ОДЕСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ імені І.І.МЕЧНИКОВА
(повне найменування вищого навчального закладу)

Факультет математики, фізики та інформаційних технологій
(повне найменування інституту, назва факультету (відділення))

Кафедра математичного забезпечення комп'ютерних систем
(повна назва кафедри (предметної, циклової комісії))

Кваліфікаційна робота

на здобуття рівня вищої освіти «бакалавр»
(рівень вищої освіти)

Система виявлення атак в комп'ютерних мережах
методами глибокого навчання
Computer networks intrusion detection system using deep learning methods

Виконав: студент денної форми навчання
спеціальності 123 – Комп'ютерна інженерія
(шифр і назва напряму підготовки, спеціальності)
Освітня програма «Комп'ютерна інженерія»
(назва освітньої програми)
Крижановський Сергій Сергійович
(прізвище, ім'я, по-батькові)

Керівник к.ф.-м.н., доц. Шпінарева І.М.
(науковий ступінь, вчене звання, прізвище та ініціали, підпис)

Рецензент к.техн.н., доц. Шестопалов С.В.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рекомендовано до захисту:

Протокол засідання кафедри

№___ від «___»_____ 2024 р.

Завідувач кафедри

_____ Євгеній МАЛАХОВ_
(підпис) (ім'я, прізвище)

Захищено на засіданні ЕК № ___

протокол № __ від «___» ____ 2024 р.

Оцінка _____/_____/_____
(за національною шкалою, шкалою ECTS, бали)

Голова ЕК

_____ Алла КОБОЗЄВА_
(підпис) (ім'я, прізвище)

Одеса – 2024

АНОТАЦІЯ

У кваліфікаційній роботі представлено реалізація системи виявлення атак на основі методів глибокого навчання, яка використовується в комп’ютерних мережах.

Зі зростанням кількості кібератак та заподіянням шкоди у різноманітних сферах, корпораціях, підприємствах тощо, стало очевидним що зловмисники вміло використовують слабкі місця комп’ютерних систем для власної вигоди. Саме системи виявлення аномалій у комп'ютерних мережах, на відміну від застарілих систем які базуються на сигнатурах, дозволяють їх користувачам ефективно виявляти нові атаки.

Метою роботи є підвищення рівня безпеки комп'ютерних мереж.

В ході дослідження розробленні компоненти системи які включають в себе модулі сенсору, обробки пакетів та аналізу на аномалії в мережі.

Сердцем системи є модуль аналізу який має два основних компоненти, а саме модуль виявлення атак так модуль їх класифікації, які у свою чергу використовуючи обученні ваги на відкритому наборі даних UNSW-NB15 визначають наявність атаки та класифікують її у разі якщо вона виявлена, відповідно. З боку архітектурного рівня використано суміш із рекурентних та згорткових шарів з довготривалою короткочасною пам’яттю, для першої мережі, а для другої – згорткова мережа так як саме цей тип мереж показав найкращу дійсність у завданні класифікації атак.

ABSTRACT

The qualification work presents the development of an attack detection system based on machine learning methods used in computer networks. With the increasing number of cyberattacks and the damage they cause in various fields, corporations, enterprises, and so on, it has become evident that attackers skillfully exploit vulnerabilities in computer systems for their own benefit. Anomaly detection systems in computer networks, unlike outdated signature-based systems, allow users to effectively identify new attacks.

The aim of the work is to increase the level of security of computer networks.

During the research, components of the system were developed and implemented, including sensor modules, packet processing, notification, and network anomaly analysis.

