення до формату дати й часу, числові поля - до відповідного числового типу, а назви протоколів - до єдиного вигляду. Наприклад, значення tcp, Tcp і TCP уніфікуються як TCP. Некоректні або порожні записи можуть видалятися або ігноруватися.

Третім етапом є статистичний аналіз. Система визначає загальну кількість записів, кількість унікальних IP-адрес, загальний обсяг переданих даних, найпоширеніший протокол, топ активних IP-адрес і топ портів. Також виконується групування подій за часом для побудови графіка активності.

Четвертим етапом є виявлення підозрілих ознак. У навчальному прототипі використано прості правила. Наприклад, якщо одна IP-адреса звертається до великої кількості різних портів, система може позначити таку активність як можливе сканування портів. Якщо подія має надто великий обсяг переданих даних або використовує ризиковий порт, вона також отримує підвищений рівень уваги.

П'ятим етапом є розрахунок умовного рівня ризику. Для кожної події система формує показник risk_score у межах від 0 до 100 балів. Бали додаються за наявність певних ознак: використання ризикового порту, великий обсяг трафіку, велика кількість пакетів або можлива ознака сканування портів. Після цього система визначає рівень ризику: низький, середній або високий.

Наприклад, події з оцінкою від 0 до 30 балів можна віднести до низького рівня ризику, від 31 до 70 - до середнього, а від 71 до 100 - до високого. Такий підхід є спрощеним, але він добре підходить для демонстраційної системи,

оскільки дозволяє швидко відокремити звичайні записи від потенційно підозрілих.

Після виконання аналізу система додає до таблиці нові поля: risk_score, risk_level, possible_port_scan та is_suspicious. Завдяки цьому користувач може не лише переглядати початкові мережеві записи, а й бачити результат їх автоматичної обробки. У dashboard-інтерфейсі ці дані використовуються для побудови таблиць, графіків і фільтрів.

Окремо реалізується формування короткого текстового висновку. Система може автоматично повідомляти, скільки записів було оброблено, який протокол є найпоширенішим, скільки подій мають середній або високий рівень ризику та яка IP-адреса є найактивнішою. Це робить результати аналізу більш зрозумілими для користувача.

3.8. Тестування системи та аналіз отриманих результатів

Після реалізації програмного прототипу необхідно провести його тестування. Метою тестування є перевірка того, чи правильно система завантажує вхідний файл, обробляє мережеві записи, розраховує статистичні показники, визначає підозрілі події та відображає результати в dashboard-інтерфейсі.

Для тестування використовується підготовлений CSV-файл із контрольним набором мережевих даних. Такий підхід є зручним для навчальної роботи, оскільки дозволяє заздалегідь додати до набору як звичайні, так і підозрілі записи. Крім того, використання тестового файлу не потребує перехоплення реального трафіку та не створює ризиків, пов’язаних із конфіденційністю даних.

Контрольний набір даних містить різні типи мережевих подій. До нього включено звичайні HTTPS-з’єднання, DNS-запити, UDP-події, ICMP-повідомлення, звернення до поширених портів, а також спеціально створені

записи з підозрілими ознаками. Наприклад, у наборі є IP-адреса, яка звертається до великої кількості різних портів. Така поведінка використовується для перевірки алгоритму виявлення можливого сканування портів.

Перші два записи можна вважати типовими. Перший запис описує TCP-з'єднання з портом 443, що характерно для HTTPS-трафіку. Другий запис описує UDP-запит до порту 53, що відповідає DNS-трафіку. Наступні записи пов'язані з IP-адресою 192.168.1.25, яка звертається до різних портів. Якщо кількість таких звернень перевищує встановлений поріг, система повинна позначити цю активність як потенційно підозрілу.

На першому етапі тестування перевіряється завантаження файлу. Користувач відкриває dashboard-систему та завантажує CSV-файл через відповідний елемент інтерфейсу. Якщо файл має правильну структуру, система переходить до аналізу. Якщо ж файл не містить обов'язкових колонок, програма виводить повідомлення про помилку. Це дозволяє перевірити коректність механізму валідації вхідних даних.

На другому етапі перевіряється попередня обробка. Система має правильно розпізнати часові мітки, числові значення портів, кількість байтів і пакетів. Також вона повинна привести назви протоколів до єдиного формату. Наприклад, значення tcp, Tcp і TCP після обробки мають бути представлені як TCP. Це потрібно для правильного групування записів і побудови статистики.

На третьому етапі перевіряється розрахунок статистичних показників. Система повинна показати загальну кількість записів, кількість унікальних IP-адрес, загальний обсяг переданих даних, найпоширеніший протокол і кількість подій із підвищеним рівнем ризику. Ці показники виводяться у верхній частині dashboard-інтерфейсу у вигляді інформаційних карток.

Далі перевіряється побудова графіків. Графік розподілу протоколів має показувати співвідношення TCP, UDP, ICMP та інших протоколів у тестовому наборі. Графік активності за часом дозволяє побачити, у які моменти було зафіксовано найбільшу кількість подій. Також система може показувати топ

активних IP-адрес і топ портів призначення. Такі графіки роблять результати аналізу більш наочними.

Окремо тестується алгоритм виявлення можливого сканування портів. Для цього використовується IP-адреса, яка звертається до багатьох різних портів. Система групує записи за IP-адресою джерела та підраховує кількість унікальних портів призначення. Якщо ця кількість перевищує встановлений поріг, записи позначаються як підозрілі. Це дозволяє перевірити, чи правильно працює правило possible_port_scan.

Після цього перевіряється розрахунок risk_score. Звичайні події, наприклад HTTPS-з'єднання на порт 443 із невеликим обсягом даних, повинні отримувати низький рівень ризику. Події з ризиковими портами, великим обсягом переданих даних або ознаками сканування портів повинні отримувати середній або високий рівень ризику. Такий результат показує, що система здатна відрізняти типові записи від потенційно підозрілих.

