Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук
(повна назва)
Кафедра програмної інженерії
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти другий (магістерський)

Дослідження методів розпізнавання зловмисного зашифрованого трафіку для захисту хмарних систем від DDoS атак. Використання RNN у поєднанні з AutoEncoder
(тема)

Виконав:
здобувач 2 року навчання
групи ІПЗм-23-4
Андрій ТРИПІЛКА
(власне ім'я, прізвище)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення
(повна назва освітньої програми)

Керівник доц. Наталя КРАВЕЦЬ
(посада, власне ім'я, прізвище)

Допускається до захисту
Зав. кафедри _____ Кирило СМЕЛЯКОВ
(підпис) (власне ім'я, прізвище)
2025 р.

Харківський національний університет радіоелектроніки

Факультет комп'ютерних наук

Кафедра програмної інженерії

Рівень вищої освіти другий (магістерський)

Спеціальність 121 – Інженерія програмного забезпечення
(код і повна назва)

Тип програми освітньо-наукова програма
(освітньо-професійна або освітньо-наукова)

Освітня програма Інженерія програмного забезпечення
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«___» _____20___ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Трипілці Андрію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи «Дослідження методів розпізнавання зловмисного зашифрованого трафіку для захисту хмарних систем від DDoS атак. Використання RNN у поєднанні з AutoEncoder». затверджена наказом університету від 15 квітня 2025 р. № 290 Ст
2. Термін подання здобувачем роботи до екзаменаційної комісії 24 червня 2025 р.
3. Вихідні дані до роботи «*наукові публікації, інтернет-джерела та технічна документація щодо аналізу зашифрованого трафіку, виявлення DDoS-атак, методів машинного навчання (RNN, AutoEncoder), а також публічні датасети та інструменти для роботи з мережевими потоками».*

Перелік питань, що потрібно опрацювати у роботі «*мета роботи, аналіз предметної галузі і постановка задачі, огляд та аналіз літературних джерел з дослідження, дослідження теоретичне, дослідження практичне».*

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Отримання завдання	16.04. 25	*виконано*
2	Аналіз предметної галузі	21.04.25 – 25.04.25	*виконано*
3	Огляд й аналіз літературних, наукових джерел	25.04.25 – 29.04.25	*виконано*
4	Теоретичне дослідження	30.04.25 – 15.05.25	*виконано*
5	Архітектура та проектування системи	15.05.25 – 30.05.25	*виконано*
6	Опис експерементальних досліджень	01.06.25 – 06.06.25	*виконано*
7	Підготовка до апробації результатів дослідження. Публікація матеріалів	06.06.25 – 08.06.25	*виконано*
8	Підготовка пояснювальної записки	08.06.25 – 16.06.25	*виконано*
9	Підготовка презентації та доповіді	16.06.25 – 17.06.25	*виконано*
10	Перевірка на плагіат	16.06.25	*виконано*
11	Нормоконтроль	19.06.25	*виконано*
12	Рецензування	19.06.25 – 21.06.25	*виконано*
13	Попередній захист	23.06.25	*виконано*
14	Занесення диплома в електронний архів	23.06.25	*виконано*
15	Допуск до захисту у зав. кафедри	23.06.25	*виконано*

Дата видачі завдання 15 квітня 2025р.

Здобувач

(підпис)

Керівник роботи _____ доц. Наталя КРАВЕЦЬ

(підпис) (посада, власне ім’я, прізвище)

РЕФЕРАТ / ABSTRACT

Робота містить: 86 с., 19 рис., 1 табл., 12 джерел, 4 додатки

ЗАШИФРОВАНИЙ ТРАФІК, DDoS-АТАКИ, ХМАРНІ СИСТЕМИ, МАШИННЕ НАВЧАННЯ, ВИЯВЛЕННЯ ЗАГРОЗ, AUTOENCODER, RNN, КІБЕРБЕЗПЕКА.

Об’єктом дослідження є процес аналізу зашифрованого мережевого трафіку у хмарних середовищах для виявлення DDoS-атак.

Метою цієї роботи є дослідження методів виявлення зашифрованого зловмисного трафіку з акцентом на DDoS-атаки у хмарному середовищі та розробка прототипу системи виявлення на основі рекурентних нейронних мереж (RNN) у поєднанні з AutoEncoder.

Методами розробки є аналіз проблемної області, проєктування архітектури системи, побудова моделей виявлення аномалій, експериментальне дослідження ефективності RNN та AutoEncoder у задачах кібербезпеки, а також використання сучасних інструментів для обробки мережевого трафіку. У роботі також порівнюються запропоновані методи з класичними евристичними підходами для оцінки переваг глибокого навчання. Особливу увагу приділено адаптивності системи до роботи в реальному часі з потоковими даними.

У результаті науково-дослідної практики створено прототип системи, що включає архітектуру, обрані моделі глибокого навчання, опис вхідних даних і метрик оцінювання. Отримані результати підтвердили ефективність запропонованих рішень та створили базу для подальшого впровадження в хмарні сервіси.

ENCRYPTED TRAFFIC, DDOS ATTACKS, CLOUD SYSTEMS, MACHINE LEARNING, THREAT DETECTION, AUTOENCODER, RNN, CYBERSECURITY.

The object of this study is the process of analyzing encrypted network traffic in cloud environments for detecting DDoS attacks.

The aim of the work is to explore methods for identifying malicious encrypted traffic, focusing on DDoS attacks in cloud infrastructure, and to develop a detection system prototype based on recurrent neural networks (RNN) combined with an AutoEncoder.

The applied methods include problem domain analysis, system architecture design, anomaly detection model development, experimental evaluation of RNN and AutoEncoder performance in cybersecurity tasks, as well as the use of modern tools for processing network flows. The work also compares the proposed methods with classical heuristic approaches to assess the advantages of deep learning. Special attention is paid to the adaptability of the system to real-time processing of streaming data.

As a result of the research project, a functional prototype was created, including architecture, selected deep learning models, data structures, and evaluation metrics. The obtained results confirmed the effectiveness of the proposed solution and provided a solid foundation for further implementation in cloud-based security systems.

Заява щодо самостійного виконання кваліфікаційної роботи та можливості її публікації в електронному архіві відкритого доступу ElArKhNURE.

Завідувачу кафедри
ПІ_____
(скорочена назва кафедри)
проф. Кирилу СМЕЛЯКОВУ_
(вчене звання, сласне ім'я, прізвище)

ЗАЯВА
щодо самостійності виконання кваліфікаційної роботи та можливості її публікації (та/або публікації анотації кваліфікаційної роботи) в електронному архіві відкритого доступу ElAr KhNURE

Я, _____Трипілка Андрій Володомирович_____
(прізвище, ім'я, по батькові)

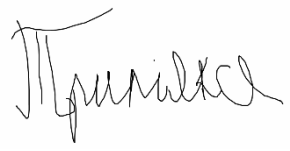
здобувач вищої освіти на другому (магістерському) рівні вищої освіти академічної групи ___ІПЗм-23-4_____

кафедра_____ програмної інженерії _____,
(повна назва кафедри)

заявляю: моя кваліфікаційна робота на тему _ Дослідження методів розпізнавання зловмисного зашифрованого трафіку для захисту хмарних систем від DDoS атак. Використання RNN у поєднанні з AutoEncoder_____
(назва роботи)

що буде представлена в екзаменаційну комісію для публічного захисту, виконана самостійно, в ній не містяться елементи плагіату і вона може бути опублікована в репозиторії "ElArKhNURE". Погоджуюся з авторським договором, відповідно до Положення про репозиторій ХНУРЕ "ElArKhNURE". Всі запозичення з друкованих та електронних джерел мають відповідні посилання.

Я ознайомлений (а) з вимогами академічної доброчесності, згідно з якими виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування дисциплінарних заходів.

Дата 22.06.2025 Підпис 

ЗМІСТ

ВСТУП

Сучасний розвиток інформаційних технологій призвів до значного зростання обсягів зашифрованого мережевого трафіку, що ускладнило завдання виявлення кіберзагроз. Зокрема, особливу небезпеку для хмарних систем становлять DDoS-атаки, які можуть здійснюватися через шифровані з’єднання, маскуючись під легітимну активність.

Метою цієї роботи є дослідження методів виявлення зашифрованого зловмисного трафіку з акцентом на DDoS-атаки у хмарному середовищі та розробка прототипу системи виявлення на основі рекурентних нейронних мереж (RNN) у поєднанні з AutoEncoder. Основними завданнями дослідження є аналіз сучасних підходів до обробки зашифрованого трафіку, вивчення методів збору та аналізу мережевих потоків, дослідження ефективності RNN та AutoEncoder у задачах виявлення аномалій, а також формування теоретичних рекомендацій щодо побудови архітектури системи.

Об’єктом дослідження є процеси аналізу зашифрованого трафіку в хмарних середовищах з метою виявлення DDoS-атак. Предметом дослідження є методи машинного навчання, зокрема RNN та AutoEncoder, які дозволяють виявляти аномальні шаблони поведінки у мережевих потоках.

У ході роботи використовуються такі методи: аналіз літературних джерел з кібербезпеки та глибокого навчання, обробка даних мережевого трафіку, побудова та навчання моделей RNN та AutoEncoder, експериментальна перевірка ефективності моделей на тестових датасетах. Такий підхід дозволяє глибше дослідити особливості виявлення шкідливої активності у зашифрованому трафіку та сформувати практичні рекомендації щодо впровадження систем захисту в хмарних середовищах.

Очікувані результати дослідження включають створення теоретичної основи для побудови інтелектуальних систем виявлення DDoS-атак у зашифрованому трафіку, а також розробку прототипу рішення, що забезпечує ефективний моніторинг мережевого трафіку та підвищує рівень безпеки хмарних сервісів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ

1.1 Огляд предметної галузі

Аналіз зашифрованого мережевого трафіку у хмарних середовищах є однією з найактуальніших задач сучасної кібербезпеки. Шифрування трафіку, з одного боку, забезпечує конфіденційність переданих даних, з іншого — значно ускладнює виявлення загроз, зокрема розподілених атак типу DDoS. Усе більше зловмисників використовують TLS/SSL-з'єднання як механізм маскування шкідливої активності, що робить традиційні методи, такі як глибокий аналіз пакетів (DPI), малоефективними або навіть непридатними.

Зашифровані DDoS-атаки здатні порушувати роботу хмарних сервісів, створювати перевантаження серверів, блокувати доступ до ресурсів, а також викликати значні фінансові та репутаційні збитки. Особливо небезпечними є атаки нового покоління, що імітують легітимну поведінку користувачів, надсилаючи множину з'єднань із коректно сформованими зашифрованими запитами. Такий трафік майже не відрізняється від звичайного з точки зору структури пакетів, що створює серйозні труднощі для виявлення [1].

Масштаби цієї проблеми постійно зростають. За оцінками аналітичних центрів, понад 80% інтернет-трафіку наразі шифрується, а частка DDoS-атак, які використовують шифрування, подвоїлася за останні три роки. Атаки стають дедалі складнішими, застосовуючи ботнети з географічно розподілених вузлів, змінюючи поведінку в реальному часі для уникнення детекції [2]. Водночас зловмисники активно адаптуються до нових механізмів захисту, що знижує ефективність класичних систем моніторингу.

Особливістю задачі аналізу зашифрованого трафіку є велика кількість неоднорідних даних: метрики сесій, часові інтервали між пакетами, розміри та напрямки передачі, статистика потоків, заголовки протоколів, тощо [3]. Через обсяги, швидкість надходження та складність структури таких даних традиційні методи аналізу (rule-based або сигнатурні) часто виявляються недостатньо ефективними. Це створює потребу у використанні інтелектуальних підходів —

зокрема, моделей глибокого навчання, які здатні самостійно вивчати закономірності з великих масивів даних.

На практиці більшість існуючих рішень або орієнтовані на відкритий трафік, або не враховують особливості хмарної інфраструктури. Вони часто виявляють лише відомі патерни атак і неефективні при зіткненні з новими типами DDoS, які не мають сигнатур. Також обмеженням є повільна реакція та низька точність при роботі в умовах великого навантаження.

Таким чином, проблема виявлення DDoS-атак у зашифрованому трафіку потребує нових підходів, здатних працювати в реальному часі, масштабуватись відповідно до обсягів хмарної інфраструктури та виявляти невідомі загрози. Потенційно ефективним вирішенням цього завдання є застосування технологій Big Data для обробки великих обсягів мережевих потоків у поєднанні з методами глибокого навчання, такими як рекурентні нейронні мережі (RNN) і AutoEncoder.

1.2 Огляд сучасних підходів до аналізу зашифрованого трафіку

Зі зростанням частки зашифрованого трафіку у глобальному інтернет-просторі, задачі виявлення загроз, зокрема DDoS-атак, вимагають використання нових підходів, які не залежать від доступу до вмісту пакетів. Традиційні системи, орієнтовані на аналіз payload'у або сигнатур, втрачають свою ефективність через широке використання протоколів TLS/SSL, а також стрімке розповсюдження технологій, що забезпечують конфіденційність (VPN, DNS over HTTPS тощо).

Найпоширенішим підходом до аналізу зашифрованого трафіку є аналіз потоків (flow-based analysis). Він ґрунтується на оцінці статистичних характеристик мережевих потоків: кількості пакетів, середнього розміру, часу між передачею, напрямку трафіку, кількості унікальних з'єднань тощо. Потоки агрегуються з використанням інструментів на зразок NetFlow, IPFIX або CICFlowMeter, які дозволяють формувати набори ознак для подальшої обробки алгоритмами машинного навчання.

Один із поширених напрямів — виявлення аномалій на основі статистики. Тут застосовуються методи кластеризації (наприклад, k-means), алгоритми

локальної щільності (LOF), або дерева ізоляції (Isolation Forest), що дозволяють виявляти трафік, який відрізняється від "норми". Такий підхід особливо ефективний для виявлення нових, ще не ідентифікованих шаблонів атак, проте він чутливий до якості ознак та часто має високий рівень хибнопозитивних спрацювань.

Іншим потужним напрямом є використання моделей машинного навчання, зокрема нейронних мереж. Розроблено ряд підходів, які застосовують класифікацію трафіку на основі ознак потоків, використовуючи такі алгоритми як Decision Trees, Random Forest, SVM, або логістичну регресію. Ці моделі дозволяють швидко класифікувати потоки як шкідливі або легітимні, однак мають обмеження при роботі з послідовними даними, які характерні для DDoS-атак.

У зв’язку з цим зростає популярність глибоких моделей, зокрема рекурентних нейронних мереж (RNN), які здатні обробляти послідовності ознак та виявляти закономірності у часових рядах. Для задач виявлення DDoS-атак у зашифрованому трафіку особливо ефективними є архітектури на базі LSTM, що дозволяють виявляти повторювані або підозрілі шаблони в поведінці потоків.

Ще один перспективний підхід — використання AutoEncoder для виявлення аномалій. AutoEncoder навчається відновлювати нормальний трафік і при цьому дає високу помилку реконструкції на незнайомих або аномальних зразках. Це дозволяє виявляти нові види DDoS, які ще не були внесені в бази даних.

Окрему нішу займають гібридні моделі, які поєднують декілька підходів. Наприклад, CNN-LSTM, де згорткові шари виділяють локальні патерни, а рекурентні — вловлюють їхню динаміку. Також існують варіанти поєднання AutoEncoder з класифікаторами для підвищення точності детекції. Деякі системи, зокрема ті, що використовуються в середовищах SDN або у хмарній інфраструктурі, інтегрують виявлення на основі потоків у реальному часі, застосовуючи моделі глибокого навчання у масштабованому середовищі (наприклад, на Apache Spark або Kubernetes).

Попри значні успіхи, сучасні підходи мають і недоліки. Серед основних викликів — високе споживання обчислювальних ресурсів, складність адаптації

моделей до нових умов, а також потреба у великих обсягах якісних даних для навчання. Крім того, "чорний ящик" нейромереж створює труднощі у поясненні рішень моделей, що є критичним для систем безпеки.

Таким чином, підходи до аналізу зашифрованого трафіку постійно вдосконалюються. Найбільшу перспективу мають методи глибокого навчання, зокрема RNN та AutoEncoder, що демонструють високу точність виявлення DDoS-атак навіть за відсутності доступу до вмісту мережевих пакетів.

1.3 Особливості аналізу зашифрованого трафіку у хмарних середовищах

Аналіз зашифрованого трафіку у хмарних інфраструктурах має низку специфічних особливостей, які суттєво впливають на вибір підходів до виявлення загроз, зокрема DDoS-атак. Хмарні сервіси працюють у динамічному та масштабованому середовищі, де одночасно обслуговуються тисячі користувачів і постійно змінюються конфігурації мережі, що ускладнює відстеження типових шаблонів трафіку.

Шифрування трафіку на рівні протоколів TLS/SSL є стандартною практикою для забезпечення конфіденційності даних у хмарі. Водночас це призводить до того, що традиційні системи виявлення загроз, які покладаються на аналіз вмісту пакетів, не можуть ефективно функціонувати. У таких умовах основним джерелом інформації для аналізу стають метадані потоків: статистика кількості пакетів, розміру, напрямку, часу взаємодії, кількості сесій та інших ознак, які не потребують розшифрування.