The core of the system is the analytics module, which has two main components: an attack detection module and an attack classification module. These modules use trained weights on the open UNSW-NB15 dataset to detect and classify attacks, respectively. At the architectural level, a combination of recurrent and convolutional layers with long short-term memory is used for the first network, and a convolutional network is used for the second network, as this type of network has shown the best performance in the task of attack classification.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ І ТЕРМІНІВ

CNN – Convolutional Neural Network,згорткова нейронна мережа

DL – Deep Learning, глибоке навчання

ICMP – Internet Control Message Protocol, протокол міжмережних керуючих повідомлень

IDS – Intrusion Detection System, система виявлення вторгнень

KDD – Knowlage Discovery Databases, виявлення бази знань

LSTM – Long Short-term Memory, довга короткочасна пам’ять

NIDS – Network Intrusion Detection Systems, системи виявлення вторгнень у мережі

OOD – Out-of-Distribution, поза розповсюдженням

OSSEC – Open Source Host-based Intrusion Detection System, хостова система виявлення вторгнень

RNN – Recurrent Neural Network, рекурентна нейронна мережа

SSL – Secure Sockets Layer, рівень захищенних сокетів

TCP – Transmission Control Protocol, протокол керування передачею

ВСТУП

З розвитком технологій та поширенням комп'ютерних мереж значно зросла потреба у забезпеченні їх безпеки. Комп'ютерні мережі є критично важливими інфраструктурами, що використовуються для передачі даних в різних сферах життя, включаючи комерцію, освіту, охорону здоров'я та національну безпеку. Однією з головних загроз для безпеки мереж є кібератаки, які можуть мати катастрофічні наслідки як для окремих користувачів, так і для організацій в цілому. Проблема забезпечення інформаційної безпеки привертають пильну увагу як фахівців в області комп'ютерних систем і мереж, так і численних користувачів, включаючи компанії, що працюють в сфері електронного бізнесу. Задачі забезпечення безпеки корпоративних інформаційних систем традиційно вирішується шляхом побудови системи інформаційної безпеки. У міру зростання і розвитку КІС система інформаційної безпеки повинна легко масштабуватися без втрати цілісності і керованості. Без знання і кваліфікованого застосування сучасних технологій, стандартів, протоколів і засобів захисту інформації неможливо досягти необхідного рівня інформаційної безпеки комп'ютерних систем і мереж.

В 2023 р. в Україні кількість кібератак зросла майже на 16% порівняно з 2022 роком. Загалом за рік було понад 2500 таких інцидентів. За даними урядової команди реагування на комп'ютерні надзвичайні події CERT-UA, 347 кібератак було зафіксовано на уряд та урядові організації, 276 – на місцеві органи влади, 175 – на організації у секторі безпеки та оборони, 127 – комерційні організації.Ще 92 рази атакували енергетичний сектор, 81 – телеком, 38 – освітні установи, 32 – транспортна галузь, 30 – фінансовий сектор, 25 – ІТ-сектор, 15 – ЗМІ, 12 – медичні установи [1].

На сьогоднішній день існує багато методів та технологій для виявлення атак у комп'ютерних мережах. Однак, зростання складності та обсягу мережевих даних вимагає більш ефективних підходів до виявлення загроз. У цьому контексті методи глубокого навчання є перспективними засобами для

розробки систем виявлення атак. Глибоке навчання, як один з напрямів машинного навчання, дозволяє створювати моделі, які здатні автоматично аналізувати великі обсяги даних та виявляти приховані закономірності, що можуть свідчити про наявність атак. Використання технологій, таких як згорткові нейронні мережі CNN та рекурентні нейронні мережі з довготривалою короткочасною пам'яттю LSTM, значно підвищує ефективність системи виявлення атак.

Методи глибокого навчання мають кілька переваг перед традиційними підходами, заснованими на сигнатурах та базі знань. Вони можуть адаптуватися до нових, невідомих раніше атак, швидше обробляти великі обсяги даних та виявляти більш складні загрози.

Метою роботи є підвищення рівня безпеки комп'ютерних мереж.

Основні задачі цієї кваліфікаційної роботи є:

- дослідження методів виявлення атак;
- визначення вимог до створеної системи;
- вибір архітектури, технологій та засобів реалізації системи;
- розробка алгоритму виявлення та класифікації атак;
- тестування алгоритму на тестових даних.