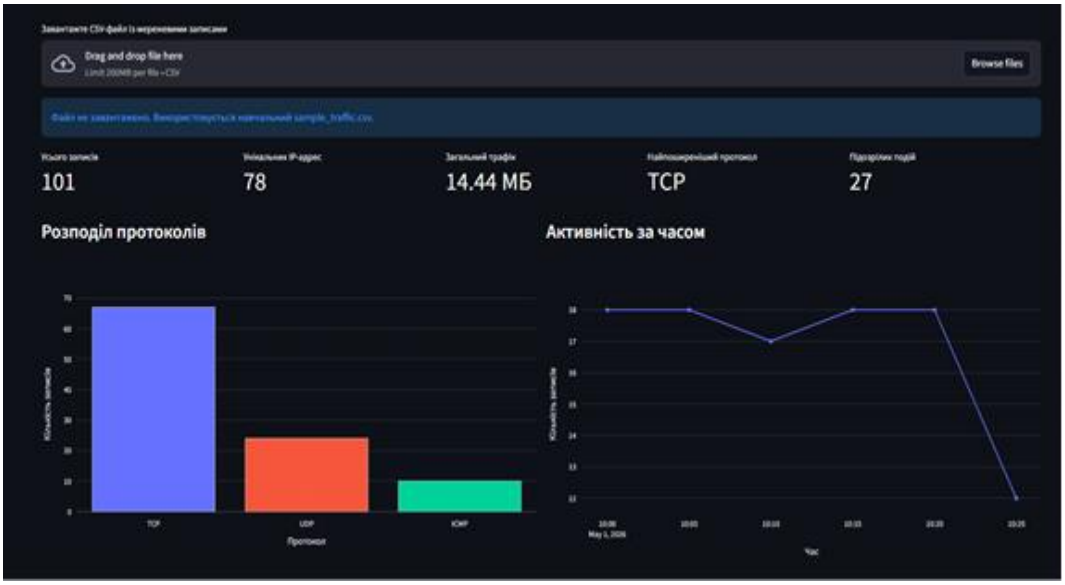


Рисунок 3.5 – Тестування дашборду

Важливою частиною тестування є перевірка фільтрів. Користувач повинен мати змогу відображати записи за протоколом, IP-адресою, портом або рівнем ризику. Наприклад, можна залишити в таблиці лише події з високим рівнем

ризику або тільки записи, пов’язані з певною IP-адресою. Якщо після застосування фільтрів таблиця оновлюється правильно, можна вважати, що механізм фільтрації працює коректно.

За результатами тестування система формує короткий автоматичний висновок. Наприклад:

За результатами аналізу було оброблено 500 записів мережевого трафіку. Найпоширенішим протоколом є TCP. Виявлено 34 події із середнім рівнем ризику та 6 подій із високим рівнем ризику. Найактивнішою IP-адресою джерела є 192.168.1.25.

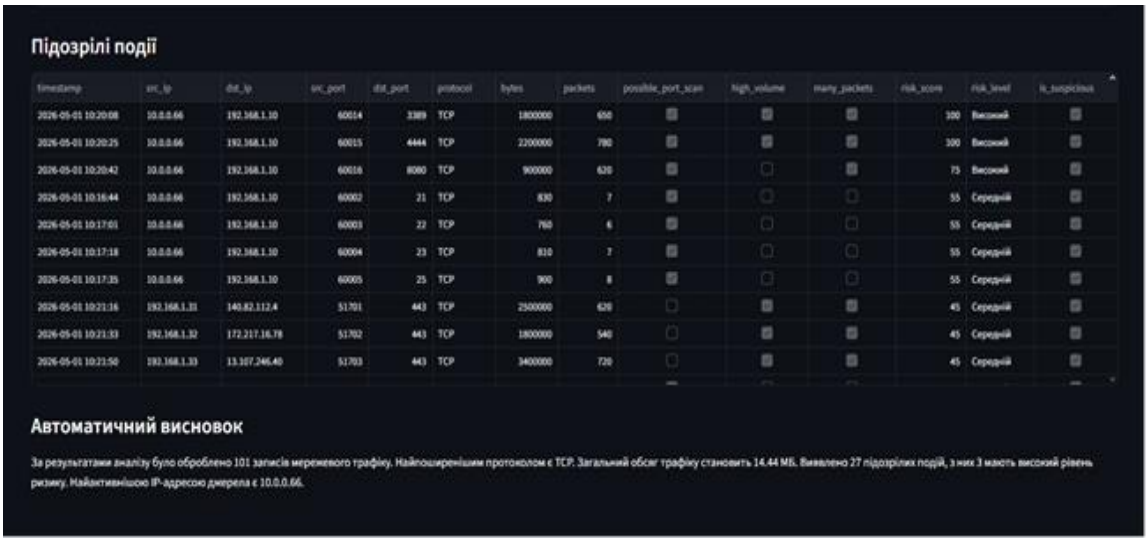


Рисунок 3.6 – Тестування висновків системи

У результаті тестування встановлено, що програмний прототип виконує основні поставлені завдання. Система завантажує CSV-файл, перевіряє його структуру, виконує попередню обробку, розраховує статистичні показники, будує графіки, визначає підозрілі події та присвоює їм умовний рівень ризику. Dashboard-інтерфейс дозволяє переглядати як загальну статистику, так і окремі мережеві записи.

Разом із цим потрібно враховувати обмеження розробленого прототипу. Система не виконує аналіз реального трафіку в режимі реального часу, не здійснює глибокий аналіз пакетів і не замінює професійні системи моніторингу

або виявлення вторгнень. Вона працює з підготовленими CSV-файлами та використовує спрощені правила оцінювання ризику. Проте для студентської роботи такий підхід є достатнім, оскільки він демонструє основні принципи автоматизованого аналізу інтернет-трафіку.