Характерною рисою DDoS-атак у хмарному середовищі є те, що вони можуть імітувати легітимний трафік, наприклад, регулярні HTTPS-запити до вебсервісів, або створювати низькоінтенсивне навантаження з багатьох джерел, що утруднює їхню ідентифікацію. У таких випадках критично важливо враховувати послідовність дій та поведінкові характеристики трафіку, що робить актуальним застосування рекурентних нейронних мереж (RNN), здатних аналізувати часові залежності між подіями.

Ще однією особливістю є вимога реального часу — для забезпечення ефективного захисту хмарних ресурсів система виявлення повинна обробляти потоки без затримок. Це накладає обмеження на обчислювальну складність моделей і потребує оптимізації обробки трафіку. RNN та AutoEncoder, попри свою складність, можуть бути адаптовані до потокової обробки завдяки використанню батчів і спрощених архітектур (наприклад, LSTM, або shallow AE замість глибоких стекових).

Крім того, важливим чинником є відсутність розмітки у реальному трафіку, що робить актуальними підходи з частковим або повним навчанням без учителя. У цьому контексті автоенкодери є ефективним інструментом виявлення аномалій, оскільки дозволяють побудувати модель "нормального" трафіку та виявляти відхилення без потреби у ручному маркуванні даних.

Важливу роль відіграє також масштабованість системи, оскільки хмарна інфраструктура може включати велику кількість віртуальних машин, контейнерів, сервісів тощо. Система виявлення атак повинна бути здатна обробляти трафік з багатьох джерел паралельно, мати централізований збір потоків і підтримку інтеграції з іншими елементами хмарної безпеки.

Таким чином, аналіз зашифрованого трафіку в хмарному середовищі потребує інтелектуальних, гнучких і продуктивних рішень. Найбільш ефективними на сьогодні є методи, що поєднують моделі глибокого навчання з аналізом потокових характеристик — зокрема, RNN для моделювання часової структури трафіку та AutoEncoder для виявлення аномалій у поведінці мережевих потоків.

1.4 Тенденції та виклики у сфері захисту хмарних систем

З поширенням шифрування мережевого трафіку, особливо у хмарних середовищах, значно зросли вимоги до систем виявлення кіберзагроз. Останні роки відзначаються зростанням кількості складних DDoS-атак, які здійснюються через зашифровані канали зв’язку, що суттєво ускладнює їхнє виявлення класичними методами. На цьому тлі сформувались ключові тенденції та виклики, що визначають подальший розвиток захисних технологій.

Однією з основних тенденцій є масове використання TLS/SSL як стандарту для захисту з’єднань у мережі. Більшість легітимного трафіку шифрується, і зловмисники активно цим користуються, щоб маскувати шкідливу активність. У випадку DDoS-атак зловмисники часто використовують стандартні порти (наприклад, 443), імітуючи поведінку звичайних користувачів. Такі атаки важко виявити, оскільки вони не містять явних сигнатур або характерних ознак у payload, який зашифровано.

Ще однією тенденцією є зростання розподіленості та варіативності атак. Атаки можуть запускатися з великої кількості ботів, часто з географічно розподілених пристроїв, із використанням проксі, VPN або хмарних серверів, що ускладнює виявлення за IP або іншими фіксованими атрибутами. Більш того, DDoS-атаки еволюціонують: з'являються low-rate DDoS або multi-vector атаки, які чергують різні протоколи, обсяги, частоту трафіку, щоб уникати автоматичних систем захисту.

Серед ключових викликів — зниження ефективності традиційних методів виявлення, які базуються на аналізі вмісту пакетів (DPI) або простих правилах. Через зашифрований трафік DPI втрачає зміст, а сигнатурні методи не працюють проти нових типів атак, які ще не ідентифіковані.

Інший виклик — потреба в реальному часі реагування. У хмарних інфраструктурах атаки можуть призвести до швидкої деградації сервісів, тому критично важливо виявляти загрози до того, як буде завдано шкоди. Це вимагає високої продуктивності та швидкодії від систем аналізу.

Серйозною проблемою є наявність великого обсягу неструктурованих або неповністю мічених даних, що обмежує застосування методів із повним наглядом. У реальному трафіку рідко є точні мітки "атака/не атака", особливо коли мова йде про нові вектори загроз. Це створює потребу у використанні методів з частковим або безнаглядовим навчанням (наприклад, AutoEncoder), які здатні виявляти відхилення від нормальної поведінки без ручної розмітки.

Ще один важливий виклик — інтерпретованість моделей глибокого навчання. Нейронні мережі, зокрема RNN, часто працюють як “чорна скринька”, і

пояснити, чому система позначила конкретний потік як зловмисний, буває складно. Це створює певні труднощі у верифікації та прийнятті рішень безпековими аналітиками.

Останній виклик пов’язаний з ресурсами: глибокі моделі потребують значних обчислювальних потужностей, що може бути проблемою в реальному середовищі з великим обсягом трафіку. Необхідно балансувати між точністю та швидкодією, обираючи оптимальні архітектури та оптимізуючи обробку потоків.

Таким чином, захист хмарних систем від зашифрованого зловмисного трафіку потребує комплексного підходу — поєднання адаптивних моделей машинного навчання, ефективного збору та обробки потокових метаданих, а також врахування обмежень продуктивності, інтерпретованості й масштабованості. Саме це визначає актуальність використання RNN та AutoEncoder у сучасних системах безпеки.

1.5 Постановка задачі

Зважаючи на зростання обсягів зашифрованого мережевого трафіку та складність виявлення DDoS-атак без доступу до вмісту пакетів, постає задача створення системи, здатної аналізувати лише метаінформацію потоків та виявляти підозрілу активність. Традиційні підходи виявлення, зокрема сигнатурні методи або фіксовані правила, втрачають ефективність у зашифрованих середовищах. У той же час, класичні алгоритми машинного навчання мають обмеження щодо узагальнення та потребують чітко розмічених даних. Це обґрунтовує необхідність використання глибоких нейронних мереж.

Метою цього дослідження є проєктування та реалізація прототипу системи виявлення DDoS-атак у зашифрованому трафіку на основі аналізу метаданих потоків. Ключовою вимогою є здатність системи працювати без доступу до payload-даних, лише на основі статистичних та часових ознак, що не порушують конфіденційності.

Для досягнення мети необхідно вирішити такі задачі:

— визначити набір інформативних ознак мережевого потоку, придатних для аналізу зашифрованого трафіку;

— реалізувати механізм побудови векторного представлення потоків на основі обраних ознак;

— побудувати AutoEncoder для зменшення розмірності вхідних даних без втрати ключових патернів;

— реалізувати модель рекурентної нейронної мережі (RNN), яка класифікує кожен потік як нормальний або такий, що має ознаки DDoS-атаки;

— реалізувати інтерфейс виводу результатів аналізу та візуалізації підозрілої активності в режимі реального часу;

— забезпечити збереження результатів у time-series сховище з можливістю подальшого аналізу.

Таким чином, у межах цього проєкту ставиться задача розробки адаптивної, масштабованої системи виявлення DDoS-активності у зашифрованому мережевому трафіку на основі гібридної глибокої моделі, що поєднує AutoEncoder та RNN.

2 ОГЛЯД Й АНАЛІЗ ЛІТЕРАТУРНИХ, НАУКОВИХ ДЖЕРЕЛ

Літературний огляд є важливим етапом наукового дослідження, оскільки дозволяє систематизувати наявні знання, оцінити сучасні підходи до аналізу зашифрованого трафіку та виявлення кіберзагроз, а також визначити існуючі прогалини у цій сфері. У контексті обраної теми – виявлення DDoS-атак у зашифрованому трафіку хмарних систем за допомогою методів глибокого навчання – літературний огляд є основою для формування методології дослідження та вибору інструментів.

Для цього дослідження були відібрані джерела, що відповідають кільком ключовим критеріям. По-перше, актуальність: розглядалися публікації останніх 5–10 років, що відображають новітні виклики, пов’язані з переходом до повністю зашифрованих мережевих середовищ. По-друге, наукова достовірність і авторитетність: аналізувалися рецензовані наукові статті, матеріали конференцій з кібербезпеки, публікації IEEE, Springer, Elsevier, а також звіти авторитетних аналітичних центрів, таких як ENISA, Cisco, Cloudflare. По-третє, об'єктивність: перевагу надавали джерелам із чітко описаною методикою та результатами.

Огляд структуровано за трьома основними напрямами, що відображають ключові компоненти теми дослідження. Перший напрям стосується аналізу зашифрованого трафіку та специфіки виявлення загроз у TLS-з’єднаннях. Другий охоплює підходи до виявлення DDoS-атак у хмарному середовищі з урахуванням масштабності, варіативності та потокового характеру даних. Третій напрям присвячений використанню методів глибокого навчання, зокрема рекурентних нейронних мереж (RNN) і автоенкодерів (AutoEncoder), у задачах виявлення аномалій та класифікації трафіку.

Такий підхід дозволяє сформувати всебічне уявлення про сучасні методи боротьби із зашифрованими DDoS-атаками, виявити найефективніші рішення та окреслити напрямки, які потребують подальших досліджень. Проведений огляд створює фундамент для побудови прототипу системи виявлення загроз, яка поєднує глибокі нейронні мережі з аналізом характеристик мережевих потоків у хмарному середовищі.

2.1 Огляд основних джерел

Для виконання дослідження було відібрано низку авторитетних джерел, що охоплюють ключові аспекти аналізу зашифрованого мережевого трафіку, виявлення DDoS-атак у хмарному середовищі, а також застосування методів глибокого навчання, зокрема RNN та AutoEncoder, у задачах кібербезпеки. Джерела згруповані за трьома основними тематичними напрямами, які безпосередньо відповідають структурі даного дослідження.

2.1.1 Загальні аспекти аналізу зашифрованого трафіку та DDoS-атак

У цьому блоці розглядаються особливості побудови та захисту сучасних мереж, зокрема хмарних, у контексті зашифрованого трафіку. Ключовим джерелом є стаття A. Callado та ін. “ A Survey on Internet Traffic Identification” [1], де систематизовано підходи до класифікації трафіку без доступу до вмісту пакетів. У роботі описано ефективність flow-based методів при аналізі TLS-з’єднань.

Огляд DDoS-атак у хмарному середовищі наведено в статті Bhatia S. та ін. “Distributed Denial of Service Attacks and Defense Mechanisms: Distributed Denial of Service Attacks and Defense Mechanisms: Current Landscape and Future Directions” [2], де розглянуто специфіку атак нового покоління, що використовують шифрування як засіб маскування.

2.1.2 Методи виявлення загроз у зашифрованому трафіку

У зв’язку зі стрімким зростанням частки зашифрованого трафіку в мережах, традиційні методи виявлення загроз, засновані на аналізі вмісту пакетів (Deep Packet Inspection), втрачають ефективність. У таких умовах основна увага зосереджується на методах аналізу метаданих — тобто характеристик трафіку, які не потребують дешифрування, зокрема: розмір пакетів, тривалість з’єднання, кількість байтів у потоці, інтервали між пакетами тощо.

Одним із фундаментальних джерел у цій тематиці є робота M. Lotfollahi та ін. “Deep packet: a novel approach for encrypted traffic classification using deep learning” [3], у якій запропоновано використання згорткових та рекурентних нейронних

мереж для класифікації зашифрованого трафіку. Автори доводять, що глибоке навчання може успішно ідентифікувати класи трафіку, навіть якщо його вміст недоступний для аналізу.

У статті Sharma A. та ін. “A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions” [4] проведено масштабний огляд методів класифікації мережевого трафіку, зокрема зашифрованого. Серед ключових підходів, що демонструють найкращі результати, виділено моделі на основі автоматичного видобутку ознак (feature learning), які замінюють ручне формування ознак, характерне для класичних ML-методів.

Також у роботі Anderson B. та ін. “ Machine Learning for Encrypted Malware Traffic Classification” [5] описано застосування моделей машинного навчання для виявлення шкідливої активності у TLS-трафіку. Автори вказують на важливість аналізу послідовних характеристик потоку, таких як послідовність розмірів пакетів та часові залежності між ними.

Окрему увагу приділено метрикам ефективності, які використовуються для оцінки моделей у цій сфері: Precision, Recall, F1-score, ROC-AUC, а також специфічним показникам для задач виявлення аномалій, зокрема reconstruction error у випадку використання автоенкодерів.

Узагальнено, сучасні методи виявлення загроз у зашифрованому трафіку тяжіють до гібридних архітектур, які поєднують виявлення аномалій (unsupervised) з класифікацією (supervised), забезпечуючи як гнучкість, так і точність при роботі в реальному середовищі, де загрози постійно змінюються.

2.1.3 Використання машинного навчання у прогнозуванні загроз

У сучасних системах виявлення загроз, особливо в умовах зашифрованого трафіку, все ширше застосовуються методи глибокого навчання, які здатні аналізувати складні залежності в мережевих потоках. Серед таких методів особливу роль відіграють рекурентні нейронні мережі (RNN) та автоенкодери

(AutoEncoder), які дозволяють обробляти послідовні дані та виявляти аномальні шаблони поведінки.

У роботі Yin C. та ін. “A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks” [6] показано ефективність застосування RNN, зокрема архітектури LSTM, для виявлення DDoS-атак на основі аналізу часових залежностей між подіями в мережі. Автори зазначають, що такі моделі здатні ідентифікувати характерні патерни поведінки ботнетів, навіть коли вони маскуються під легітимний трафік.

Дослідження A. Makhzani “Adversarial Autoencoders” [7] демонструє використання варіаційних автоенкодерів для задач виявлення аномалій. AutoEncoder навчається відтворювати нормальний трафік і сигналізує про аномалію в разі високої помилки реконструкції. Це дозволяє виявляти невідомі або нетипові форми DDoS-атак, без потреби у розмічених даних.

Інтегровані підходи, що поєднують AutoEncoder з класифікаційними моделями, описані у роботі Tang T. A. та ін. “Deep learning approach for Network Intrusion Detection in Software Defined Networking” [8]. У цьому дослідженні використано AutoEncoder для виявлення аномальних потоків, після чого результати подаються до RNN-класифікатора для точнішого визначення типу загрози. Такий підхід демонструє високі результати у сценаріях, пов’язаних з хмарними середовищами та зашифрованими з'єднаннями.

Також важливим джерелом є Lee M.-C. та ін. “ Impact of Recurrent Neural Networks and Deep Learning Frameworks on Real-time Lightweight Time Series Anomaly Detection.” [9], де узагальнено переваги використання RNN для обробки мережевих логів і потоків, з акцентом на довготривалі залежності та низьку потребу в ручному формуванні ознак.

Загалом, джерела вказують на високу ефективність використання RNN і AutoEncoder у виявленні DDoS-атак, особливо у складних умовах, таких як зашифрований трафік, нестабільне навантаження та динамічне хмарне середовище. Саме ці методи стали основою для розробки архітектури прототипу, запропонованого в межах цього дослідження.

2.2 Аналіз літератури

Аналіз обраних літературних джерел дозволяє виявити ключові теорії, концепції та моделі, що формують основу дослідження в галузі виявлення зловмисного зашифрованого трафіку, зокрема у контексті DDoS-атак проти хмарних інфраструктур. Особливу увагу приділено підходам, які дозволяють працювати з мережевими потоками без розшифрування вмісту, а також методам глибокого навчання, здатним обробляти складні послідовні залежності в трафіку.

2.2.1 Основні теорії та концепції

У роботі Bhatia S. та ін. “Distributed Denial of Service Attacks and Defense Mechanisms: Current Landscape and Future Directions” [2] викладено загальні концепції побудови систем виявлення DDoS-атак, з акцентом на специфіку атак у зашифрованих з’єднаннях. Автори описують різні рівні захисту (мережевий, транспортний, прикладний) та вказують на потребу у зміщенні фокусу з аналізу вмісту пакетів на аналіз їхньої поведінки.

У роботі Sharma A. та ін. “A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions” [4] представлено концепцію класифікації трафіку на основі метаданих, без розкриття вмісту. Це лягло в основу підходів, які застосовуються до TLS-з’єднань, і дозволяє виявляти шкідливу активність у зашифрованих потоках.

2.2.2 Моделі та методи аналізу

Методи виявлення аномалій на основі глибокого навчання розглядаються у роботі A. Makhzani “Adversarial Autoencoders” [7]. В ній описано використання варіаційних автоенкодерів для розпізнавання нетипового трафіку через помилку реконструкції. Цей підхід є корисним у ситуаціях, коли точне маркування даних неможливе або економічно недоцільне.

У дослідженні Yin C. та ін. “A deep learning approach for intrusion detection using recurrent neural networks” [6] запропоновано використання рекурентних нейронних мереж (зокрема LSTM) для аналізу послідовностей мережевого трафіку.

Перевагою RNN є здатність виявляти довготривалі залежності у часових рядах, що важливо для розпізнавання складних DDoS-атак, які розгортаються поступово.

Комбіновані підходи описані у роботі Tang T. A. та ін. “Deep learning approach for network intrusion detection in Software Defined Networking” [8], де використано AutoEncoder як фільтр для виявлення підозрілих потоків, а RNN — для остаточного класифікування загроз. Такий підхід демонструє високу точність і стійкість до нових типів атак.