1 АНАЛІЗ СИСТЕМ ТА МЕТОДІВ ВИЯВЛЕННЯ АТАК

1.1 Аналіз існуючих систем виявлення вторгнень

З розвитком технологій та поширенням комп’ютерних мереж значно зросла потреба у забезпеченні їх безпеки. Комп’ютерні мережі є критично важливими інфраструктурами, що використовуються для передачі даних в різних сферах життя, включаючи комерцію, освіту, охорону здоров’я та національну безпеку. Саме кібератаки складають одну з найголовніших загроз для безпеки мереж, що несе за собою катастрофічні матеріальні та фінансові наслідки не лише для підприємств, а й для самих клієнтів.

Системи виявлення мережевих вторгнень виконують роль невпинних вартових, постійно контролюючи мережевий трафік. Їх основне завдання – виявлення несанкціонованих або потенційно шкідливих дій, які загрожують конфіденційності, доступності та цілісності комп'ютерних і мережевих систем [2]. Сучасні методи виявлення аномалій IDS використовують алгоритми глубокого навчання DL для розпізнавання аномальних шаблонів атак [3]. Проте кіберзлочинці безперервно розробляють нові тактики для проведення складних маневрів, включаючи атаки поза розподілом OOD та атаки-обманки, щоб обійти механізми виявлення. Крім того, вони можуть створювати нові атаки нулевого дня, використовуючи невідомі вразливості, які ще не були усунуті системами безпеки. Це ставить серйозні виклики для виявлення вторгнень у мережевому трафіку, що вимагає постійного вдосконалення та інноваційного розвитку можливостей IDS. Постійне протистояння між захисниками та нападниками змушує IDS удосконалюватися, щоб ефективно виявляти та протидіяти змінюваним стратегіям супротивників [4].

Основна задача IDS полягае в тому, що ця система є потужним механізмом для моніторингу та анлізу дій мережевого трафіка на виявлення небезпечної поведінки з боку системи. Системи виявлення вторгнень IDS классифікуются за кількома критеріями [4] :

– сигнатурне виявлення (Signature-Based Detection) полягає у порівнянні мережевого трафіку з базою даних відомих сигнатур атак, які є шаблонами або відбитками відомих атак. Ця система заточена під виявлення вже відомих атак з чітко визначеними сигнатурами. Її основні переваги включають швидке та точне виявлення відомих атак, а також низьку кількість помилкових спрацьовувань. Однак, вона не здатна виявляти нові або змінені атаки, які не входять у базу даних сигнатур, і потребує постійного оновлення цієї бази для ефективної роботи;

– виявлення аномалій (Anomaly-Based Detection) створює модель нормальної поведінки мережі та виявляє відхилення від цієї моделі, які можуть вказувати на аномальні або шкідливі дії. Ця система заточена під виявлення нових, раніше невідомих атак, які не відповідають нормальній поведінці. Серед її переваг можна виділити здатність виявляти нові та невідомі атаки, а також швидке реагування на нові загрози завдяки виявленню відхилень від нормальної поведінки. Проте, система має високий рівень помилкових спрацьовувань через непередбачувані легальні зміни в поведінці мережі та складність у налаштуванні моделі нормальної поведінки;

– виявлення на основі станів (Stateful Protocol Analysis) аналізує протоколи та перевіряє, чи відповідає поточна поведінка протоколу визначеним правилом і очікуванням. Ця система заточена під виявлення аномалій на рівні протоколів, що відрізняються від стандартного або очікуваного поведінки. Її основні переваги включають глибокий аналіз протокольних станів, що дозволяє виявляти складні атаки, та ефективне виявлення відхилень у поведінці протоколів. Однак, система має високі вимоги до обчислювальних ресурсів через складний аналіз і потребує детального знання нормальної поведінки протоколів для точного налаштування;