3.8 Висновок до 3 розділу

У третьому розділі було виконано проєктування, реалізацію та тестування програмної системи аналізу інтернет-трафіку. На початку розділу було сформульовано функціональні та нефункціональні вимоги до системи. Визначено, що система повинна завантажувати CSV-файл із мережевими даними, перевіряти його структуру, виконувати попередню обробку, розраховувати статистичні показники, виявляти підозрілі події, оцінювати рівень ризику та відображати результати у dashboard-інтерфейсі.

Було запропоновано архітектуру системи, яка складається з окремих модулів: завантаження даних, перевірки структури, попередньої обробки, статистичного аналізу, виявлення аномалій, оцінювання ризику, візуалізації та формування короткого звіту. Такий модульний підхід дозволяє зробити програму зрозумілою, зручною для тестування та придатною для подальшого розширення.

У розділі було описано алгоритм аналізу інтернет-трафіку. Він включає завантаження CSV-файлу, перевірку обов’язкових полів, очищення даних, групування записів за протоколами, IP-адресами, портами й часом, виявлення підозрілих ознак, розрахунок показника risk_score і визначення рівня ризику. Такий алгоритм забезпечує послідовну обробку мережевих даних і дозволяє отримати зрозумілий результат для користувача.

Також було визначено структуру вхідних даних для демонстраційної системи. Основними полями є timestamp, src_ip, dst_ip, src_port, dst_port, protocol, bytes і packets. Цих параметрів достатньо для реалізації статистичного аналізу,

фільтрації, виявлення підозрілих подій і побудови графіків у dashboard-інтерфейсі.

Для реалізації програмного прототипу було обрано Python, Streamlit, pandas і Plotly. Python забезпечує основну логіку аналізу, pandas використовується для обробки табличних даних, Plotly - для побудови графіків, а Streamlit - для створення dashboard-інтерфейсу. Такий технологічний стек дозволяє створити локальну систему без складної серверної інфраструктури.

ВИСНОВКИ

У процесі виконання роботи було досліджено теоретичні та практичні аспекти створення комп'ютерної системи аналізу інтернет-трафіку. Актуальність теми зумовлена тим, що сучасні комп'ютерні мережі щоденно обробляють значні обсяги даних, а контроль мережевої активності є важливим для стабільної роботи сервісів, виявлення перевантажень і пошуку потенційно підозрілих подій.

У першому розділі було розглянуто теоретичні основи аналізу інтернет-трафіку. Визначено, що інтернет-трафік є сукупністю даних, які передаються між пристроями в мережі, а для його аналізу важливими є IP-адреси, порти, протоколи, часові мітки, кількість пакетів і обсяг переданих даних. Також було розглянуто основні мережеві протоколи, класифікацію трафіку та типові аномалії, які можуть свідчити про підозрілу активність.

У другому розділі було досліджено методи та засоби аналізу інтернет-трафіку. Розглянуто способи збору мережевих даних, статистичні методи аналізу, сигнатурний та евристичний підходи до виявлення загроз. Окрему увагу приділено інструментам Wireshark, Zeek, Suricata та Snort, які використовуються для аналізу пакетів, моніторингу мережевих подій і виявлення вторгнень.

У третьому розділі було виконано проєктування, реалізацію та тестування програмної системи аналізу інтернет-трафіку. Було сформульовано функціональні та нефункціональні вимоги до системи, описано її архітектуру, алгоритм роботи, структуру вхідних даних і технології реалізації. Для створення прототипу обрано Python, Streamlit, pandas і Plotly, що дозволило реалізувати локальну dashboard-систему без складної серверної інфраструктури.

У межах практичної частини було запропоновано структуру Python-проєкту, який складається з окремих модулів для завантаження даних, попередньої обробки, статистичного аналізу, виявлення аномалій, оцінювання

ризику та формування короткого звіту. Такий підхід забезпечує зрозумілу організацію програми та можливість її подальшого розширення.

Розроблений програмний прототип працює з підготовленими мережевими даними у форматі CSV. Система завантажує файл, перевіряє його структуру, виконує попередню обробку, розраховує статистичні показники, визначає найактивніші IP-адреси, аналізує протоколи й порти, виявляє потенційно підозрілі події та присвоює їм умовний рівень ризику.

Для оцінювання подій використано спрощений алгоритм risk_score, який враховує ризикові порти, великий обсяг переданих даних, значну кількість пакетів і можливу ознаку сканування портів. На основі отриманого показника події поділяються на три рівні ризику: низький, середній і високий. Такий підхід не замінює професійні системи кіберзахисту, але добре демонструє базові принципи автоматизованого аналізу трафіку.

Тестування системи було проведено на контрольному наборі мережевих даних. У результаті перевірено завантаження CSV-файлу, коректність обробки полів, побудову статистики, роботу фільтрів, виявлення підозрілих подій і відображення результатів у dashboard-інтерфейсі. Тестування підтвердило працездатність системи та відповідність основних функцій та вимогам.

Мету роботи досягнуто. Було досліджено методи аналізу інтернет-трафіку та розроблено програмний прототип системи, яка забезпечує обробку мережевих даних, виявлення підозрілих подій і візуалізацію результатів. Поставлені завдання виконано в повному обсязі.

Розроблена система має навчальний характер і певні обмеження. Вона не виконує аналіз реального трафіку в режимі реального часу, не здійснює глибокий пакетний аналіз і не замінює професійні IDS/IPS-рішення. Проте для студентської роботи такий прототип є достатнім, оскільки він демонструє основні етапи аналізу інтернет-трафіку: завантаження даних, обробку, статистику, оцінювання ризику та візуалізацію.

Подальший розвиток системи може передбачати підтримку PCAP-файлів, інтеграцію з журналами Zeek або Suricata, розширення правил оцінювання ризику, експорт результатів у PDF або HTML, збереження історії аналізу та використання методів машинного навчання для виявлення складніших аномалій. Це дозволило б наблизити навчальний прототип до реальних систем моніторингу й аналізу мережевого трафіку.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1.National Institute of Standards and Technology. The NIST Cybersecurity Framework (CSF) 2.0. Gaithersburg : NIST, 2024. URL: https://nvlpubs.nist.gov/nistpubs/CSWP/NIST.CSWP.29.pdf. (дата звернення: 27.05.2026).