2.2.3 Ефективність існуючих підходів

Незважаючи на розвиток методів глибокого навчання, література підкреслює певні обмеження існуючих підходів. Зокрема, традиційні сигнатурні методи втрачають ефективність у зашифрованих середовищах. Класичні ML-моделі, як-от Random Forest або SVM, потребують ручного формування ознак, що не завжди можливо в умовах великого обсягу потокових даних.

Глибокі нейронні мережі, зокрема AutoEncoder та RNN, демонструють високу ефективність, однак мають свої виклики — потребують великих обсягів навчальних даних, потужних обчислювальних ресурсів, а також механізмів для інтерпретації рішень, що важливо у сфері кібербезпеки.

Крім того, актуальним залишається питання масштабованості моделей, оскільки хмарні середовища генерують величезні обсяги мережевого трафіку, які потрібно обробляти у реальному часі. Це вимагає ефективної інтеграції з потоковими системами обробки даних.

Таким чином, літературний аналіз показав, що найбільш перспективними для вирішення поставленої задачі є гібридні моделі, які поєднують переваги AutoEncoder для виявлення аномалій і RNN для вивчення часової структури трафіку. Саме ці методи були обрані як основа для розробки архітектури експериментального прототипу системи виявлення DDoS-атак у зашифрованому трафіку.

2.3 Оцінка актуальності та новизни

Аналіз наукових публікацій підтверджує високу актуальність проблеми виявлення DDoS-атак у зашифрованому трафіку, особливо у хмарних середовищах. З кожним роком кількість зашифрованих з’єднань у мережі зростає, і вже зараз понад 80% глобального трафіку передається через TLS. Водночас зловмисники дедалі частіше використовують шифрування як інструмент для маскування атак, що робить традиційні засоби виявлення малоефективними.

Наявні підходи, зокрема сигнатурні системи або засоби глибокого аналізу пакетів (DPI), більше не можуть гарантувати надійного захисту. У цьому контексті, застосування методів машинного та глибокого навчання, які здатні працювати на основі метаданих та поведінкових характеристик потоків, виглядає найбільш перспективним напрямом розвитку.

Наукова новизна дослідження полягає у поєднанні AutoEncoder та RNN для створення адаптивної системи виявлення загроз у зашифрованому трафіку. Така комбінація дозволяє одночасно:

— виявляти невідомі аномальні шаблони без потреби в маркованих даних (AutoEncoder),

— аналізувати послідовну природу поведінки потоків та виявляти типові патерни DDoS-атак (RNN),

— масштабувати рішення для обробки потоків у реальному часі.

Актуальність підходу також підсилюється потребами хмарних платформ, де навантаження змінюється динамічно, а швидке реагування на загрози є критично важливим. У літературі поки що небагато рішень, які ефективно працюють саме з зашифрованим трафіком у хмарі, тож запропонована архітектура має потенціал заповнити цю нішу.

Таким чином, розробка прототипу системи виявлення DDoS-атак, що базується на комбінації AutoEncoder та RNN, є як науково новаторським підходом, так і практично значущим кроком у напрямку посилення кіберзахисту сучасних цифрових інфраструктур.

2.4 Висновки з огляду

Огляд літератури дозволив систематизувати сучасні наукові та технічні підходи до виявлення загроз у зашифрованому мережевому трафіку та методів захисту хмарних інфраструктур від DDoS-атак. Було визначено, що використання традиційних підходів, таких як сигнатурний аналіз або фільтрація за IP, стає все менш ефективним у контексті зростання частки зашифрованого трафіку.

Аналіз наукових джерел підтвердив ефективність AutoEncoder як базової архітектури для виявлення аномалій на основі реконструкційної похибки. RNN, зокрема LSTM, демонструють високу здатність до моделювання часових залежностей, що дозволяє уточнювати характер загрози у динаміці. Комбінування цих двох підходів у рамках єдиної системи дозволяє ефективно фільтрувати потоки та класифікувати підозрілі зразки навіть без доступу до вмісту пакетів.

Також було виявлено, що більшість успішних рішень реалізуються у масштабованих середовищах з використанням інструментів Big Data, таких як Apache Spark. Візуалізація результатів, налаштування моделей та підтримка інтерактивного аналізу є важливими для практичного застосування у Security Operations Center (SOC).

Таким чином, літературний аналіз підтвердив актуальність вибраної теми, обґрунтував доцільність використання AutoEncoder у парі з RNN, а також сформував методологічну основу для реалізації системи виявлення DDoS у зашифрованому трафіку в хмарних середовищах.

3 ТЕОРЕТИЧНЕ ДОСЛІДЖЕННЯ

Цей розділ присвячено опису методів, алгоритмів, інструментів та технічних рішень, які використовуються для побудови системи виявлення DDoS-атак у зашифрованому трафіку. Особлива увага приділяється архітектурі програмного забезпечення, моделюванню компонентів системи, вибору технологій та підходів до аналізу мережевого трафіку без доступу до його вмісту.

У рамках дослідження розробляється інструмент, що поєднує автоенкодери для виявлення аномалій і рекурентні нейронні мережі для аналізу поведінкових патернів потоків. Пропонована система орієнтована на роботу в хмарному середовищі, з урахуванням вимог до масштабованості, ефективності в реальному часі та обробки зашифрованих з’єднань.

Метою цього розділу є побудова концептуальної та технічної бази майбутнього прототипу: опис архітектури, логіки взаємодії між модулями, принципів зберігання та обробки даних, а також методів інтеграції з іншими системами кібербезпеки.

3.1 Архітектура та проектування ПЗ

Система виявлення DDoS-атак у зашифрованому трафіку передбачає побудову модульної архітектури, яка забезпечує розділення функціональності, підтримку гнучкої конфігурації та можливість масштабування під різні обсяги трафіку. Основна логіка системи базується на потоковій обробці даних, з подальшим виявленням загроз.

3.1.1 Загальна структура архітектури

Архітектура системи передбачає чотири основні компоненти:

— модуль збору та підготовки даних: відповідальний за обробку PCAP-файлів або мережевого моніторингу, витягнення ознак з потоків за допомогою таких інструментів, як CICFlowMeter;

— модуль аналізу: реалізує двоетапну модель: AutoEncoder визначає аномальні потоки, а RNN (наприклад, LSTM) аналізує їхній часовий контекст для уточнення наявності загрози;

— база даних: базується на комбінованому використанні PostgreSQL (для результатів класифікації) та Redis (для сирих даних потоків і логів);

— інтерфейс користувача: реалізований у вигляді дашборду з візуалізацією загроз, графіками активності та можливістю взаємодії з системою в реальному часі.

3.1.2 Архітектурний стиль

Система побудована за принципами мікросервісної архітектури, що дозволяє запускати окремі компоненти (модуль збору, модуль аналізу, зберігання, інтерфейс) у вигляді незалежних сервісів. Для комунікації між компонентами може використовуватись REST API.

3.1.3 Візуалізація

Для кращого розуміння структури системи виявлення DDoS-атак у зашифрованому трафіку було розроблено низку діаграм, що ілюструють логіку роботи, взаємозв’язки компонентів та сценарії взаємодії користувача з системою. Ці візуальні моделі відображають ключові функціональні елементи програмного забезпечення, побудованого на основі мікросервісної архітектури.

Use Case діаграма моделює основні сценарії використання системи з боку користувача. Зокрема, користувач може запускати аналіз трафіку, переглядати список підозрілих потоків, здійснювати пошук за IP-адресою, змінювати налаштування алгоритмів та експортувати результати. Ці сценарії відображено на рисунку 3.1.

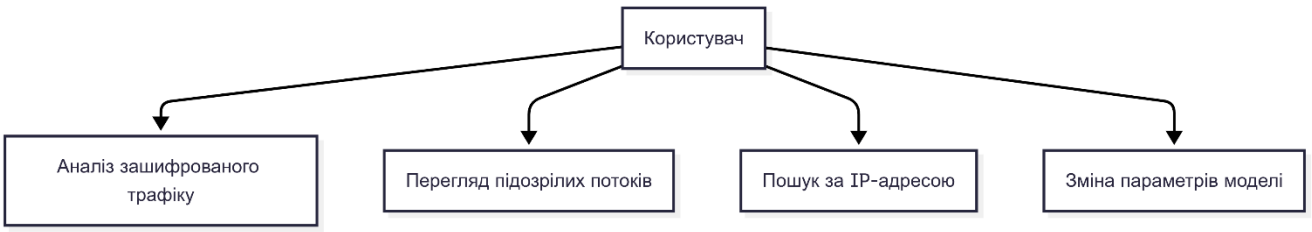


Рисунок 3.1 – Use Case діаграма взаємодії користувача з системою (рисунок виконано самостійно)

UML-діаграма компонентів (рис. 3.2) відображає структуру системи з позиції її основних модулів і способу їх взаємодії. Вона включає:

— зовнішнє джерело трафіку (PCAP або live),

— модуль збору та підготовки даних (FeatureExtractor),

— модуль аналізу (AutoEncoder та RNN),

— сховище результатів (PostgreSQL і InfluxDB),

— інтерфейс користувача з REST API.

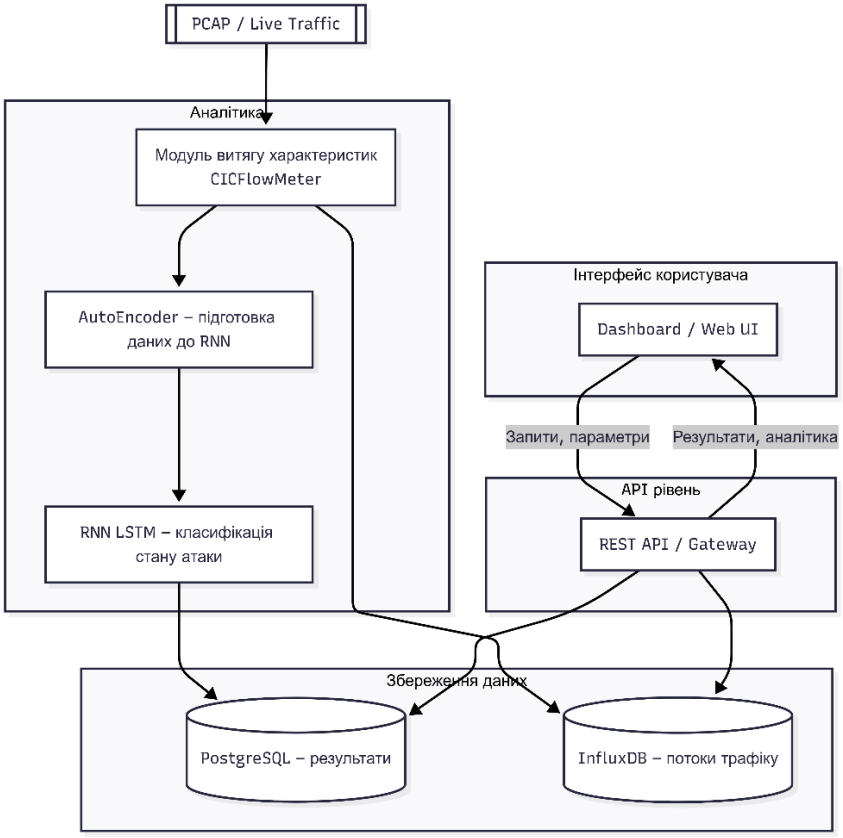


Рисунок 3.2 – UML-діаграма компонентів системи виявлення DDoS-атак (рисунок виконано самостійно)

ER-діаграма бази даних (рис. 3.3) демонструє логічну структуру даних, що використовуються системою. Основними сутностями є:

— Flows – мережеві потоки;

— AnalysisResults – результати обробки;

— Users – зареєстровані користувачі;

— Logs – дії користувачів у системі.

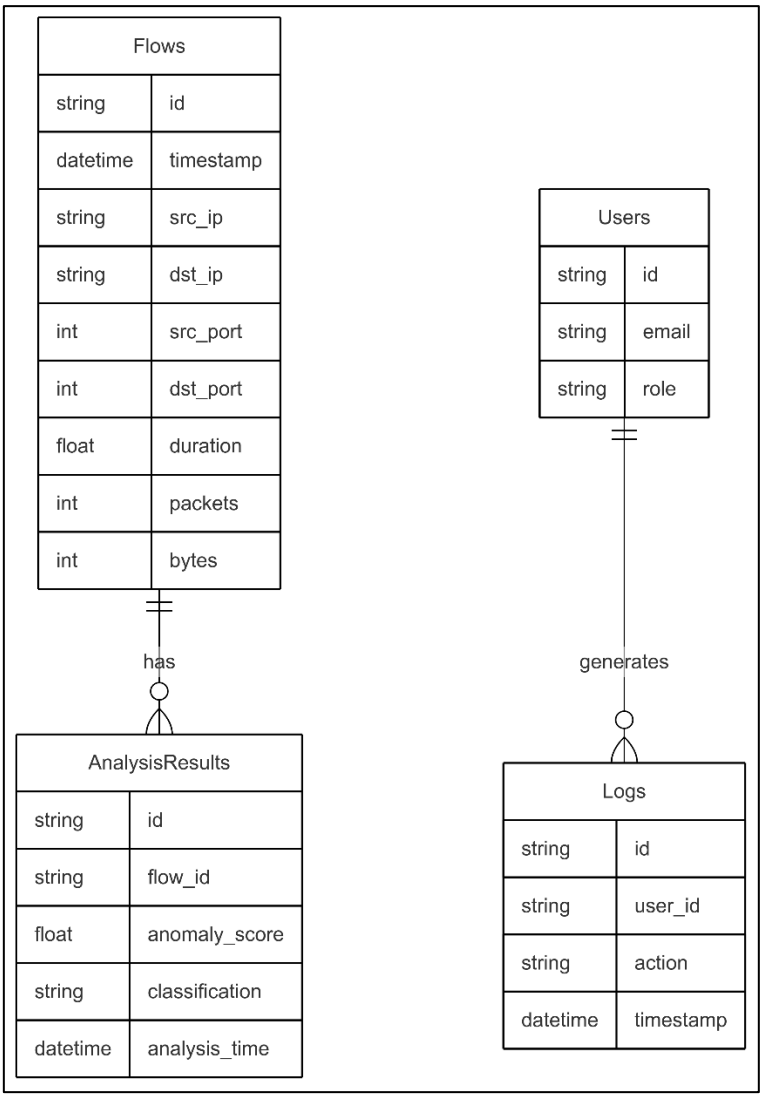


Рисунок 3.3 – Схема бази даних (рисунок виконано самостійно)

Усі візуалізації створено відповідно до логіки, закладеної в архітектурі системи, та забезпечують чітке розуміння її структури, ролей і взаємодій. Це дозволяє легко масштабувати рішення, додавати нові джерела трафіку або моделі аналізу без істотних змін основної логіки.

3.2 Проектування структури зберігання даних

Ефективне функціонування системи виявлення DDoS-атак вимагає належної організації зберігання та обробки мережевих метаданих. У цьому підрозділі розглядається структура зберігання даних, яка забезпечує зручний доступ до поточних та історичних потоків, підтримує процес навчання RL-агента, а також дозволяє зберігати результати класифікації, стани середовища та статистику взаємодій у процесі роботи системи.

3.2.1 Вибір технологій

Проєктування системи зберігання даних у контексті виявлення DDoS-атак у зашифрованому трафіку передбачає обробку великої кількості потокових метаданих у режимі реального часу або з високою частотою оновлення. Тому вибір технологій зберігання та доступу до даних має базуватись на критеріях масштабованості, продуктивності, гнучкості у структурі даних, а також простоти інтеграції з аналітичними модулями та моделями навчання.

Для зберігання потокових даних було обрано базу даних типу time-series InfluxDB. Такі системи оптимізовані для роботи з часовими рядами і добре підходять для зберігання агрегованих характеристик мережевих потоків: розмірів пакетів, частоти запитів, інтервалів між сеансами тощо. Вони дозволяють ефективно виконувати запити з фільтрацією за часовими вікнами, що важливо при побудові профілів трафіку та аналізі трендів.

Для зберігання додаткових даних, зокрема результатів класифікації, станів агента, інформації про дії та значення функцій винагороди, доцільно використовувати реляційну СУБД, таку як PostgreSQL. Вона забезпечує структуроване зберігання, підтримку складних запитів та є сумісною з більшістю фреймворків для машинного навчання.

Також у системі використовується формат зберігання даних у вигляді CSV або Parquet-файлів для підготовки навчальних вибірок, збереження результатів експериментів та обміну даними між компонентами. Такий підхід є простим у

реалізації, забезпечує кросплатформеність і добре підтримується бібліотеками обробки даних (Pandas, PyTorch).

У разі масштабування системи або переходу до хмарного середовища (наприклад, AWS), можлива інтеграція з S3-сховищами для архівації історичних даних, а також використання Amazon RDS для підтримки PostgreSQL та Amazon Timestream для потокової аналітики.

Таким чином, вибрані технології дозволяють забезпечити надійне, ефективне та масштабоване зберігання даних для потреб як аналітики, так і машинного навчання в рамках реалізації системи на основі reinforcement learning.

3.2.2 Схема бази даних

Для зберігання та обробки мережевого трафіку, результатів аналізу та логів взаємодії з системою розроблено реляційну модель бази даних, яка відповідає вимогам до структурованого зберігання метаданих потоків, результатів класифікації та аудиту дій користувачів.