– хостові (HIDS) встановлюються безпосередньо на окремих хостах або пристроях і моніторять активність на цьому хості, включаючи журнали подій, системні виклики, мережеві підключення тощо. Ця система заточена під захист окремих хостів або серверів від внутрішніх та зовнішніх загроз. Її

основні переваги включають глибокий аналіз подій на рівні хоста та можливість виявлення атак, які не видно на мережевому рівні. Однак, система вимагає встановлення на кожен хост, що може бути трудомістким, і може не виявляти атаки, що проходять через мережу, але не впливають на конкретний хост;

– мережеві (NIDS) моніторять весь трафік мережі або її сегментів, аналізуючи пакети, що проходять через мережу. Ця система заточена під загальний моніторинг мережі для виявлення атак на рівні мережевого трафіку. Її основні переваги включають можливість моніторингу великої кількості трафіку та виявлення атак, що націлені на мережеві ресурси. Однак, система може пропустити атаки, які не залишають слідів у мережевому трафіку, наприклад, атаки на рівні додатків, і вимагає правильного розміщення у мережі для ефективного моніторингу всього трафіку;

– мережеві пакети аналізуються IDS для виявлення шкідливої активності на рівні окремих пакетів мережевого трафіку. Ця система заточена під детальний аналіз кожного мережевого пакета для виявлення аномалій або відомих шкідливих патернів. Основні переваги включають високу точність виявлення на рівні окремих пакетів і можливість швидкого реагування на виявлені загрози. Однак, система має високу обчислювальну навантаженість через необхідність аналізувати кожен пакет і може пропускати атаки, що вимагають аналізу більшої кількості пакетів, таких як сесії;

– мережеві сесії аналізуються IDS для виявлення шкідливої активності у потоках або сесіях даних, що включають кілька пакетів. Ця система заточена під аналіз тривалих з'єднань або сесій для виявлення складних атак, які не можуть бути виявлені на рівні окремих пакетів. Основні переваги включають можливість виявлення складних атак, що проявляються лише у контексті сесії, і більш глибокий аналіз трафіку. Однак, система вимагає більше часу та ресурсів для аналізу сесій і може бути складнішою у налаштуванні та в використані;

– журнали подій аналізуються IDS для виявлення аномальної або шкідливої активності в операційній системі або додатках. Ця система заточена під виявлення внутрішніх загроз та шкідливої активності, яка може не бути

видимою на мережевому рівні. Основні переваги включають глибокий аналіз дій всередині системи та можливість виявлення складних внутрішніх атак. Однак, великий обсяг даних для аналізу може потребувати значних ресурсів, і система може не виявляти зовнішні атаки, що не відображаються у журналах подій.

Сигнатурне виявлення, виявляючи вторгнення шляхом порівняння мережевого трафіку з базою даних відомих сигнатур атак, ефективно виявляє лише відомі атаки. Однак цей метод не може виявляти нові або змінені атаки, які ще не мають відповідних сигнатур у базі даних, що робить його недостатнім для захисту від постійно змінюваних загроз. Виявлення аномалій, яке виявляє відхилення від нормальної поведінки системи або мережевого трафіку, здатне ідентифікувати нові, раніше невідомі атаки. Проте цей метод часто страждає від високого рівня помилкових спрацьовувань через непередбачувані зміни в легальній поведінці мережі, що ускладнює його використання. Виявлення на основі станів аналізує протоколи і перевіряє відповідність поведінки протоколу визначеним правилам і очікуванням, але потребує значних обчислювальних ресурсів і детального знання нормальної поведінки протоколів для точного налаштування.

Мережеві NIDS моніторять трафік всієї мережі або її сегментів, забезпечуючи загальний моніторинг мережі для виявлення атак на рівні мережевого трафіку. Вони можуть пропустити атаки, які не залишають слідів у мережевому трафіку, наприклад, атаки на рівні додатків, і вимагають правильного розміщення у мережі для ефективного моніторингу.