2.Joint Task Force. Security and Privacy Controls for Information Systems and Organizations : NIST Special Publication 800-53, Revision 5. Gaithersburg : National Institute of Standards and Technology, 2020. URL: https://csrc.nist.gov/pubs/sp/800/53/r5/upd1/final. (дата звернення: 27.05.2026).

3.Rose S., Borchert O., Mitchell S., Connelly S. Zero Trust Architecture : NIST Special Publication 800-207. Gaithersburg : National Institute of Standards and Technology, 2020. URL: https://csrc.nist.gov/pubs/sp/800/207/final. (дата звернення: 27.05.2026).

4.Cisco. Cisco Annual Internet Report (2018–2023) White Paper. San Jose : Cisco Systems, 2020. URL: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html. (дата звернення: 27.05.2026).

5.European Union Agency for Cybersecurity. ENISA Threat Landscape 2024. Athens : ENISA, 2024. URL: https://www.enisa.europa.eu/publications/enisa-threat-landscape-2024. (дата звернення: 27.05.2026).

6.Cybersecurity and Infrastructure Security Agency. Cybersecurity Performance Goals 2.0. Washington : CISA, 2025. URL: https://www.cisa.gov/cybersecurity-performance-goals-2-0-cpg-2-0. Дата звернення: 27.05.2026.

7.OWASP Foundation. OWASP Top 10:2021. OWASP Foundation, 2021. URL: https://owasp.org/Top10/2021/. (дата звернення: 27.05.2026).

8.Internet Assigned Numbers Authority. Service Name and Transport Protocol Port Number Registry. IANA, 2026. URL: https://www.iana.org/assignments/service-names-port-numbers. (дата звернення: 27.05.2026).

9.Internet Assigned Numbers Authority. Assigned Internet Protocol Numbers. IANA, 2026. URL: https://www.iana.org/assignments/protocol-numbers. (дата звернення: 27.05.2026).

10.Internet Assigned Numbers Authority. Hypertext Transfer Protocol (HTTP) Field Name Registry. IANA, 2026. URL: https://www.iana.org/assignments/http-fields. (дата звернення: 27.05.2026).

11.Eddy W. RFC 9293: Transmission Control Protocol (TCP). Fremont : Internet Engineering Task Force, 2022. URL: https://datatracker.ietf.org/doc/rfc9293/. (дата звернення: 27.05.2026).

12.Fielding R., Nottingham M., Reschke J. RFC 9110: HTTP Semantics. Fremont : Internet Engineering Task Force, 2022. URL: https://datatracker.ietf.org/doc/rfc9110/. (дата звернення: 27.05.2026).

13.Fielding R., Nottingham M., Reschke J. RFC 9112: HTTP/1.1. Fremont : Internet Engineering Task Force, 2022. URL: https://datatracker.ietf.org/doc/html/rfc9112. (дата звернення: 27.05.2026).

14.Bishop M. RFC 9114: HTTP/3. Fremont : Internet Engineering Task Force, 2022. URL: https://www.rfc-editor.org/info/rfc9114. (дата звернення: 27.05.2026).

15.Iyengar J., Thomson M. RFC 9000: QUIC: A UDP-Based Multiplexed and Secure Transport. Fremont : Internet Engineering Task Force, 2021. URL: https://datatracker.ietf.org/doc/rfc9000/. (дата звернення: 27.05.2026).

16.Thomson M., Turner S. RFC 9001: Using TLS to Secure QUIC. Fremont : Internet Engineering Task Force, 2021. URL: https://datatracker.ietf.org/doc/html/rfc9001. (дата звернення: 27.05.2026).

17.Sheffer Y., Saint-Andre P., Fossati T. RFC 9325: Recommendations for Secure Use of Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS). Fremont : Internet Engineering Task Force, 2022. URL: https://datatracker.ietf.org/doc/rfc9325/. (дата звернення: 27.05.2026).

18.Huitema C., Mankin A., Dickinson S. RFC 9250: DNS over Dedicated QUIC Connections. Fremont : Internet Engineering Task Force, 2022. URL: https://datatracker.ietf.org/doc/rfc9250/. (дата звернення: 27.05.2026).

19.Wireshark Foundation. Wireshark User's Guide. Official Documentation. 2026. URL: https://www.wireshark.org/docs/wsug_html_chunked/. (дата звернення: 27.05.2026).

20.Wireshark Foundation. Wireshark Display Filter Reference. Official Documentation. 2026. URL: https://www.wireshark.org/docs/dfref/. (дата звернення: 27.05.2026).

21.Wireshark Foundation. TShark Manual Page. Official Documentation. 2026. URL: https://www.wireshark.org/docs/man-pages/tshark.html. (дата звернення: 27.05.2026).

22.Zeek Project. Zeek Documentation : Book of Zeek. Official Documentation. 2026. URL: https://docs.zeek.org/. (дата звернення: 27.05.2026).

23.Zeek Project. The Zeek Network Security Monitor. Official Website. 2026. URL: https://zeek.org/. (дата звернення: 27.05.2026).

24.Open Information Security Foundation. Suricata User Guide. Official Documentation. 2026. URL: https://docs.suricata.io/. (дата звернення: 27.05.2026).

25.Open Information Security Foundation. Suricata Documentation. Official Documentation. 2026. URL: https://suricata.io/documentation/. (дата звернення: 27.05.2026).

26.Cisco Talos. Snort 3 User Manual. San Jose : Cisco, 2024. URL: https://snort.org/downloads/snortplus/snort_manual.pdf. (дата звернення: 27.05.2026).

27.Cisco Talos. Getting Started with Snort 3. Official Documentation. 2026. URL: https://docs.snort.org/start/. (дата звернення: 27.05.2026).

28.The Python Software Foundation. Python 3.14.5 Documentation. Official Documentation. 2026. URL: https://docs.python.org/. (дата звернення: 27.05.2026).