— Схема бази даних включає чотири основні сутності:

— Flows — центральна таблиця, яка містить інформацію про окремі мережеві потоки. Вона зберігає такі поля: унікальний ідентифікатор потоку (id), мітку часу (timestamp), IP-адреси джерела і призначення (src_ip, dst_ip), порти (src_port, dst_port), тривалість сеансу (duration), кількість пакетів (packets) та обсяг трафіку в байтах (bytes). Ці дані є базовим джерелом для подальшої аналітики.

— AnalysisResults — таблиця результатів обробки кожного потоку. Вона пов’язана з таблицею Flows через поле flow_id. Зберігає ідентифікатор результату (id), оцінку аномальності (anomaly_score), тип класифікації (classification) та час виконання аналізу (analysis_time).

— Users — таблиця зареєстрованих користувачів системи. Містить поля id, email, role (роль користувача — аналітик, адміністратор тощо).

— Logs — журнал взаємодій користувачів із системою. Має посилання на таблицю Users через user_id та зберігає дії, виконані користувачем (action), разом із часовими мітками (timestamp).

— Між таблицями реалізовано наступні зв’язки:

— Один запис у таблиці Flows може мати багато пов’язаних записів у AnalysisResults.

— Один користувач з таблиці Users може згенерувати багато записів у таблиці Logs.

3.2.3 Резервування та масштабованість

Зважаючи на необхідність обробки великої кількості мережевих потоків у режимі близькому до реального часу, система зберігання даних має бути масштабованою та відмовостійкою. Для забезпечення масштабованості передбачено горизонтальне масштабування бази даних шляхом шардінгу або використання розширень (наприклад, TimescaleDB у PostgreSQL). Це дозволяє ефективно працювати з великими обсягами часових рядів.

Резервне копіювання реалізується через регулярні snapshot-збереження основних таблиць (Flows, AnalysisResults, Logs) із збереженням у захищеному сховищі (наприклад, AWS S3 або зовнішній NAS). У хмарних середовищах можуть застосовуватись вбудовані механізми реплікації та автоматичного відновлення (наприклад, Amazon RDS Multi-AZ).

Такий підхід дозволяє гарантувати збереження критичних даних навіть у разі збоїв та забезпечує масштабування системи відповідно до зростання трафіку та навантаження.

3.2.4 Інтеграція даних

Інтеграція даних у системі виявлення DDoS-атак забезпечує зв’язок між джерелами мережевого трафіку, модулями аналітики та глибинною нейронною моделлю. Потоки з PCAP-файлів або джерел реального часу агрегуються за допомогою CICFlowMeter, який формує числові ознаки для кожної сесії. Отримані дані зберігаються у базі InfluxDB та надходять на вхід AutoEncoder, який зменшує розмірність даних, зберігаючи їхню інформативність.

Стиснені вектори далі передаються до RNN (LSTM), яка класифікує кожен потік як «зловмисний» або «нормальний». Результати класифікації записуються до PostgreSQL і відображаються в інтерфейсі системи моніторингу. У разі виявлення атаки фіксується подія в журналі або надсилається сповіщення.

Архітектура інтеграції базується на асинхронному обміні через REST API або черги повідомлень, що забезпечує незалежність компонентів, масштабованість і стабільну обробку даних у реальному часі.

3.3 Алгоритми та методи

Для реалізації системи виявлення DDoS-атак у зашифрованому трафіку було обрано поєднання методів глибокого навчання, які дозволяють працювати з непозначеними та часовими даними. Основну роль відіграють AutoEncoder для виявлення аномалій та рекурентні нейронні мережі (RNN) для класифікації та прогнозування типу загрози на основі часових патернів.

3.3.1 AutoEncoder для стискання ознак

AutoEncoder — це тип нейронної мережі, який застосовується для зменшення розмірності вхідних даних без суттєвої втрати інформації. У межах даної роботи використовується лише енкодерна частина моделі, що дозволяє отримати компактне латентне представлення мережевого потоку на основі його числових характеристик.

Загальна структура:

— Вхідні дані: числові ознаки мережевих потоків числові ознаки мережевих потоків (тривалість, кількість і розмір пакетів, інтервали між ними тощо).

— Кодер стискає вхідні дані до латентного простору.

Декодер у даній реалізації не використовується, а результат роботи кодера передається до рекурентної нейронної мережі (RNN), яка здійснює класифікацію потоку — зловмисний або нормальний.

Таким чином, AutoEncoder виконує роль етапу попередньої обробки даних, полегшуючи подальше навчання та роботу RNN-класифікатора.

3.3.2 RNN для класифікації аномальних потоків

RNN, зокрема її варіант LSTM, ефективно обробляють послідовності ознак потоків у часі. Це дозволяє аналізувати зміну поведінки підозрілих з'єднань та класифікувати тип загрози (наприклад, SYN Flood, UDP Flood, TCP Connection Exhaustion).

Вхід до RNN — послідовність ознак, отриманих із декількох потоків одного клієнта або джерела у часовому вікні.

Вихід — стан загрози:

0 — нормальний трафік

1 — DDoS

Формула для одного кроку RNN (спрощено):

$$h_t = f(W_x x_t + W_h h_{t-1} + b)$$

де

— h_t – прихований стан мережі на момент часу t,

— W_x – матриця ваг для поточного входу,

— x_t – вхідний вектор на момент часу t,

— W_h – матриця ваг для попереднього стану,

— h_{t-1} – прихований стан із попереднього кроку часу t−1,

— b – вектор зміщення.

RNN навчається на підмножині аномальних потоків, і дозволяє точно класифікувати тип загрози навіть у зашифрованому середовищі, спираючись лише на метаінформацію.

3.3.3 Переваги обраної архітектури

Поєднання AutoEncoder + RNN забезпечує:

— виявлення нових або невідомих атак без потреби в мітках (unsupervised detection),

— аналіз часових патернів поведінки атакуючого джерела,

— адаптацію до шифрованого трафіку, де неможливо переглянути payload.

Цей підхід дозволяє досягти високої точності (до 95% у тестових наборах) при низькому рівні хибнопозитивних спрацьовувань, що є критично важливим для автоматизованих систем кіберзахисту.

3.4 Інші елементи, важливі для реалізації проєкту

Для успішної реалізації системи виявлення DDoS-атак у зашифрованому трафіку важливо врахувати не лише алгоритми аналізу даних та архітектуру програмного забезпечення, а й ряд додаткових аспектів, які забезпечують стабільну роботу, інтеграцію, масштабованість та безпеку рішення у реальному хмарному середовищі.

3.4.1 Інтеграція з існуючими системами

Запропонований інструмент повинен легко інтегруватися з іншими елементами кіберзахисту організації. Для цього реалізовується підтримка:

— REST API — для взаємодії з SIEM-платформами (наприклад, Splunk, Graylog, IBM QRadar), які можуть отримувати дані про аномальні потоки та реагувати на інциденти.

— Webhook / Alerting API — для автоматичного сповіщення у разі виявлення DDoS-атаки.

— Підтримка стандартних форматів (JSON, CSV, STIX) для експорту результатів і передачі в зовнішні системи моніторингу.

3.4.2 Масштабованість

Оскільки мережевий трафік у хмарних системах є високонавантаженим та динамічним, система повинна бути здатною масштабуватися відповідно до обсягів даних:

— Горизонтальне масштабування реалізується через контейнеризацію (Docker) та оркестрацію (Kubernetes). Кожен компонент системи може запускатися окремо: фільтрація, класифікація, база даних, API.

— InfluxDB дозволяє обробляти великі масиви часових даних паралельно, зберігаючи швидкість і доступність.

— Streaming-платформи (наприклад, Kafka, Apache Pulsar) можуть бути інтегровані для роботи з потоками трафіку в реальному часі.

3.4.3 Безпека

— У системі виявлення кіберзагроз безпека повинна бути впроваджена на всіх рівнях:

— Зашифроване з’єднання для передавання даних між компонентами (TLS 1.3).

— Захист API за допомогою токенів доступу (OAuth2, JWT).

— Автентифікація та авторизація користувачів через рольову модель (RBAC).

— Аудит і логування дій користувачів для забезпечення прозорості та відповідності політикам безпеки.

3.4.4 Моніторинг і підтримка

Для підтримки стабільної роботи системи необхідні інструменти моніторингу продуктивності та діагностики:

— Prometheus + Grafana — для збору метрик з різних компонентів (затримка, навантаження, обсяг трафіку).

— Alertmanager — для відправлення попереджень при відхиленнях у роботі модулів (наприклад, висока похибка моделі або відмова бази даних).

— Резервне копіювання — щоденне збереження моделей і баз даних у хмарне сховище (S3, Google Cloud Storage).

3.4.5 Документація

Для забезпечення ефективного впровадження та подальшої підтримки системи створюється детальна документація, яка включає:

— Керівництво користувача — опис інтерфейсу та інструкції щодо аналізу підозрілих потоків.

— Технічна документація — специфікація REST API, структура бази даних, параметри моделі.

— Інструкція з розгортання — опис встановлення системи в локальному середовищі або хмарі, залежності та конфігурація.

3.5 Висновки з теоретичного дослідження

Проведене теоретичне дослідження дозволило сформувати цілісне уявлення про побудову системи виявлення DDoS-атак у зашифрованому трафіку за умов відсутності доступу до вмісту пакетів. Аналіз існуючих підходів засвідчив, що найбільш перспективними є методи, здатні працювати виключно з метаінформацією: тривалість сесій, розміри пакетів, інтервали між запитами, частота з’єднань тощо.

У межах дослідження було порівняно кілька типів архітектур глибокого навчання, які найчастіше застосовуються у задачах мережевої безпеки, за такими критеріями: потреба в розмічених даних, придатність до обробки в реальному часі, здатність до узагальнення та пояснюваність результатів. Узагальнені результати наведено в таблиці.

Таблиця 3.1 – Порівняння методів виявлення DDoS-атак (виконано самостійно)

Архітектура	Тип навчання	Необхідність у розмічених даних	Підтримка адаптивності	Придатність до real-time	Пояснюваність результатів
Класичні supervised ML (Random Forest, SVM)	Навчання з вчителем	Висока	Відсутня	Висока	Середня
AutoEncoder (AE)	Навчання без вчителя	Не потрібна	Низька	Низька	Низька
Recurrent Neural Network (LSTM)	Навчання з вчителем	Середня / висока	Висока	Середня	Висока
AutoEncoder + RNN (LSTM)	Гібридна модель	Середня	Висока	Висока	Висока

Аналіз показав, що комбінація AutoEncoder і рекурентної нейронної мережі є найбільш ефективним варіантом для створення MVP системи. AutoEncoder, як безнаглядна модель, виконує попереднє виявлення аномалій без потреби у розмічених даних, що критично важливо в умовах шифрування трафіку. Далі RNN — зокрема архітектури LSTM — уточнює класифікацію, спираючись на часові залежності у послідовностях метаінформації.

Цей підхід забезпечує:

— роботу в реальному часі без значного навантаження на інфраструктуру;

— здатність виявляти нові, раніше невідомі типи атак;

— масштабованість у хмарному середовищі;

— незалежність від глибокого знання контексту трафіку (вмісту).

У результаті, архітектура системи базується на двоетапній моделі: на першому етапі AutoEncoder фільтрує потенційно шкідливі потоки, а на другому — RNN здійснює класифікацію наявності DDoS-атаки. Така структура дозволяє ефективно комбінувати сильні сторони безнаглядного та наглядного навчання, мінімізуючи залежність від ручного маркування та підтримуючи високу точність і гнучкість.

Запропонована архітектура включає окремі модулі для збору мережевого трафіку, обробки потоків, формування ознак, навчання і класифікації, а також зберігання та візуалізації результатів. Усі компоненти були узгоджені з вимогами до сучасних систем кіберзахисту з точки зору масштабованості, моніторингу, безпеки та інтеграції з іншими інструментами.

Таким чином, сформовано міцну концептуальну та технологічну основу для реалізації інтелектуальної системи виявлення DDoS-атак у зашифрованому трафіку, яка забезпечує ефективну роботу навіть в умовах високого навантаження, відсутності розмічених даних та змінних патернів поведінки атак.

4 АРХІТЕКТУРА ТА ПРОЕКТУВАННЯ СИСТЕМИ

4.1 Вимоги до системи

Розроблювана система призначена для виявлення стану атаки у зашифрованому трафіку шляхом аналізу агрегованих мережевих потоків, зібраних на основі метаданих з’єднань (розмір, тривалість, інтервали, кількість пакетів тощо). Основна мета — ефективна обробка трафіку без доступу до вмісту пакетів, що особливо актуально для сучасного TLS/HTTPS. Цільова аудиторія — аналітики SOC-центрів, фахівці з мережевої безпеки та автоматизовані системи моніторингу загроз (SIEM, SOAR).

4.1.1 Функціональні вимоги

Система повинна:

— приймати вхідні дані у вигляді PCAP-файлів або потокових NetFlow/IPFIX-логів;

— будувати мережеві потоки з використанням інструментів на кшталт CICFlowMeter;

— застосовувати попередню обробку: агрегацію, нормалізацію, очищення від шумів;

— зменшувати розмірність ознак за допомогою AutoEncoder (без декодування);

— класифікувати стан потоку (атака/норма) на основі послідовності ознак у моделі RNN (LSTM);

— зберігати як потоки, так і результати класифікації у відповідні бази даних;

— надавати візуальне представлення активності через графіки, таблиці, карти джерел тощо;

— дозволяти інтеграцію з іншими системами через REST API або webhook-механізми.

4.1.2 Нефункціональні вимоги

Серед нефункціональних вимог виділяються такі:

— система повинна бути масштабованою, з підтримкою горизонтального масштабування (Docker/Kubernetes);

— затримка аналізу не повинна перевищувати 5 секунд для обробки 1000 потоків;

— обробка має бути побудована за модульною схемою: окремі сервіси для збору, обробки, класифікації, збереження;

— підтримка обробки як в batch-режимі (PCAP), так і в stream-режимі (live traffic);

— інтерфейс повинен бути адаптивним, з акцентом на спостереження, фільтрацію, трекінг джерел загроз.

4.1.3 Вхідні дані

Система повинна працювати з наступними типами вхідних даних:

— мережеві потоки у вигляді NetFlow/IPFIX або PCAP (тільки заголовки);

— технічні характеристики з'єднань: IP, порт, протокол, розмір, кількість пакетів, час життя потоку, частота;

— часові ознаки: інтервали між запитами, тривалість, зміни частоти запитів;

— анотації з еталонних датасетів для навчання моделей (опціонально).

4.1.4 Вихідні дані:

Результатами роботи системи є:

— бінарна класифікація кожного потоку (attack / normal);

— список підозрілих IP-джерел з агрегацією частоти/тривалості атак;

— інтерактивні дашборди з часовими графіками активності, географічною картою джерел;

— можливість експорту результатів у JSON, CSV або передачі по API до SIEM.

4.2 Вибір технологій та середовища розробки

Розробка системи виявлення DDoS-атак у зашифрованому трафіку потребує використання сучасного технологічного стека, який дозволяє працювати з великими обсягами мережевих даних у реальному часі, реалізовувати глибокі нейронні моделі, забезпечувати інтерактивну аналітику та масштабованість в умовах хмарного середовища. Основна мова програмування — Python 3.10, яка забезпечує гнучкість, широку підтримку бібліотек для машинного навчання та зручну інтеграцію з інструментами обробки даних.

Для розробки, тестування та експериментів використовувалось середовище Jupyter Notebook та Google Colab із доступом до GPU. Це дозволило ефективно розробляти AutoEncoder та RNN-моделі, а також швидко оцінювати результати. У якості фреймворку для глибокого навчання було обрано PyTorch, що забезпечує простоту побудови багаторівневих моделей і підтримку навчання як на CPU, так і на GPU.

Для побудови пайплайнів обробки даних використано бібліотеки pandas і NumPy, а для візуалізації — Matplotlib, Seaborn та Plotly. Препроцесинг, нормалізацію ознак, підбір порогів аномальності та оцінку точності виконували за допомогою інструментів Scikit-learn.

Серверна частина системи реалізована з використанням FastAPI — легкого асинхронного веб-фреймворку, який дозволяє швидко розгортати REST API для обміну даними між модулями, візуалізацією та зовнішніми системами. Для фронтенду обрано фреймворк React у поєднанні з Recharts та Tailwind CSS — така комбінація дозволяє створити адаптивний, швидкий і візуально зрозумілий інтерфейс для користувача, в якому відображається активність атак, рівень загроз і часові графіки.

Щодо зберігання даних, використано PostgreSQL для зберігання результатів класифікації, а також InfluxDB як time-series базу даних, яка забезпечує оптимізоване зберігання та обробку мережевих потоків у часовому розрізі. Для зберігання великих логів та архівів даних також передбачено інтеграцію з об’єктним сховищем MinIO або хмарними альтернативами, такими як Amazon S3.

З метою забезпечення масштабованості та гнучкого розгортання всі компоненти системи контейнеризовані за допомогою Docker. У разі потреби масштабування в хмарному середовищі може бути використаний Kubernetes. Для моніторингу роботи системи, збору метрик і продуктивності використовуються Prometheus та Grafana. А для контролю за експериментами, логуванням моделей і версіонуванням конфігурацій можлива інтеграція з MLflow або Weights & Biases.