Хостові IDS HIDS, встановлені на окремих хостах або пристроях, моніторять і аналізують активність на цих хостах, включаючи журнали подій, системні виклики і мережеві підключення, але вимагають встановлення на кожен хост і можуть не виявляти атаки, які проходять через мережу, але не впливають на конкретний хост.

Альтернативні системи виявлення вторгнень, такі як Suricata, можуть забезпечити більш комплексний підхід до захисту мереж. Suricata, наприклад, є потужною системою виявлення вторгнень, яка може працювати як NIDS або

HIDS і підтримує як сигнатурне виявлення, так і виявлення на основі аномалій [4]. Suricata здатна аналізувати мережеві пакети, потоки даних і журнали подій, забезпечуючи багаторівневий захист мережі від різних типів загроз. Завдяки підтримці багатопотоковості та використанню сучасних технологій, таких як Deep Packet Inspection (DPI), Suricata забезпечує високу продуктивність і точність виявлення атак, що робить її ефективною альтернативою для захисту мережі.

Suricata підтримує кілька режимів функціонування, що дозволяє гнучко налаштовувати систему під різні потреби мережевої безпеки. Основні режими включають: режим виявлення; режим запобігання; режим мережевого моніторингу; режим журналювання.

Архітектура Suricata є модульною та багатопотоковою, що забезпечує високу продуктивність і гнучкість. Основні компоненти архітектури Suricata включають: вхідні модулі; декодери; сигнатурні двигуни; двигуни виявлення аномалій; модулі обробки; вихідні модулі; модулі запобігання.

Вхідні модулі в Suricata відповідають за прийом мережевого трафіку з різних джерел, таких як мережеві інтерфейси, файли Pcap або Unix-сокети.

Suricata підтримує декілька інтерфейсів одночасно, що дозволяє обробляти трафік з різних сегментів мережі. Це забезпечує гнучкість і масштабованість у налаштуванні моніторингу різноманітних мережевих середовищ [5].

Декодери аналізують отримані пакети, інтерпретуючи їхні заголовки і корисне навантаження. Suricata підтримує велику кількість протоколів, включаючи Ethernet, IP, TCP, UDP, HTTP, SSL та багато інших.

Завдяки цій підтримці декодери можуть глибоко аналізувати мережевий трафік на різних рівнях, від фізичного до прикладного [5].

Двигуни виявлення аномалій використовуються для виявлення відхилень від нормальної поведінки мережевого трафіку. Це дозволяє виявляти нові або невідомі загрози, які не можуть бути знайдені за допомогою сигнатур. Ці двигуни аналізують патерни трафіку та виявляють аномалії, що можуть свідчити про потенційні атаки або інші шкідливі дії [6].

Модулі обробки включають різні компоненти для додаткової обробки даних, такі як модулі для агрегування даних, нормалізації, фільтрації тощо. Вони забезпечують попередню обробку та підготовку даних для подальшого аналізу, допомагаючи зменшити шум і підвищити точність виявлення загроз.

Ця архітектура забезпечує комплексний підхід до виявлення та запобігання мережевим загрозам, поєднуючи різні методи аналізу трафіку та гнучкість у налаштуванні й інтеграції з іншими системами безпеки.

Архітектура Suricata забезпечує високу масштабованість і гнучкість, дозволяючи адаптуватися до різних вимог та сценаріїв використання, що робить її потужним інструментом для забезпечення мережевої безпеки.

Bro, що тепер відома як Zeek, – це потужна система виявлення мережевих вторгнень, яка спеціалізується на глибокому аналізі мережевого трафіку. Вона відзначається своєю здатністю аналізувати мережевий трафік на високому рівні, що дозволяє виявляти складні атаки та аномалії в поведінці мережі. Zeek підтримує широкий спектр протоколів, включаючи HTTP, DNS, FTP та багато інших, що забезпечує гнучке налаштування для різних потреб безпеки. Завдяки своєму багатому функціоналу, Zeek може не тільки виявляти загрози, але й проводити детальний аналіз трафіку для розслідування інцидентів.