29.The Python Software Foundation. Python Documentation by Version. Official Website. 2026. URL: https://www.python.org/doc/versions/. (дата звернення: 27.05.2026).

30.pandas Development Team. pandas Documentation. Official Documentation. 2026. URL: https://pandas.pydata.org/docs/. (дата звернення: 27.05.2026).

31.pandas Development Team. pandas User Guide. Official Documentation. 2026. URL: https://pandas.pydata.org/docs/user_guide/index.html. (дата звернення: 27.05.2026).

32.Streamlit Inc. Streamlit Documentation. Official Documentation. 2026. URL: https://docs.streamlit.io/. (дата звернення: 27.05.2026).

33.Streamlit Inc. Streamlit API Reference. Official Documentation. 2026. URL: https://docs.streamlit.io/develop/api-reference. (дата звернення: 27.05.2026).

34.Plotly Technologies Inc. Plotly Express in Python. Official Documentation. 2026. URL: https://plotly.com/python/plotly-express/. (дата звернення: 27.05.2026).

35.Scapy Community. Scapy Documentation. Official Documentation. 2026. URL: https://scapy.readthedocs.io/. (дата звернення: 27.05.2026).

36.PyShark Project. PyShark Documentation. Official Documentation. 2026. URL: https://kiminewt.github.io/pyshark/. (дата звернення: 27.05.2026).

37.Neto E. C. P., Dadkhah S., Ferreira R., Zohourian A., Lu R., Ghorbani A. A. CICIoT2023: A Real-Time Dataset and Benchmark for Large-Scale Attacks in IoT Environment. *Sensors*. 2023. Vol. 23, No. 13. Article 5941. URL: https://www.mdpi.com/1424-8220/23/13/5941. (дата звернення: 27.05.2026).

38.Ferrag M. A., Friha O., Hamouda D., Maglaras L., Janicke H. Edge-IIoTset: A New Comprehensive Realistic Cyber Security Dataset of IoT and IIoT Applications. *IEEE Access*. 2022. URL: https://ieee-dataport.org/documents/edge-iiotset-new-comprehensive-realistic-cyber-security-dataset-iot-and-iiot-applications. (дата звернення: 27.05.2026).

39.Sarhan M., Layeghy S., Moustafa N., Portmann M. NetFlow Datasets for Machine Learning-Based Network Intrusion Detection Systems. Big Data

Technologies and Applications. Cham : Springer, 2021. URL: https://link.springer.com/chapter/10.1007/978-3-030-72802-1_9. (дата звернення: 27.05.2026).

40.Sarhan M., Layeghy S., Moustafa N., Portmann M. Evaluating Standard Feature Sets Towards Increased Generalisability and Explainability of ML-Based Network Intrusion Detection. *Big Data Research*. 2022. Vol. 30. Article 100359. URL: https://www.sciencedirect.com/science/article/abs/pii/S2214579622000533. (дата звернення: 27.05.2026).

41.National Institute of Standards and Technology. Cybersecurity Log Management Planning Guide : Draft NIST Special Publication 800-92r1. Gaithersburg : NIST, 2023. URL: https://csrc.nist.gov/pubs/sp/800/92/r1/ipd. (дата звернення: 27.05.2026).

42.Elastic. Elastic Common Schema (ECS) Reference. Official Documentation. 2026. URL: https://www.elastic.co/docs/reference/ecs. (дата звернення: 27.05.2026).

43.MITRE. MITRE ATT&CK Enterprise Matrix. Official Knowledge Base. 2026. URL: https://attack.mitre.org/. (дата звернення: 27.05.2026).

44.SigmaHQ. Sigma Rules Specification. Official Documentation. 2026. URL: https://github.com/SigmaHQ/sigma-specification. (дата звернення: 27.05.2026).

45.SigmaHQ. Sigma Rules. Sigma Detection Format Documentation. 2026. URL: https://sigmahq.io/docs/basics/rules.html. (дата звернення: 27.05.2026).

46.Wazuh. Network IDS Integration: Suricata. Official Documentation. 2026. URL: https://documentation.wazuh.com/current/proof-of-concept-guide/integrate-network-ids-suricata.html. (дата звернення: 27.05.2026).