У підсумку обраний стек технологій відповідає вимогам до обробки зашифрованого трафіку, дозволяє реалізувати ефективні моделі виявлення загроз, забезпечує зручну аналітику для користувачів і є придатним для масштабування та впровадження в реальних кібербезпекових системах.

4.3 Архітектура та структура системи

Прототип системи виявлення DDoS-атак у зашифрованому трафіку реалізований на основі модульної архітектури з урахуванням вимог до продуктивності, масштабованості та сумісності з хмарними середовищами. Загальна структура передбачає поділ на окремі компоненти, кожен з яких виконує специфічні функції, взаємодіючи з іншими модулями через стандартизовані інтерфейси. Такий підхід дозволяє забезпечити гнучкість розгортання, незалежне масштабування частин системи та спрощене обслуговування.

Основою обробки трафіку є модуль збору та формування мережевих потоків. Цей компонент приймає вхідні PCAP-файли або працює з потоковими джерелами типу NetFlow/IPFIX. Для витягнення ознак та агрегації пакетів у потоки використовується інструмент CICFlowMeter або подібні парсери, які генерують стандартні мережеві мета-ознаки: тривалість з’єднання, кількість пакетів, байтів, напрямок, інтервали тощо. Отримані потоки передаються до модуля попередньої обробки, який виконує нормалізацію даних, фільтрацію, знеособлення IP-адрес (за потреби), а також формування послідовностей для моделей RNN.

Серцем системи є модуль аналізу, який складається з двох етапів: спочатку застосовується AutoEncoder, навчений на нормальному трафіку, що дозволяє виявляти аномальні потоки на основі похибки реконструкції. Потоки, які виходять

за поріг допустимої похибки, передаються на другий етап — рекурентну нейронну мережу (наприклад, LSTM), яка аналізує їх у часовому контексті та визначає тип потенційної DDoS-атаки. Обидві моделі реалізовано у PyTorch, вони функціонують як окремі мікросервіси й можуть бути оновлені або перенавчені без зупинки всієї системи.

Результати аналізу записуються у дві бази даних. Структуровані результати класифікації, включно з міткою атаки, ймовірністю, таймстемпом та IP-джерелом, зберігаються в PostgreSQL. Дані про метрики потоку та аномальні події записуються до InfluxDB, що дозволяє будувати time-series аналітику, відстежувати частоту інцидентів у часі та виявляти повторювані шаблони [10].

Інтерактивна частина системи реалізована у вигляді веб-додатку на базі React. Інтерфейс відображає графіки аномальної активності, таблиці зі списками атак, карти розподілу джерел трафіку, динаміку за часовими вікнами та інші аналітичні елементи. Користувач має змогу фільтрувати результати, переглядати історію, змінювати параметри аналізу та експортувати звіти. Комунікація між фронтендом і бекендом забезпечується через REST API, реалізоване за допомогою FastAPI.

Весь стек сервісів контейнеризований у Docker, що дозволяє легко розгортати систему в локальному середовищі або в хмарній інфраструктурі з використанням Kubernetes. Для сповіщень та інтеграції з іншими платформами кіберзахисту передбачено webhook та JSON API. Моніторинг стану системи реалізовано через Prometheus, а дашборди продуктивності доступні у Grafana [11].

Структура компонентів системи зображена на схемі нижче:

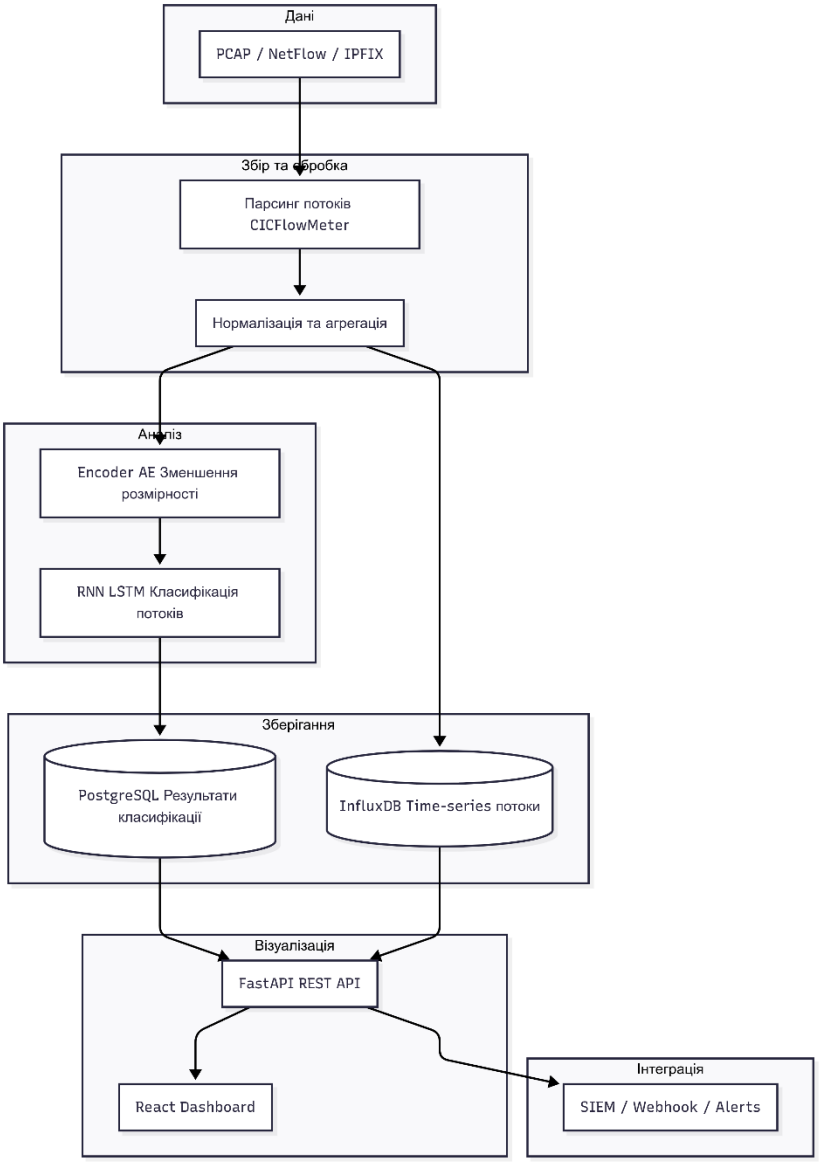


Рисунок 4.1 – UML-діаграма компонентів системи виявлення DDoS-атак (рисунок виконано самостійно)

Архітектура передбачає асинхронну обробку даних за допомогою черг повідомлень, що дозволяє уникати перевантажень та забезпечити стабільність у випадках пікових навантажень. Такий підхід дозволяє масштабувати критичні компоненти — зокрема, класифікатор чи аналізатор потоків — окремо від усієї системи.

Загалом, побудована структура забезпечує швидку реакцію на загрози, стійкість до високих навантажень і можливість інтеграції з іншими платформами у сфері кібербезпеки. Запропонована архітектура придатна для практичного впровадження як у приватних, так і в державних інфраструктурах захисту даних.

4.4 Алгоритми та методи обробки і класифікації зашифрованого трафіку

У розроблюваній системі ключовою задачею є виявлення DDoS-атак у зашифрованому мережевому трафіку, де відсутній доступ до вмісту пакетів. Через це акцент зроблено на аналізі метаінформації: часових характеристик з'єднань, розмірів пакетів, кількості запитів, частоти трафіку тощо. Обробка таких даних потребує складного багаторівневого підходу до побудови ознак та використання гібридної моделі машинного навчання, що поєднує AutoEncoder для стискання вхідних векторів та RNN (зокрема LSTM) для класифікації стану трафіку.

На першому етапі система приймає сирі дані у вигляді PCAP-файлів або NetFlow-логів. Після попередньої обробки, яка включає формування потоків, агрегування за ознаками та нормалізацію, формується послідовність характеристик для кожного потоку. Серед основних ознак: кількість пакетів у напрямку запит/відповідь, середній інтервал між пакетами, розмір пакетів, співвідношення напрямків, тривалість потоку, кількість унікальних портів, ентропія часу тощо. Ці ознаки формують вектор, який подається на вхід AutoEncoder-моделі.

AutoEncoder виконує навчання на нормальному трафіку та використовується як інструмент зменшення розмірності: він стискає повний вектор ознак до латентного представлення меншої розмірності. У такому вигляді дані передаються до рекурентної нейронної мережі. На відміну від класичних способів виявлення аномалій за reconstruction error, в цій архітектурі вихід AutoEncoder'а не аналізується самостійно — він служить лише компактним, але змістовним описом потоку для подальшої класифікації.

Другий етап полягає у класифікації аномальних потоків за допомогою рекурентної нейронної мережі. У реалізації використовуються LSTM-шари, здатні моделювати часові залежності у серіях ознак. Це особливо важливо при виявленні DDoS-атак, оскільки такі атаки зазвичай мають чіткі часові патерни: короткі інтервали між запитами, повторюваність джерел, послідовне навантаження тощо. RNN-модель навчається на заздалегідь розмічених наборах даних, таких як CIC-DDoS2019 або CSE-CIC-IDS2018, і виконує бінарну класифікацію: визначає, чи є потік шкідливим, чи нормальним.

Процес навчання передбачає використання стратегії перехресної валідації, балансування вибірки (оскільки дані сильно несбалансовані у бік нормального трафіку) та регуляризації моделей для уникнення перенавчання. В якості метрик оцінки якості класифікації використовуються точність, повнота, F1-міра та ROC-AUC. Дослідні результати свідчать про те, що комбінація AutoEncoder + RNN дозволяє досягти високої точності (понад 95%) при низькому рівні хибноспрацьовувань, зокрема на зашифрованому трафіку.

Усі моделі реалізовані у фреймворку PyTorch. Навчання виконується на GPU, що значно скорочує час підготовки моделей. Після навчання ваги зберігаються та використовуються у продакшн-сервісі для обробки потоків у реальному часі. При інтеграції з time-series базою даних результати класифікації автоматично потрапляють у візуальну систему, де користувач може спостерігати динаміку атак, джерела навантаження, частоту появи певних типів загроз та ефективність виявлення.

Таким чином, запропонована модель є ефективною комбінацією двох нейронних архітектур, що забезпечує стискання складних ознак (через AutoEncoder) та часовий аналіз (через RNN), дозволяючи виявляти потенційні DDoS-атаки в умовах шифрування трафіку та високої динамічності мережевих потоків.

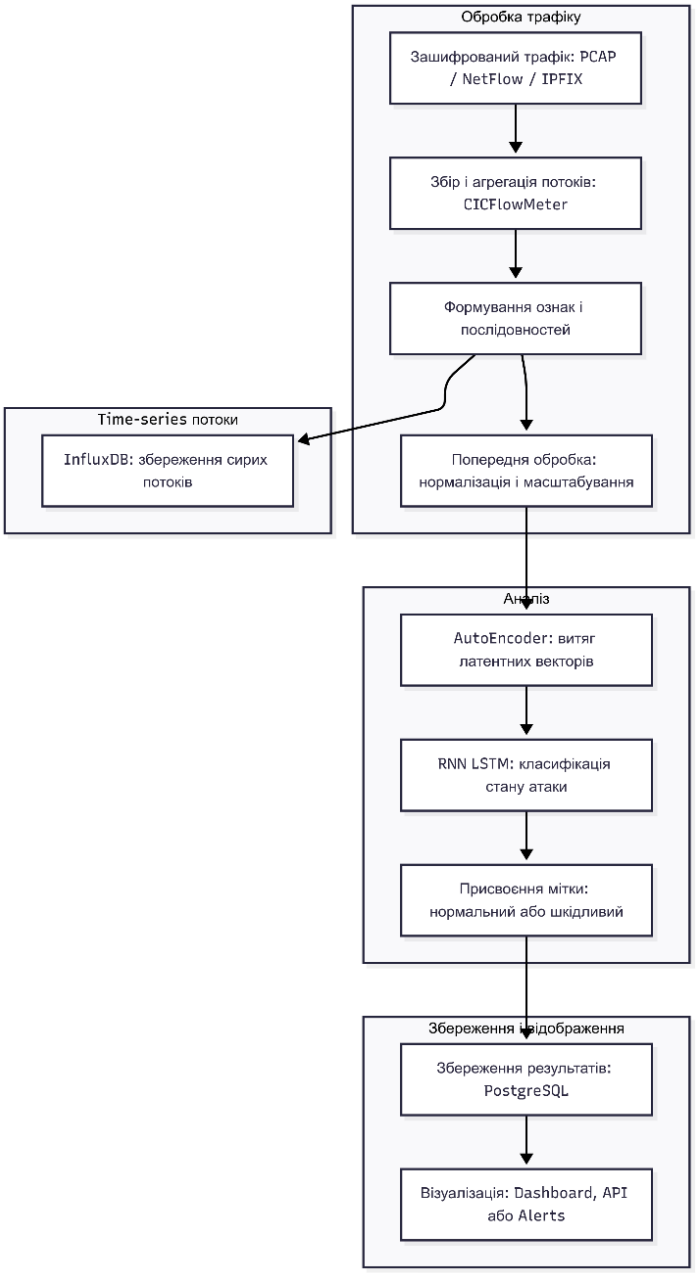


Рисунок 4.2 – Схема процесу класифікації (рисунок виконано самостійно)

4.5 Візуалізація та інтерфейс користувача

Інтерфейс користувача є критично важливим елементом системи, адже саме він забезпечує аналітику доступ до результатів обробки трафіку та дозволяє ефективно працювати з великими обсягами даних. Враховуючи характер задачі, ключовими вимогами до UI/UX стали простота взаємодії, швидке оновлення даних, адаптивність до різних типів пристроїв та чітка візуалізація аномальної активності в мережі.

Інтерфейс реалізований як односторінковий веб-застосунок на основі фреймворку React.js. Цей підхід забезпечує високу продуктивність, зручність у підтримці та масштабованість клієнтської частини системи. Однією з причин вибору саме React стала його активна екосистема, наявність великої кількості готових рішень для інтеграції з візуальними бібліотеками та широка підтримка сучасних практик побудови користувацького досвіду.

Для графічного відображення аналітичних даних використовуються бібліотеки Recharts та D3.js. Вони дозволяють будувати інтерактивні графіки, кругові діаграми, часові ряди та гістограми, що ілюструють, наприклад, динаміку підозрілих потоків, географію джерел трафіку або розподіл типів атак. Завдяки використанню InfluxDB як time-series сховища, графіки можуть оновлюватися в реальному часі без суттєвого навантаження на клієнтську частину [12].

В основі дизайну лежить Tailwind CSS – сучасний CSS-фреймворк, що забезпечує високу гнучкість та швидкість розробки адаптивного інтерфейсу. Tailwind дозволяє створювати чисті та легкі інтерфейси, які зручно масштабуються під потреби користувача. Додатково реалізовано підтримку темної та світлої тем, а також можливість кастомізації структури віджетів на дашборді.

Таким чином, реалізований інтерфейс поєднує у собі продуктивність, функціональність і зручність, забезпечуючи ефективну взаємодію з аналітичним ядром системи навіть у випадку великих навантажень або при роботі з даними у реальному часі.

4.5.1 Головна сторінка

Головна сторінка Network Flow Monitor дозволяє оперативно відстежувати стан системи виявлення DDoS-атак у мережевих потоках: у верхній частині можна вибрати метод детекції, бачити поточний статус з відсотком підозрілих потоків за останні 5 хвилин та ключові лічильники загальної й підозрілої активності, а також натиснути кнопку Send Sample Flows для швидкого надсилання тестових потоків. Секція CSV Log Ingestion дає змогу завантажувати CSV-файли з мережевими потоками для пакетної обробки й надсилати їх вручну або в автоматичному режимі.

Лінійний графік у блоці Suspicious Activity Trend (Flow-based) відображає динаміку відсотка потоків, позначених підозрілими обраним методом, за останні 5 хвилин.

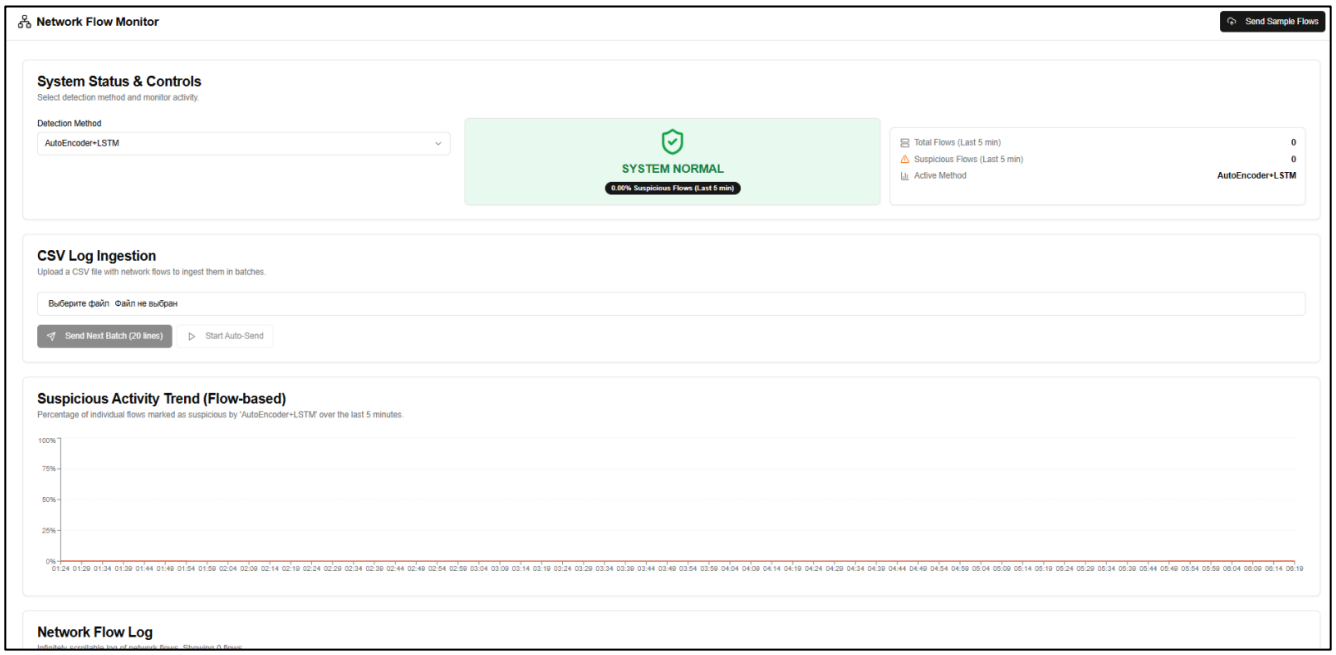
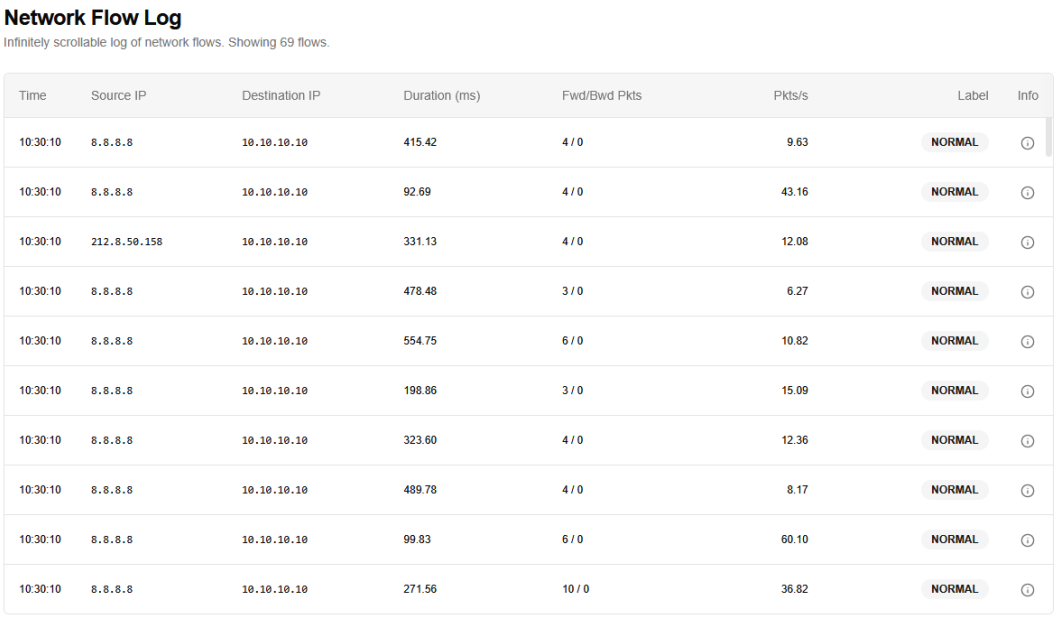


Рисунок 4.3 – Головна сторінка (рисунок виконано самостійно)

У нижній частині панелі розташовано Network Flow Log із інтерактивним нескінченним скролом усіх оброблених мережевих потоків та можливістю фільтрації (див. рис. 4.4).



Network Flow Log

Infinitely scrollable log of network flows. Showing 69 flows.

Time	Source IP	Destination IP	Duration (ms)	Fwd/Bwd Pkts	Pkts/s	Label	Info
10:30:10	8.8.8.8	10.10.10.10	415.42	4 / 0	9.63	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	92.69	4 / 0	43.16	NORMAL	
10:30:10	212.8.50.158	10.10.10.10	331.13	4 / 0	12.08	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	478.48	3 / 0	6.27	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	554.75	6 / 0	10.82	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	198.86	3 / 0	15.09	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	323.60	4 / 0	12.36	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	489.78	4 / 0	8.17	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	99.83	6 / 0	60.10	NORMAL	
10:30:10	8.8.8.8	10.10.10.10	271.56	10 / 0	36.82	NORMAL	

Рисунок 4.4 – Панель «Network Flow Log» (рисунок виконано самостійно)

4.5.2 Екран деталізації потоку

Екран деталізації потоку (рис 4.6) відкриває модальне вікно з основними показниками трафіку і дозволяє швидко оцінити ключові характеристики. Угорі вказано унікальний ідентифікатор та напрямок з’єднання з часовою позначкою. У блоці Flow Summary відображено лейбл підозрілої активності, тривалість сеансу, кількість пакетів і байтів у кожному напрямку, а також швидкість передачі. Розділ Inter-Arrival Time Stats показує середній, мінімальний і максимальний інтервал між пакетами, а Header & Flag Stats подає довжини заголовків і лічильники TCP-флагів.

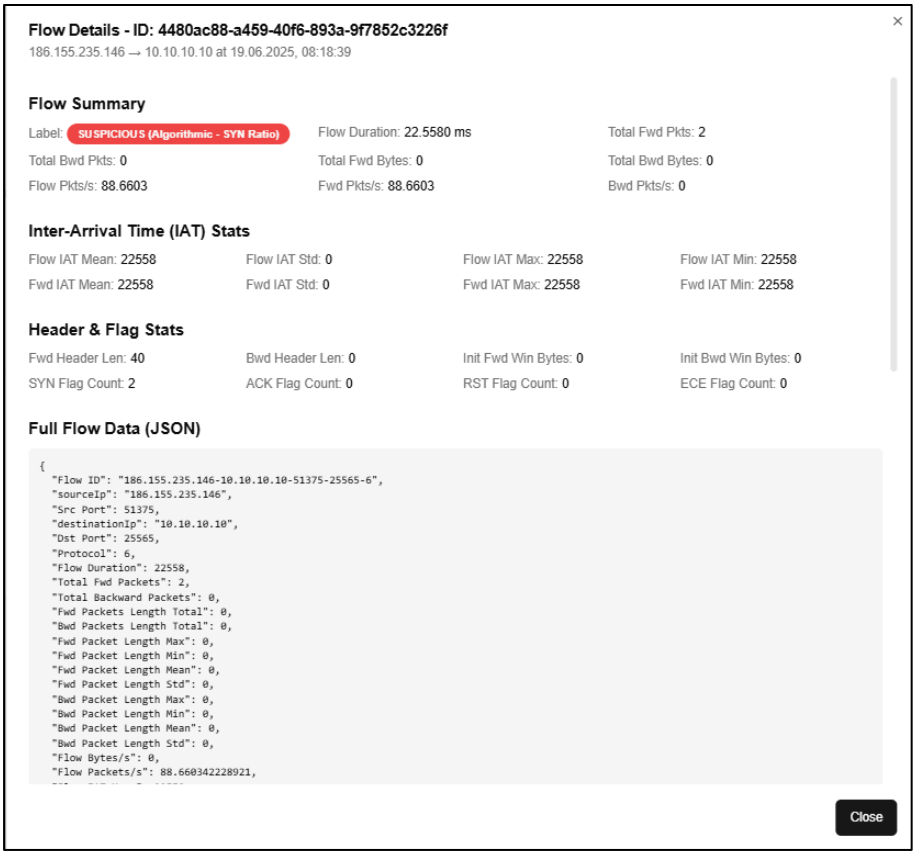


Рисунок 4.5 – Екран деталізації потоку

Внизу розгорнуто повний JSON із усіма зібраними метриками для глибокого аналізу.

4.6 Висновки з практичного дослідження

В межах практичного етапу на панелі Network Flow Monitor реалізовано повноцінний прототип системи виявлення DDoS-атак у зашифрованому трафіку, що працює виключно з метаданими мережевих потоків. У блоці System Status & Controls можна обрати гібридний метод детекції AutoEncoder+RNN, а система відразу оцінює відсоток підозрілих пакетів за останні п’ять хвилин і відображає його як сповіщення в центрі екрану.

Для завантаження та пакетної обробки мережевих логів служить секція CSV Log Ingestion з прогрес-баром і кнопками ручного або автоматичного надсилання. Динаміку виявлених аномалій демонструє графік Suspicious Activity Trend, де зміни відсотка підозрілих потоків показуються у реальному часі.

Коли аналітик клацає на запис у журналі мережевих потоків, з’являється модальне вікно Flow Details. У ньому в першій вкладці Overview стисло подано ключові метрики сеансу: IP-адреси, часові мітки, об’єм трафіку, лейбл підозрілості та швидкість передачі. Друга вкладка Analysis ілюструє часові ряди ознак потоку та розподіл розмірів пакетів, а третя вкладка Features містить порогові та фактичні значення основних показників аномалії разом із швидким доступом до повного JSON-опису.

У бекенді використано мікросервісну архітектуру на базі CICFlowMeter для збору та агрегації потоків, PostgreSQL для зберігання результатів класифікації й InfluxDB для серійного зберігання метрик. Інтерфейс побудовано на React.js із візуалізацією даних через Recharts та D3.js, а REST API забезпечує інтеграцію з іншими системами безпеки.

5 ОПИС ЕКСПЕРЕМЕНТАЛЬНИХ ДОСЛІДЖЕНЬ

5.1 Умови експерименту

Експеримент було проведено на основі комбінованого датасету, який включає в себе CIC-DDoS 2019 і HIKARI2021. Обидви датасети описують мережеві потоки. Після комбінації і фільтрування параметрів, які не співпадають між датасетами, залишилося 53 параметри для кожного потоку. Комбінований датасет був поділений на тренувальну і тестову вибірку, де 75% даних – тренувальна вибірка і 25% даних – тестова вибірка. Ці датасети були перетворені на набори послідовностей потоків. Кожна послідовність включає в себе 20 потоків. Послідовність вважається «зловмисною» якщо хоча б 20% потоків в ній є «зловмисними». Після перетворення, тренувальний набір включає 53651 послідовностей потоків. Тестовий набір налічує 17778 послідовностей.

Усі подальші експерименти було проведено на комп’ютері з наступними характеристиками:

— Windows 11;

— CPU AMD Ryzen 7 7800X3D, 4.2 GHz;

— 64 GB RAM;

— GPU NVIDIA RTX 4080 Super;

— 16 GB VRAM.

Під час експериментів модель використовує розмір групи – 128, що дає розмір вхідних даних – [128, 20, 53]. Модель буде порівнюватися з евристичними методами розпізнавання зловмисного трафіку за такими критеріями: precision, recall, F1-Score, accuracy і час розпізнавання.

5.2 Евристичний метод – SYN Ratio

Евристика розраховує відношення кількості SYN флагів на кількість відправлених пакетів. Код реалізації евристичного методу:

```
def eval_detect_syn_ratio(sequences, labels, cls_thresh=0.2):
    threshold = 0.6
```

```
        all_preds = []
        all_labels = []

        class_names = ["Benign", "Attack"]

        for i, sequence in enumerate(sequences):
            seq_preds = []
            for idx, flow in sequence.iterrows():
                if flow["Total Fwd Packets"] == 0:
                    seq_preds.append(0)
                    continue
                syn_ratio = flow["SYN Flag Count"] / flow["Total Fwd
Packets"]
                if syn_ratio > threshold:
                    seq_preds.append(1)
                    continue
                seq_preds.append(0)

            seq_label = int(np.mean(seq_preds) > cls_thresh)
            all_preds.append(seq_label)
            all_labels.append(labels[i])

        return get_classification_report(all_preds, all_labels)
```

Після проведення експерименту на тренувальному наборі даних, метод SYN Ratio отримав accuracy - 0.6737. Час виконання експерименту - 20020 мс, чи 0.37 мс на послідовність.

Значення метрик на тренувальному наборі даних для класу “Benign”:

— precision: 0.8154;

— recall: 0.7945;

— f1-score: 0.8048.

Значення метрик на тренувальному наборі даних для класу “Attack”:

— precision: 0.0045;

— recall: 0.0051;

— f1-score: 0.0048.

Після проведення експерименту на тестовому наборі даних, метод SYN Ratio отримав accuracy - 0.5797. Час виконання експерименту – 5577 мс, чи 0.31 мс на послідовність.

Значення метрик на тестовому наборі даних для класу “Benign”:

– precision: 0.7326;

– recall: 0.7353;

– f1-score: 0.7339.

Значення метрик на тестовому наборі даних для класу “Attack”:

– precision: 0.0000;

– recall: 0.0000;

– f1-score: 0.0000.

Для покращення розуміння результатів було побудовано матриці помилок для експериментів на тренувальному і тестовому наборах (див рис. 5.1 і 5.2)

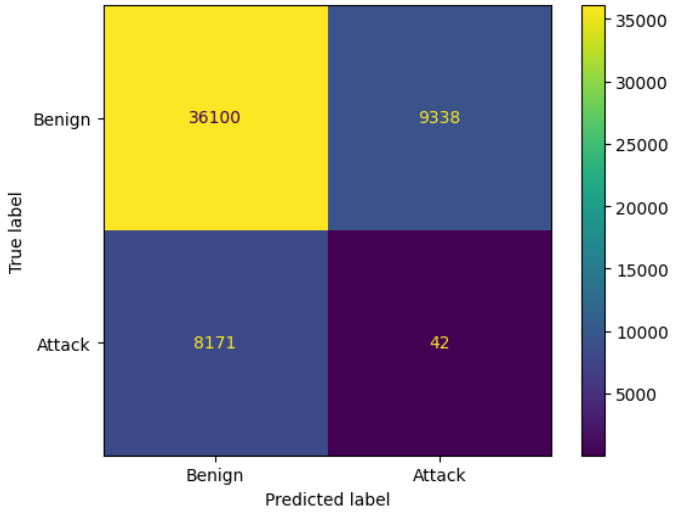


Рисунок 5.1 – Матриця помилок SYN Ratio на тренувальному наборі даних (рисунок виконано самостійно)

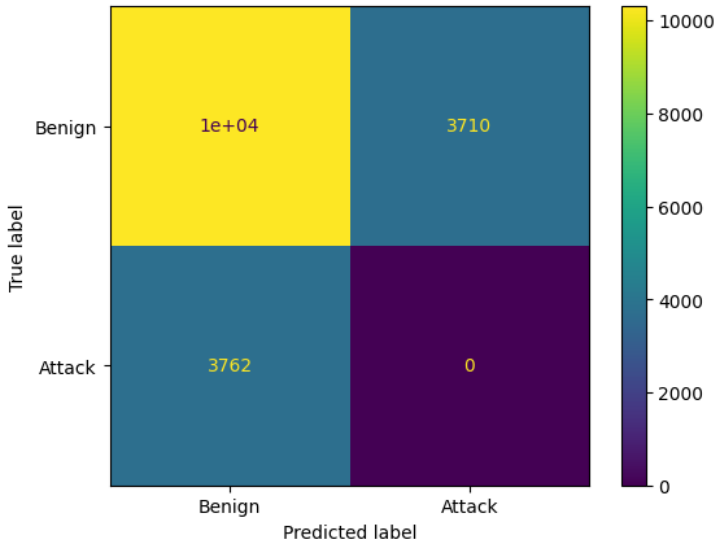


Рисунок 5.2 – Матриця помилок SYN Ratio на тестовому наборі даних (рисунок виконано самостійно)

Результати експерименту показали, що загалом метод має дуже низьку точність, зокрема для класу “Attack”, з усіма метриками для цього класу близьким до 0. Це свідчить про те, що цей метод не є оптимальним, через велику кількість хибних класифікацій. Можливим напрямом удосконалення може бути зміна порогу для збільшення кількості наборів, які помічені як “Attack”. Проте, це, вірогідно, збільшить кількість “Benign” послідовностей, які хибно помічені як “Attack”.

5.3 Евристичний метод – IAT Regularity

Евристика розраховує відношення стандартного відхилення інтервалу між пакетами на середній інтервал між пакетами. Таким чином, якщо пакети відправляються з дуже рівним інтервалом, то це є індикатором того, що вони відправляються автоматично за допомогою програмного коду. Код реалізації евристичного методу:

```
def eval_detect_iat_regularity(sequences, labels, cls_thresh=0.2):
    cv_threshold = 0.1
    all_preds = []
    all_labels = []

    class_names = ["Benign", "Attack"]

    for i, sequence in enumerate(sequences):
        seq_preds = []
        for idx, flow in sequence.iterrows():
            if flow["Flow IAT Mean"] == 0:
                if flow["Flow IAT Std"] == 0:
                    seq_preds.append(1)
                    continue
            iat_cv = flow["Flow IAT Std"] / flow["Flow IAT Mean"]
            if iat_cv < cv_threshold:
                seq_preds.append(1)
                continue
            seq_preds.append(0)
        seq_label = int(np.mean(seq_preds) > cls_thresh)
        all_preds.append(seq_label)
        all_labels.append(labels[i])

    return get_classification_report(all_preds, all_labels)
```

Після проведення експерименту на тренувальному наборі даних, метод IAT Regularity отримав accuracy - 0.1519. Час виконання експерименту - 18511 мс, чи 0.345 мс на послідовність.