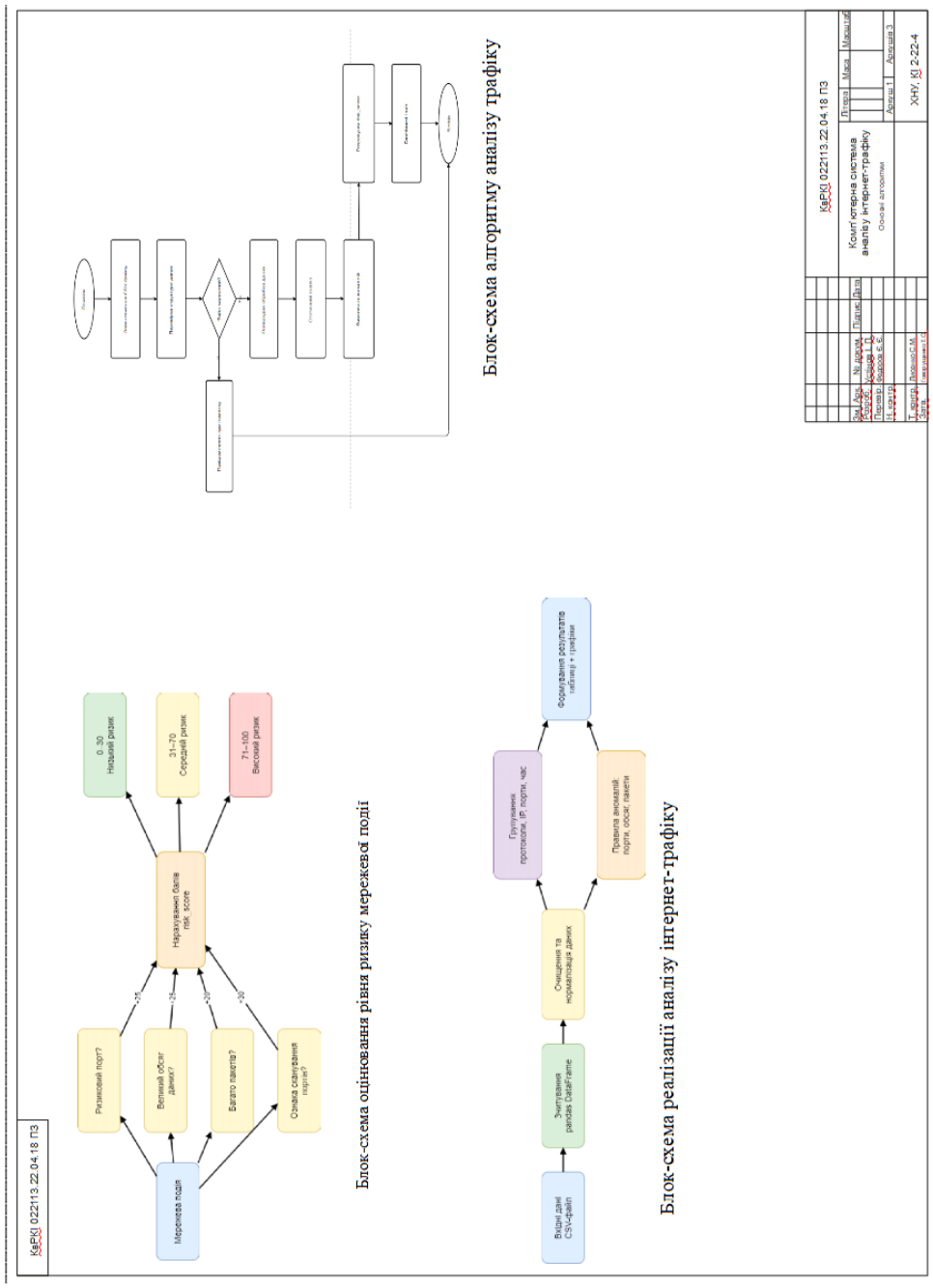
47.Wazuh. Network Security Monitoring with Wazuh and Zeek. Wazuh Blog. 2025. URL: https://wazuh.com/blog/network-security-monitoring-with-wazuh-and-zeek/. (дата звернення: 27.05.2026).

48. Grafana Labs. Logs Visualization. Grafana Documentation. 2026. URL: https://grafana.com/docs/grafana/latest/visualizations/panels-visualizations/visualizations/logs/. (дата звернення: 27.05.2026).

ДОДАТОК А

(обов’язковий)

«Алгоритм подій проведення аналізу даних інтернет трафіку»



ДОДАТОК Б

(обов’язковий)

« Класифікація інтернет-трафіку за основними ознаками »