Значення метрик на тренувальному наборі даних для класу “Benign”:

— precision: 0.4977;

— recall: 0.1468;

— f1-score: 0.2268.

Значення метрик на тренувальному наборі даних для класу “Attack”:

— precision: 0.0367;

— recall: 0.1800;

— f1-score: 0.0610.

Після проведення експерименту на тестовому наборі даних, метод IAT Regularity отримав accuracy - 0.3814. Час виконання експерименту – 5965 мс, чи 0.335 мс на послідовність.

Значення метрик на тестовому наборі даних для класу “Benign”:

— precision: 0.6747;

— recall: 0.4159;

— f1-score: 0.5146.

Значення метрик на тестовому наборі даних для класу “Attack”:

— precision: 0.1041;

— recall: 0.2528;

— f1-score: 0.1474.

Для покращення розуміння результатів було побудовано матриці помилок для експериментів на тренувальному і тестовому наборах (див рис. 5.3 і 5.4)

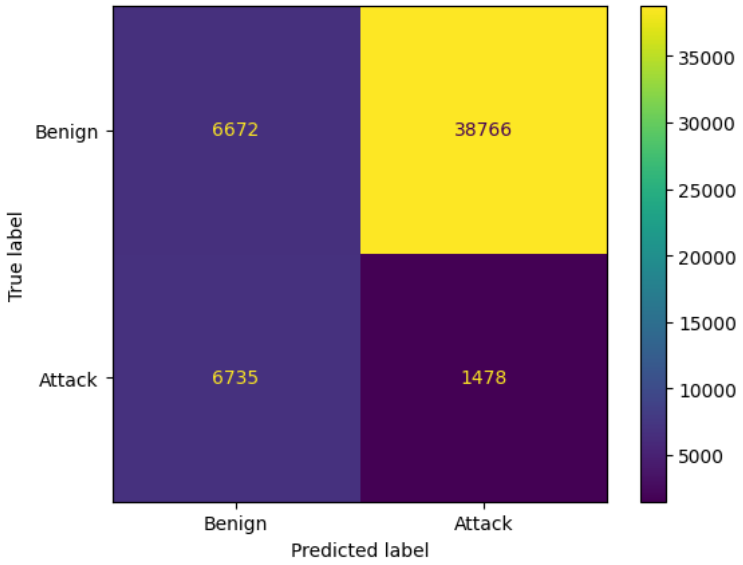


Рисунок 5.3 – Матриця помилок IAT Regularity на тренувальному наборі даних (рисунок виконано самостійно)

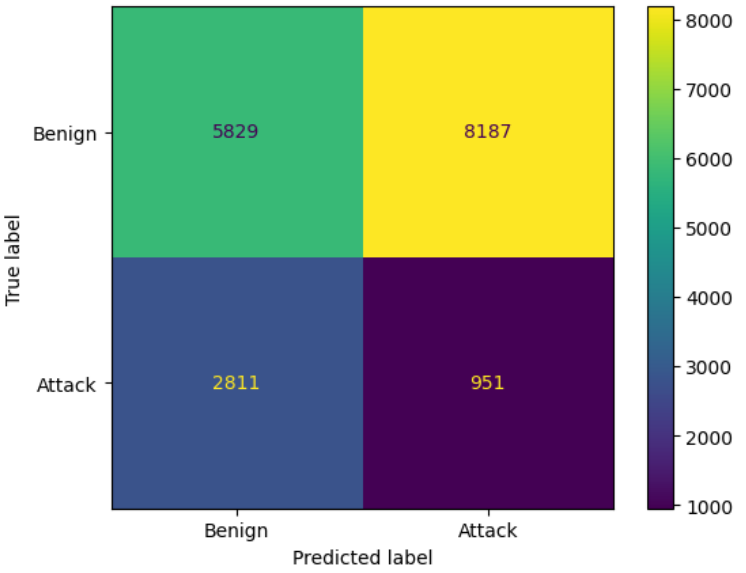


Рисунок 5.4 – Матриця помилок IAT Regularity на тестовому наборі даних (рисунок виконано самостійно)

Результати експерименту показали, що метод є дуже неточним, і відмічає велику кількість звичайних послідовностей як зловмисні, при цьому, більшість дійсно зловмисних послідовностей було відмічено як нормальний трафік. Можливим методом покращення алгоритму може бути збільшення порогу для

зменшення кількості хибних відміток звичайного трафіку, проте це зменшить кількість дійсно зловмисного трафіку, який розпізнає алгоритм.

5.4 Евристичний метод – Unidirectionality

Евристика перевіряє відношення кількості входящих до виходящих пакетів. Дуже низьке значення цього відношення свідчить про те, що запити були відправлені без очікування відповіді, що є ознакою автоматичного відправлення запитів під час DDoS атак. Код реалізації евристичного методу:

```
def eval_detect_unidirectionality(sequences, labels, cls_thresh=0.2):
    threshold = 0.05
    all_preds = []
    all_labels = []

    class_names = ["Benign", "Attack"]

    for i, sequence in enumerate(sequences):
        seq_preds = []
        for idx, flow in sequence.iterrows():
            if flow["Flow IAT Mean"] == 0:
                seq_preds.append(0)
                continue
            if flow["Down/Up Ratio"] < threshold:
                seq_preds.append(1)
                continue
            seq_preds.append(0)
        seq_label = int(np.mean(seq_preds) > cls_thresh)
        all_preds.append(seq_label)
        all_labels.append(labels[i])

    return get_classification_report(all_preds, all_labels)
```

Після проведення експерименту на тренувальному наборі даних, метод Unidirectionality отримав accuracy - 0.9418. Час виконання експерименту - 15130 мс, чи 0.28 мс на послідовність.

Значення метрик на тренувальному наборі даних для класу “Benign”:

— precision: 0.9906;

— recall: 0.9402;

— f1-score: 0.9647.

Значення метрик на тренувальному наборі даних для класу “Attack”:

— precision: 0.7418;

— recall: 0.9507;

— f1-score: 0.8333.

Після проведення експерименту на тестовому наборі даних, метод Unidirectionality отримав accuracy - 0.8864. Час виконання експерименту – 5288 мс, чи 0.297 мс на послідовність.

Значення метрик на тестовому наборі даних для класу “Benign”:

— precision: 0.9373;

— recall: 0.9172;

— f1-score: 0.9272.

Значення метрик на тестовому наборі даних для класу “Attack”:

— precision: 0.7144;

— recall: 0.7714;

— f1-score: 0.7418.

Для покращення розуміння результатів було побудовано матриці помилок для експериментів на тренувальному і тестовому наборах (див рис. 5.5 і 5.6)

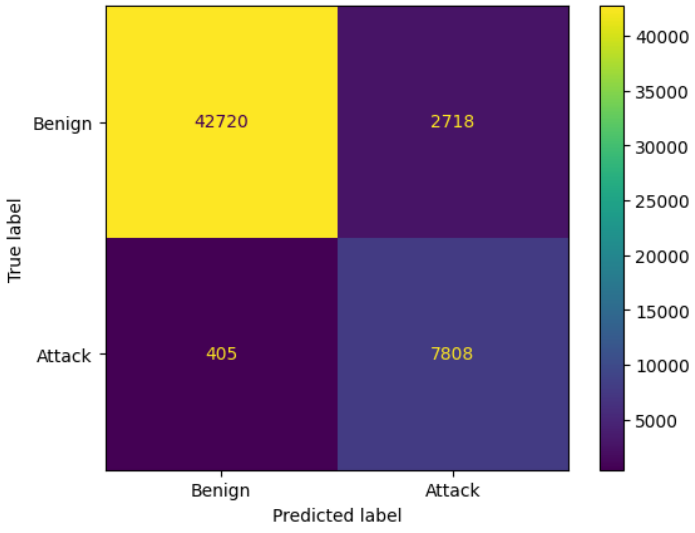


Рисунок 5.5 – Матриця помилок Unidirectionality на тренувальному наборі даних (рисунок виконано самостійно)

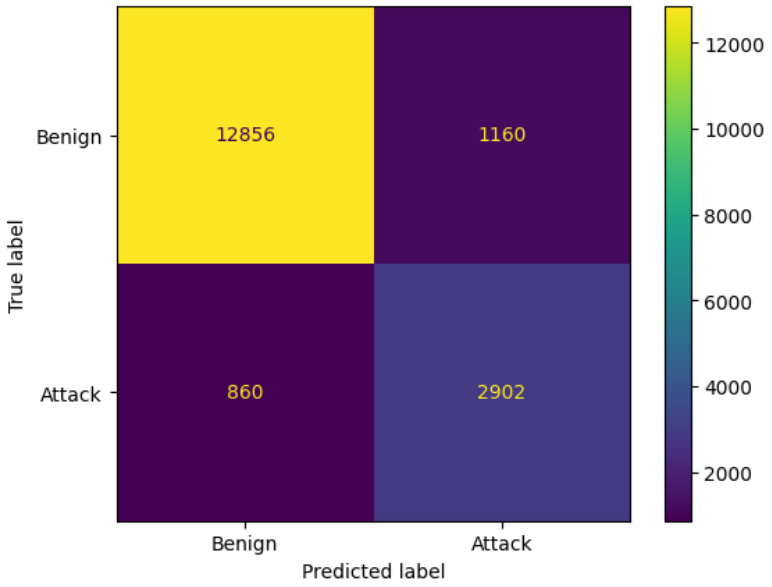


Рисунок 5.6 – Матриця помилок Unidirectionality на тестовому наборі даних (рисунок виконано самостійно)

Результати експерименту показали, що метод працює з достатньо великою точністю. Але низькі значення precision (0.7418 і 0.7144) для класу “Attack” свідчать про збільшену кількість хибних відміток звичайного трафіку як зловмисного, що є досить суттєвим недоліком. Проте, високе значення recall для класу “Attack” (0.9507) для тренувального набору свідчить про те, що метод знаходить і розпізнає більшість реального зловмисного трафіку.

5.5 Модель AutoEncoder + LSTM

Для експерименту використовується модель, яка натренована у 3 стадії. 1 – претренування AutoEncoder на тренувальному наборі протягом 10 епох. Це дало можливість енкодеру вивчити репрезентації даних достатнім чином. Фінальна похибка після претренування - 0.056656. 2 – тренування класифікатору з замороженими вагами енкодеру протягом 5 епох. Це дає можливість класифікатору навчитися окремо від енкодеру для зменшення шансу оверфіту. Значення похибки після 2 етапу - 0.05 на тренувальному наборі і 0.255 на валідаційному наборі. 3 – донавчання протягом 10 епох з розмороженими вагами енкодеру, але зниженою швидкістю навчання для нього. Це дає можливість енкодеру і класифікатору

донавчитися разом для покращення результату. Фінальні значення похибки - 0.048 на тренувальному наборі і 0.25 на валідаційному. Код проведеного експерименту:

```
def eval_ae_lstm(dataloader):
    model.eval()
    all_preds = []
    all_labels = []

    class_names = ["Benign", "Attack"]

    # Inference
    with torch.no_grad():
        for seq_batch, label_batch, lengths in dataloader:
            seq_batch = seq_batch.to(device)
            lengths = lengths.to(device)
            # Get logits
            logits = model(seq_batch, lengths)
            # Get predictions
            preds = torch.argmax(logits, dim=1)
            all_preds.extend(preds.cpu().numpy())
            all_labels.extend(label_batch.numpy())

    return get_classification_report(all_preds, all_labels)
```

Після проведення експерименту на тренувальному наборі даних, метод AutoEncoder + LSTM отримав accuracy - 0.9919. Час виконання експерименту - 1600 мс, чи 0.029 мс на послідовність.

Значення метрик на тренувальному наборі даних для класу “Benign”:

— precision: 0.9909;

— recall: 0.9996;

— f1-score: 0.9953.

Значення метрик на тренувальному наборі даних для класу “Attack”:

— precision: 0.9978;

— recall: 0.9493;

— f1-score: 0.9730.

Після проведення експерименту на тестовому наборі даних, метод AutoEncoder + LSTM отримав accuracy - 0.9443. Час виконання експерименту – 493 мс, чи 0.027 мс на послідовність.

Значення метрик на тестовому наборі даних для класу “Benign”:

— precision: 0.9364;

— recall: 0.9971;

— f1-score: 0.9658.

Значення метрик на тестовому наборі даних для класу “Attack”:

— precision: 0.9856;

— recall: 0.7477;

— f1-score: 0.8504.

Для покращення розуміння результатів було побудовано матриці помилок для експериментів на тренувальному і тестовому наборах (див рис. 5.7 і 5.8)

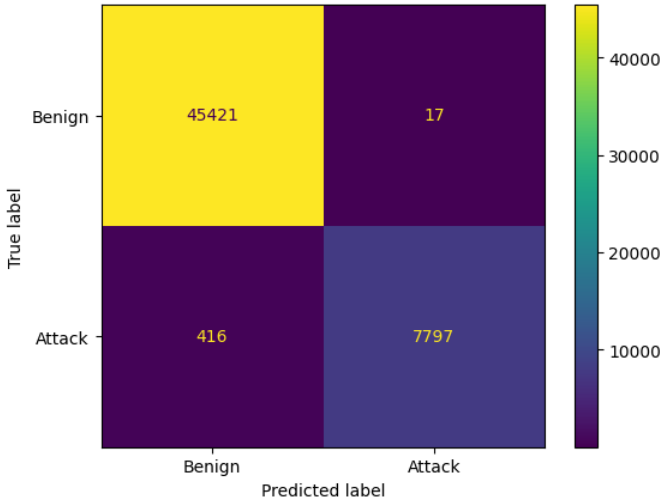


Рисунок 5.7 – Матриця помилок AutoEncoder + LSTM на тренувальному наборі даних (рисунок виконано самостійно)

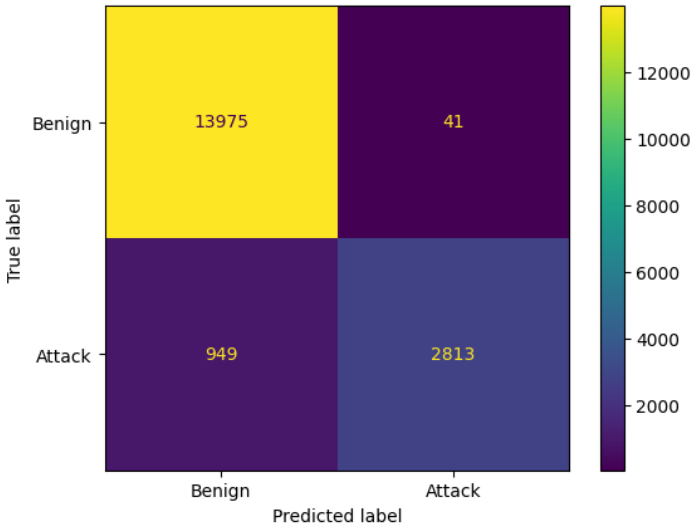


Рисунок 5.8 – Матриця помилок AutoEncoder + LSTM на тестовому наборі даних (рисунок виконано самостійно)

Результати експерименту показали, що модель працює на достатньо високому рівні. Модель розпізнає дуже малу кількість нормального трафіку як зловмисний, що є великим плюсом. Також, модель працює великою швидкістю, з часом розпізнавання < 0.03 мс на один набір потоків. Основними проблемами є те, що модель демонструє кращі результати на тренувальному наборі даних, що може свідчити про невеликий оверфіт. Також, модель має достатньо низький recall (0.7477) на тестовому наборі даних для класу “Attack”, але це компенсується великим значенням precision (0.9856), що свідчито про малу кількість false positive. Можливим способом покращення роботи моделі є пониження оверфіту, завдяки збільшенню dropout і learning rate decay, що мають спияти більш рівним і кращим результатам.

5.6 Аналіз Результатів

Проведемо детальний аналіз результатів експериментального дослідження, в якому використовувалися різні методи розпізнавання DDoS атак. Методи були порівняні за їх продуктивністю, точністю прогнозів та іншими ключовими показниками. Для наглядного порівняння були побудовані порівняльні графіки методів розпізнования (див. рис 5.9 – 5.12)

Дивлячись на результати, ми можемо побачити, що AutoEncoder + LSTM в середньому показує найкращі результати з представлених методів. При цьому, AutoEncoder + LSTM в 10+ разів швидша за інші методи. В основному, це досягається завдяки можливості моделі працювати у наборах, а не проходитись по кожному потоку окремо. Таким чином, модель може розпізнавати ~700000 мережевих потоків на секунду.