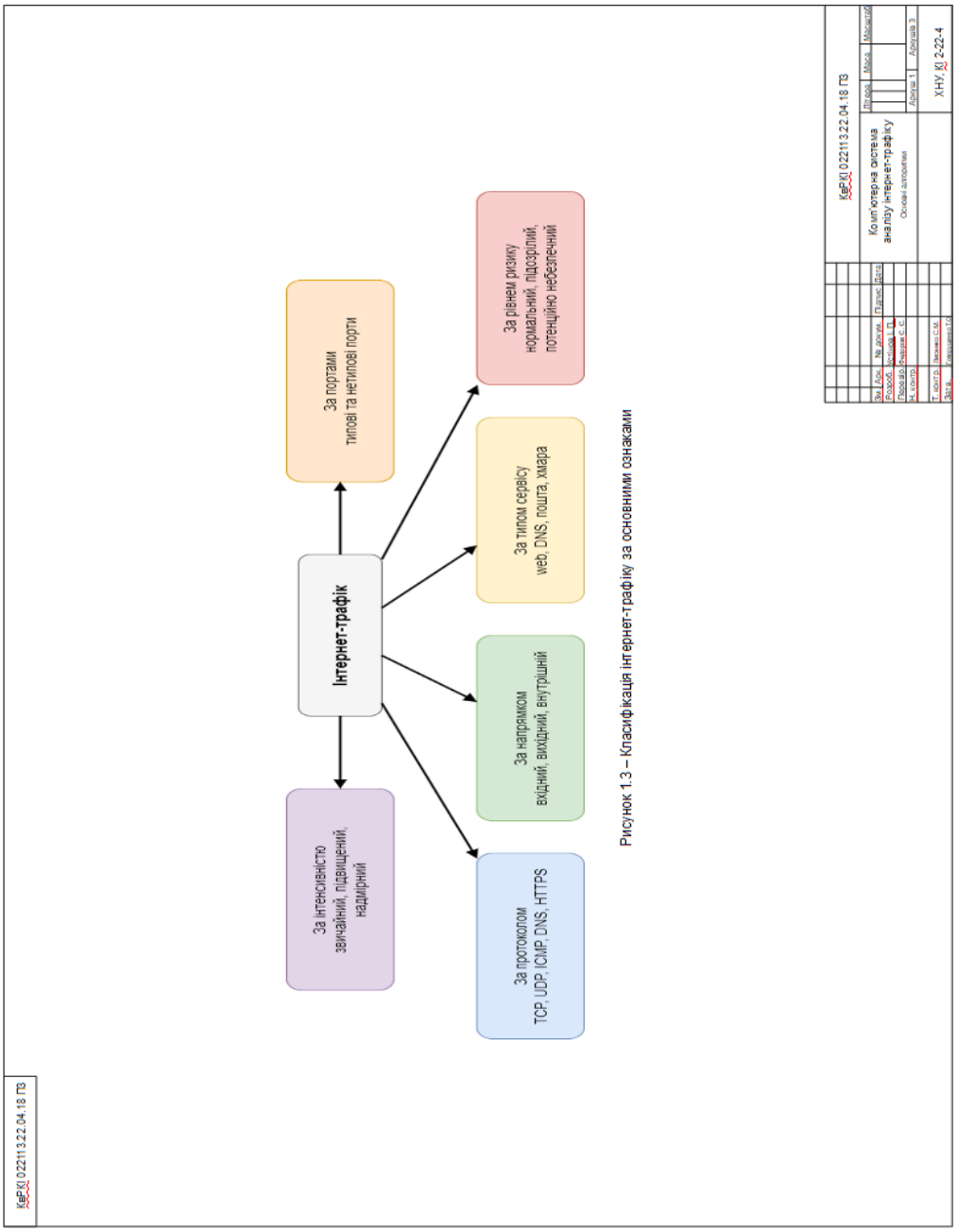
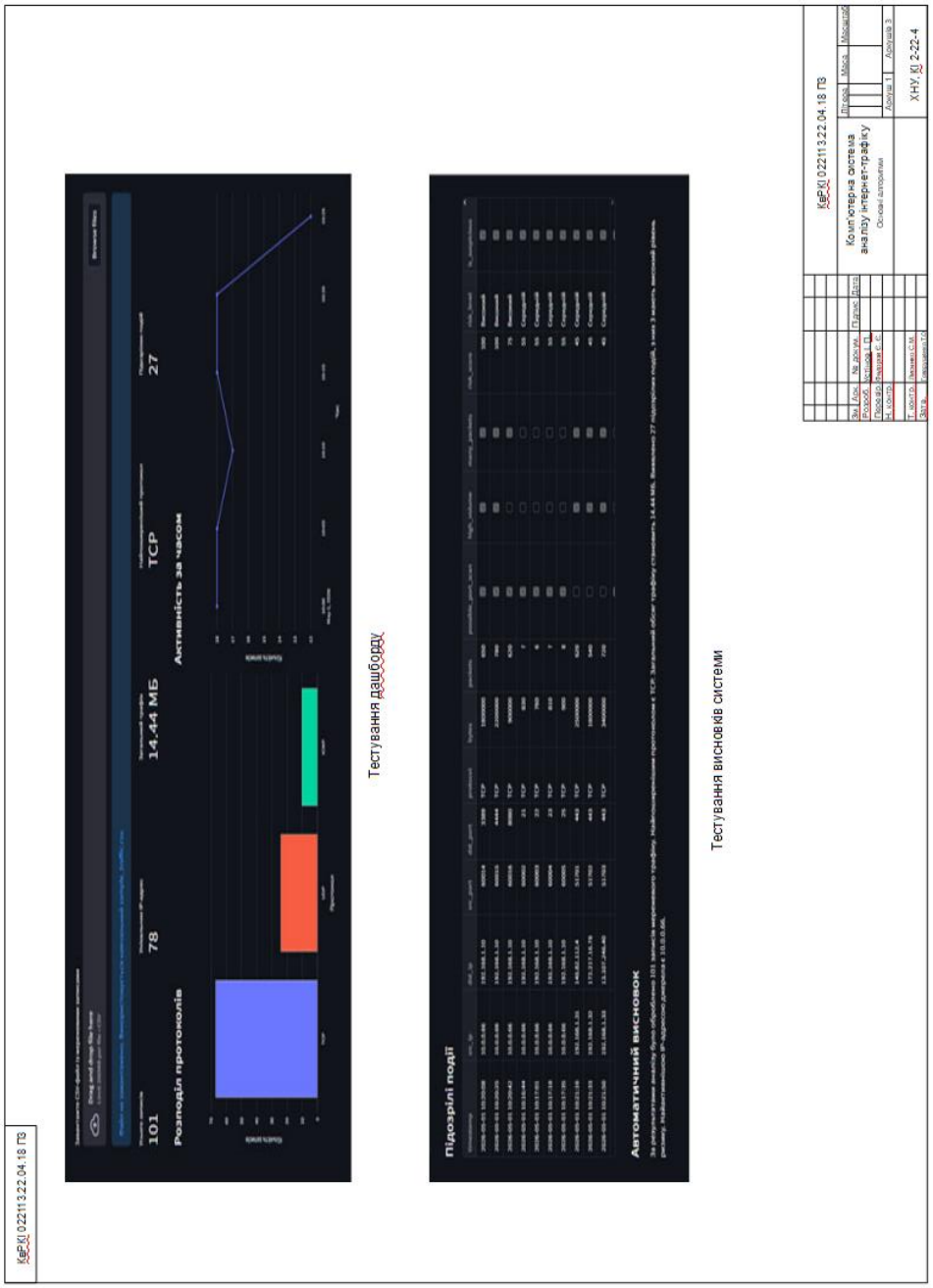


Рисунок 1.3 – Класифікація інтернет-трафіку за основними ознаками

ДОДАТОК В

(обов’язковий)

«Тестування висновків системи»



РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ
ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Назва кваліфікаційної роботи Комп'ютерна система аналізу інтернет-трафік
Автор Ілля УСТІНОВ
Освітня програма Комп'ютерна інженерія та програмування
Рівень вищої освіти перший (бакалаврський)
Спеціальність 123 Комп'ютерна інженерія
Науковий керівник: д-т технічних наук, професор Євген ФЕДОРОВ

На основі аналізу кваліфікаційної роботи на дотримання вимог академічної доброчесності (у т.ч. відсутності ознак академічного плагіату) з урахуванням результатів перевірки роботи спеціалізованим програмним засобом(ами) комісія зробила такий висновок:

№	Висновок	Позначка про відповідність
1	Ознаки академічного плагіату	
1.1	Запозичення, виявлені в роботі, є законними і не є академічним плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних, якщо потрібно). Робота приймається до захисту.	відповідає
1.2	Виявлені запозичення не є академічним плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована.	
1.3	Виявлені запозичення не є академічним плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. Робота може бути допущена до захисту після того як буде відкоригована та доопрацьована і успішно пройде повторну перевірку на академічний плагіат.	
1.4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття текстових запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	
2	Інші види порушень академічної доброчесності	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) усі запозичення фрагментарні, або мають належним чином оформленні посилання;

2) окремі виявлені збіги є загальновживаними фразами або виразами, про що свідчить посилання системи на збіг з джерелами на один фрагмент речення;

3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

4) значна частина знайденого плагіату відноситься до списку використаних джерел

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості StrikePlagiarism, складає 2.3%; та системою Anti-Plagiarism складає 0.6%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

01.06.2026

Завідувач кафедри — Підпис — Ольга ПАВЛОВА (Ім'я, ПРІЗВИЩЕ)

Гарант освітньої програми — Підпис — Андрій НІЧЕПОРУК (Ім'я, ПРІЗВИЩЕ)

Керівник кваліфікаційної роботи — Підпис — Євген ФЕДОРОВ (Ім'я, ПРІЗВИЩЕ)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Дипломник: Устінов Ілля Петрович

Тема: Комп'ютерна система аналізу інтернет-трафіку

Спеціальність: 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи:

Кількість листів креслень ___3___ Кількість сторінок записки ___76___

1. Короткий зміст роботи та прийнятих рішень: Метою роботи є проєктування, реалізація та тестування програмного прототипу комп'ютерної системи аналізу інтернет-трафіку. Система забезпечує завантаження підготовлених мережевих даних, їх попередню обробку, розрахунок статистичних показників, виявлення підозрілих ознак, оцінювання рівня ризику подій і візуалізацію результатів у dashboard-інтерфейсі.

2. Висновок про відповідність роботи дипломному завданню: Робота повністю відповідає поставленому завданню.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи:

В першому розділі кваліфікаційної роботи проведено дослідження теоретичних основ аналізу інтернет-трафіку: розглянуто поняття трафіку, основні мережеві протоколи (TCP/IP, UDP, ICMP, DNS, HTTP/HTTPS), класифікацію трафіку, а також типові аномалії та загрози, які можна виявити за допомогою аналізу (сканування портів, DoS-подібна активність, високий обсяг даних тощо). На основі проведеного аналізу сформульовано постановку задачі.

В другому розділі кваліфікаційної роботи досліджено методи та засоби аналізу інтернет-трафіку: методи збору даних, статистичні методи, сигнатурний та евристичний підходи до виявлення загроз, а також сучасні інструменти (Wireshark, Zeek, Suricata, Snort). Виконано порівняльний аналіз та обґрунтовано вибір технологій для реалізації прототипу.

В третьому розділі кваліфікаційної роботи виконано проєктування, реалізацію та тестування програмної системи. Розроблено архітектуру системи, алгоритми аналізу, структуру даних, модульну структуру Python-проєкту. Реалізовано завантаження та попередню обробку даних, статистичний аналіз, виявлення аномалій, розрахунок risk_score, а також інтерактивний dashboard на базі Streamlit та Plotly. Проведено тестування на контрольному наборі даних з оцінкою результатів.

4. Позитивні сторони роботи: висока практична цінність роботи.

5. Негативні Недостатньо уваги приділено тестуванню на великих обсягах даних та інтеграції з реальними інструментами захоплення трафіку

6. Оцінка графічного оформлення та пояснювальної записки роботи: Пояснювальна записка оформлена коректно, згідно діючих стандартів оформлення документації.

7. Відгук про роботу в цілому: Робота виконана на належному науково-технічному рівні.

8. Інші зауваження: _____

9. Оцінка дипломної роботи: задовільно

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) _____

Онищко О.Г., доцент каф. ІПЗ, ХНУ

_____

"__"_____2026 р. _____(підпис)

Зав. кафедри КІІС
д-р. філософії Ользі ПАВЛОВІЙ
Ілля УСТІНОВ

ПІБ здобувача вищої освіти

ФІТ, 4 курсу, групи КІ2-22-4

ЗАЯВА

З правилами чинного Положення про систему забезпечення академічної доброчесності у Хмельницькому національному університеті, згідно з яким виявлення академічного плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту і застосування заходів академічної відповідальності, ознайомлений (а). Про використання спеціалізованих програмних засобів (СПЗ) StrikePlagiarism та Anti-Plagiarism для перевірки кваліфікаційних робіт здобувачів вищої освіти на наявність академічного плагіату оповіщений (а). Надаю університету право на передачу моєї роботи для обробки та збереження в базах даних СПЗ і використання роботи для виявлення академічного плагіату в інших роботах, які перевіряються СПЗ.

Також надаю свою згоду на обробку й збереження університетом моєї роботи в Інституційному репозитарії Хмельницького національного університету.

Робота надається для перевірки в електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

1 травня 2026 року

Протокол аналізу звіту подібності експертом

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Ілля УСТІНОВ

Співавтор:

Назва: Комп'ютерна система аналізу інтернет-трафіку

Експерт: Євген ФЕДОРОВ

Підрозділ: Кафедра комп'ютерної інженерії та інформаційних систем

Коефіцієнт подібності 1:2.3%

Коефіцієнт подібності 2:0.6%

Мікропробіли: 3

Заміна букв: 0

Інтервали: 0

Білі знаки: 14

Дата створення звіту: 2026-05-30 14:39:20.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

2026-05-30

Дата

Доцент Андрій Нічепорук

експерт

Anti-Plagiarism (http://ap.km.ua) v-15.701

Максимальне співпадіння з одним документом 1.0%

Словники перевірки: en_US, ru_RU, ua_UA. **Помилок в документах: 10%**

	Документ		Сумарний збіг по Базі Даних	
ID: 272847 Назва: БКР Комп'ютерна система аналізу інтернет-трафіку Додано в БД: 2026-05-30 Автора: Ілля УСТІНОВ Керівники: Євген ФЕДОРОВ Консультанти: Опоненти:	Символи	Лексеми	Символи	Лексеми
	127889	1164	2236 (2%)	30 (3%)

Джерело плагіату

ID	Опис	Наявність плагіату в документі	
		Символи	Лексеми