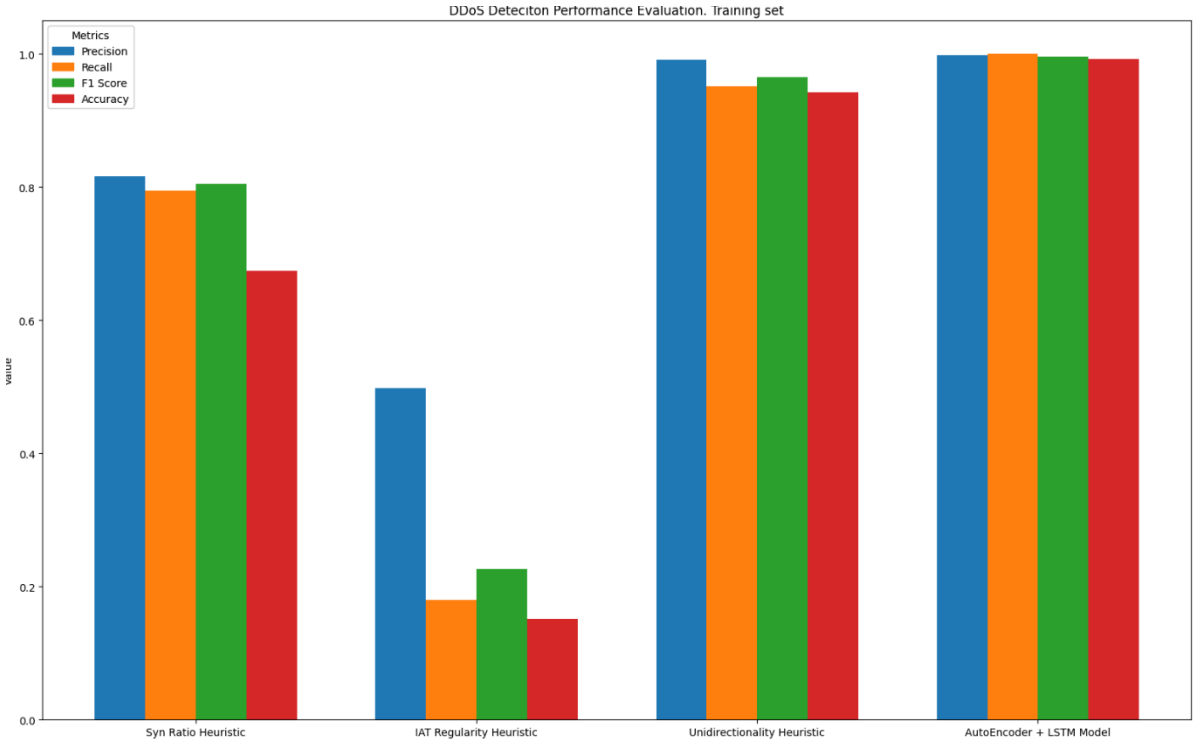


Рисунок 5.9 – Порівняння метрик на тренувальному наборі даних (рисунок виконано самостійно)

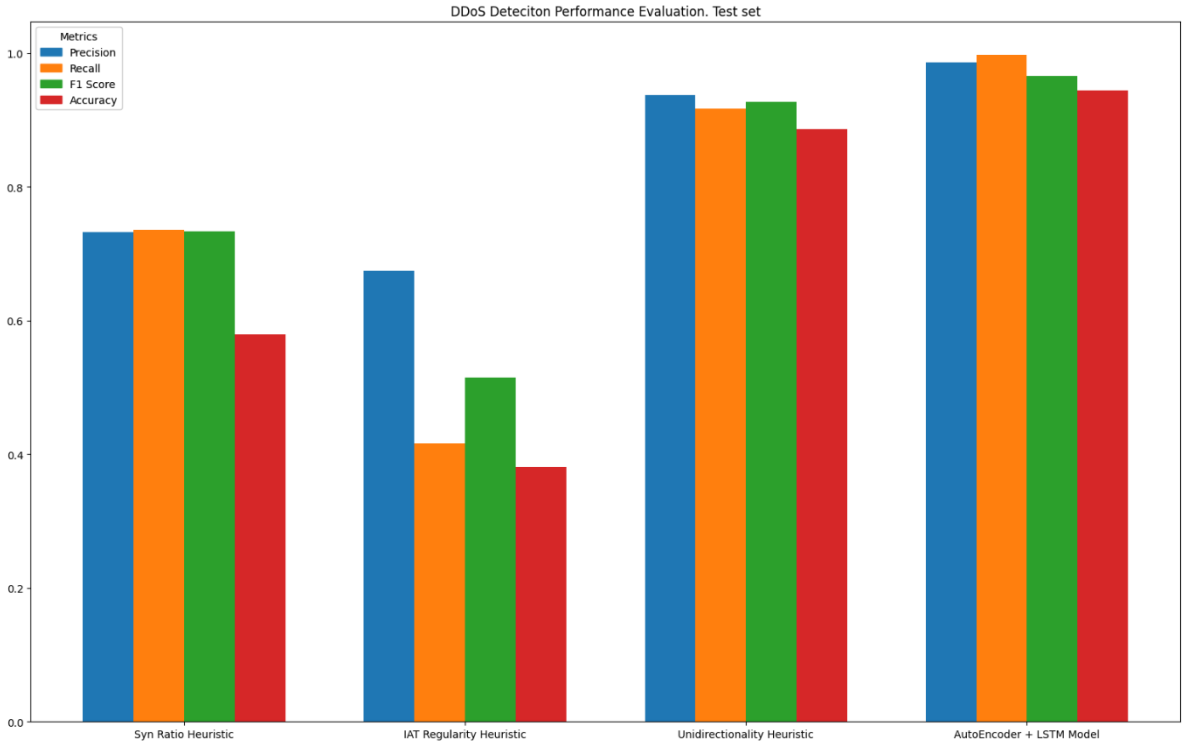


Рисунок 5.10 – Порівняння метрик на тестовому наборі даних(рисунок виконано самостійно)

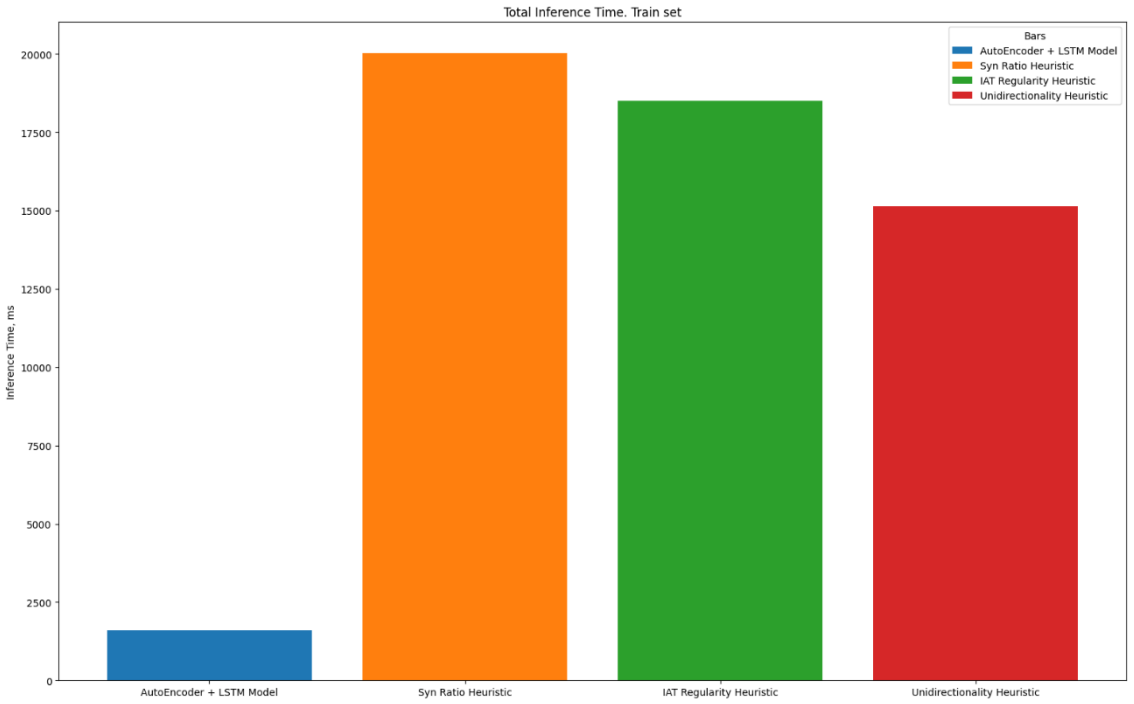


Рисунок 5.11 – Порівняння часу виконання експерименту на тренувальному наборі даних (рисунок виконано самостійно)

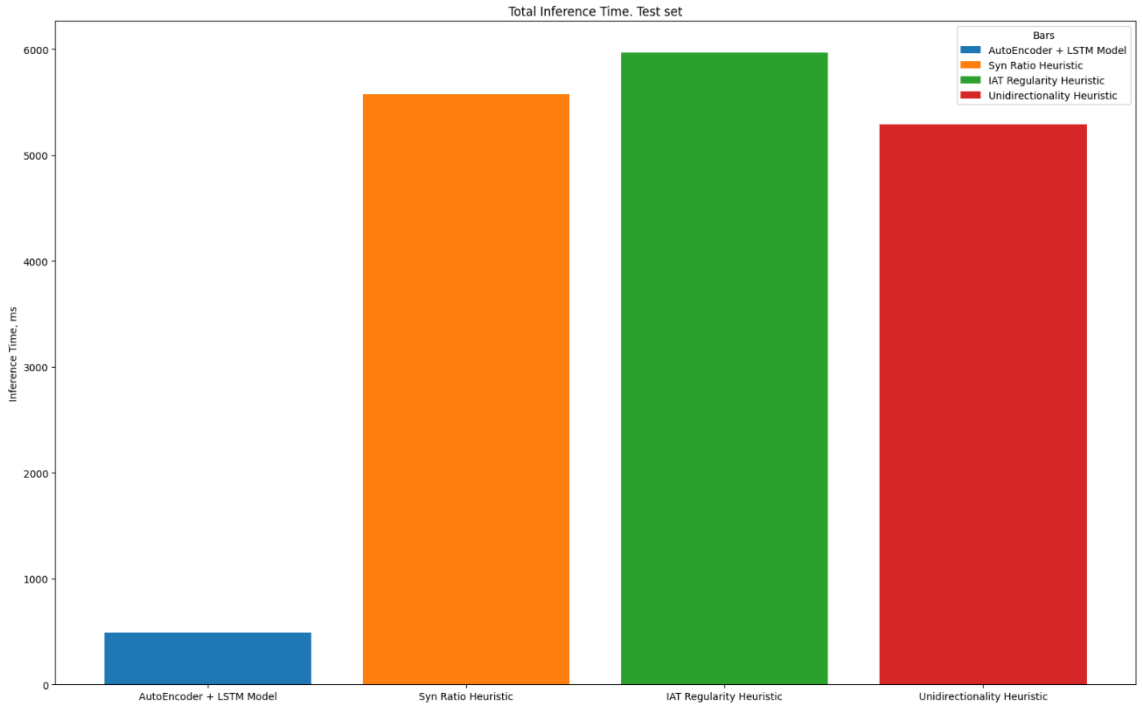


Рисунок 5.12 – Порівняння часу виконання експерименту на тестовому наборі даних (рисунок виконано самостійно)

Більшість розглянутих евристичних методів, зокрема SYN Ratio та IAT Regularity, продемонстрували низьку ефективність саме у виявленні зловмисного трафіку, з близькими до нуля значеннями precision та recall для класу “Attack”. Це свідчить про обмежену здатність таких підходів до коректної ідентифікації складних або адаптивних атак.

Метод Unidirectionality виявився винятком — він показав суттєво кращі результати, з високим recall і прийнятним рівнем precision. Такий підхід може бути корисним у системах, де пріоритетом є чутливість до атак, навіть за ціною зростання хибних спрацювань. Однак, навіть попри це, результати Unidirectionality поступаються гібридній моделі AutoEncoder + LSTM, яка демонструє більш збалансовані метрики — високу точність, низьку кількість хибнопозитивних спрацювань і стабільну класифікацію як нормального, так і зловмисного трафіку.

Таким чином, модель AutoEncoder + LSTM показала найкращу загальну ефективність серед протестованих методів. Вона виявляє широкий спектр атак на основі агрегованих характеристик потоків, зберігаючи здатність адаптуватися до варіативності послідовностей у часі. У той час як евристики можуть використовуватися як швидкі фільтри першого рівня або у системах із простішими вимогами, запропонована глибока модель є більш надійним і точним інструментом для виявлення DDoS-атак у складному середовищі сучасних мереж.

ВИСНОВКИ

У результаті проведеного дослідження було спроєктовано та реалізовано прототип системи виявлення DDoS-атак у зашифрованому мережевому трафіку з використанням методів глибокого навчання. Запропоноване рішення поєднує класичні принципи мережевого моніторингу з сучасними архітектурами моделювання потокових даних у хмарному середовищі.

У теоретичному розділі роботи проаналізовано виклики виявлення загроз у TLS-трафіку, окреслено особливості побудови ознак на основі метаданих та охарактеризовано актуальні підходи до аномалійного аналізу. Обґрунтовано вибір гібридної моделі, в якій AutoEncoder використовується для стискання багатовимірних вхідних даних, а рекурентна нейронна мережа (LSTM) — для аналізу часової структури потоків.

У практичній частині створено систему, що охоплює збір та обробку потоків, класифікацію послідовностей, збереження результатів у InfluxDB та PostgreSQL, а також веб-інтерфейс для аналітиків. Проведені експерименти підтвердили ефективність запропонованої архітектури: модель AutoEncoder + LSTM суттєво перевершує прості евристичні підходи за точністю, особливо у виявленні зловмисного трафіку в складному середовищі.

Отже, поставлену мету дослідження досягнуто. Результати можуть бути використані для подальшої розробки адаптивних систем виявлення DDoS-атак у зашифрованому трафіку, а також для інтеграції в інструменти корпоративного моніторингу безпеки.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. A Survey on Internet Traffic Identification / A. Callado та ін. IEEE Communications Surveys & Tutorials. 2009. Т. 11, № 3. С. 37–52. URL: https://doi.org/10.1109/surv.2009.090304 (дата звернення: 05.02.2025).

2. Bhatia S., Behal S., Ahmed I. Distributed Denial of Service Attacks and Defense Mechanisms: Current Landscape and Future Directions. SpringerLink. URL: https://link.springer.com/chapter/10.1007/978-3-319-97643-3_3 (дата звернення: 06.02.2025).

3. Deep packet: a novel approach for encrypted traffic classification using deep learning / M. Lotfollahi та ін. Soft Computing. 2019. Т. 24, № 3. С. 1999–2012. URL: https://doi.org/10.1007/s00500-019-04030-2 (дата звернення: 08.02.2025).

4. Sharma A., Lashkari A. H. A survey on encrypted network traffic: A comprehensive survey of identification/classification techniques, challenges, and future directions. Computer Networks. 2025. Т. 257. С. 110984. URL: https://doi.org/10.1016/j.comnet.2024.110984 (дата звернення: 10.02.2025).

5. Anderson B., McGrew D. Machine Learning for Encrypted Malware Traffic Classification: Accounting for Noisy Labels and Non-Stationarity. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2017.

6. A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks / C. Yin та ін. IEEE Access. 2017. Т. 5. С. 21954–21961. URL: https://doi.org/10.1109/access.2017.2762418 (дата звернення: 15.02.2025)..

7. Adversarial Autoencoders / A. Makhzani та ін. arXiv.org. URL: https://arxiv.org/abs/1511.05644 (дата звернення: 05.04.2025).

8. Deep learning approach for Network Intrusion Detection in Software Defined Networking / T. A. Tang та ін. 2016 International Conference on Wireless Networks and Mobile Communications. 2016..

9. Lee M.-C., Lin J.-C., Katsikas S. Impact of Recurrent Neural Networks and Deep Learning Frameworks on Real-time Lightweight Time Series Anomaly Detection. 2024.

10. Лазар М. Технології big data у аналізі ризиків страхової компанії / М. Лазар, В. Кобзєв // Інформаційні системи та технології : матеріали статей 7-ї Міжнародної науково-технічної конференції, Коблеве-Харків, 10-15 вересня 2018 р. – Харків : ХНУРЕ, 2018. – С. 364–367.

11. Chala O., Bodyanskiy Y. Matrix Neo-Fuzzy-System and its Online Learning in Image Recognition Task. Information Technology and Management Science. 2021. Т. 24. С. 39–44. URL: https://doi.org/10.7250/itms-2021-0006 (дата звернення: 06.04.2025)..

12. An Approach to the Selection of Behavior Patterns Autonomous Intelligent Mobile Systems / O. Zolotukhin та ін. 2021 IEEE 8th International Conference on Problems of Infocommunications, Science and Technology (PIC S&T). 2021.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ ЗА НАУКОВИМИ НАПРЯМАМИ КЕРІВНИКА ТА НАУКОВЦІВ КАФЕДРИ ПРОГРАМНОЇ ІНЖЕНЕРІЇ

10. Лазар М. Технології big data у аналізі ризиків страхової компанії / М. Лазар, В. Кобзєв // Інформаційні системи та технології : матеріали статей 7-ї Міжнародної науково-технічної конференції, Коблеве-Харків, 10-15 вересня 2018 р. – Харків : ХНУРЕ, 2018. – С. 364–367.