Zeek використовує скриптову мову для написання власних політик безпеки та виявлення атак. Це дозволяє користувачам створювати кастомізовані правила та аналітичні модулі, що робить її особливо корисною для детального аналізу та розслідування інцидентів безпеки. Крім того, Zeek забезпечує можливість інтеграції з іншими інструментами та платформами, що підвищує його функціональність і дозволяє створити комплексну систему безпеки. Завдяки цим можливостям, Zeek є відмінним вибором для організацій, які потребують потужного та гнучкого інструменту для забезпечення безпеки мереж [7].

Open Source Security Information and Event Management – це потужна хостова система виявлення вторгнень, яка також забезпечує функціональність системи управління подіями та інформацією безпеки SIEM. OSSEC

спеціалізується на моніторингу журналів подій, системних викликів та інших аспектів безпеки хостів.

Вона дозволяє виявляти як внутрішні, так і зовнішні загрози, забезпечуючи високий рівень захисту для різних операційних систем, включаючи Windows, Linux та MacOS.

Однією з головних переваг OSSEC є її здатність до інтеграції з іншими інструментами безпеки, що дозволяє створити комплексну систему моніторингу та реагування на інциденти. Вона підтримує централізовану конфігурацію і управління, що спрощує її розгортання та підтримку в великих мережах. OSSEC також включає можливості для автоматичного реагування на виявлені загрози, що дозволяє миттєво вживати заходів для захисту систем [8].

Snort – це широко відома система виявлення мережевих вторгнень, яка використовує сигнатурне виявлення для ідентифікації відомих атак.

Snort здатна аналізувати мережевий трафік у реальному часі, застосовуючи правила для виявлення шкідливої активності. Вона підтримує великий набір протоколів, включаючи TCP, UDP, ICMP та багато інших, що дозволяє використовувати її в різних мережевих середовищах. Система Snort є відкритим програмним забезпеченням, що дозволяє користувачам налаштовувати та розширювати систему під свої потреби. Це забезпечує високу гнучкість і можливість інтеграції з іншими системами безпеки.

Завдяки підтримці різних модулів і плагінів, Snort може бути адаптована для специфічних вимог безпеки, що робить її одним з найпопулярніших інструментів для виявлення загроз у світі. Вона широко використовується як у корпоративних мережах, так і в академічних середовищах для навчання та досліджень у сфері кібербезпеки [9].

В результаті аналізу з'ясувалося, що Suricata та SNORT підтримають ОС Unix та Windows, Bro підтримує ОС Unix та MacOS, а OSSEC всі ОС.

Але важно порівняти сучасні системи за методами виявлення атак та за рівнями спостереження (табл. 1.1).

Таблиця 1.1 – Порівняння існуючих систем виявлення вторгнень

	Методи виявлення атак				Рівень спостереження	
	Експерт-ний	Сигна-турний	Статис-тичний	Машин-ного навчання	Систем-ний	Мереже-вий
Suricata	-	+	+	-	-	+
SNORT	-	+	+	-	-	+
Bro	-	+	-	-	-	+
OSSEC	-	+	-	-	+	-

1.2 Аналіз методів виявлення вторгнень, заснованих на знаннях

Методи виявлення вторгнень, засновані на знаннях, є однією з двох основних тенденцій у цій галузі. Вони використовують накопичені знання про конкретні атаки та вразливості системи, шукаючи докази їх експлуатації. Система виявлення вторгнень зберігає інформацію про ці вразливості та розраховує на спроби їх використання. Коли така спроба виявляється, спрацьовує сигнал тривоги. Іншими словами, будь-яка дія, яка явно не визнана атакою, вважається прийнятною. Завдяки цьому точність систем виявлення вторгнень, заснованих на знаннях, вважається високою, хоча їх ефективність залежить від регулярного оновлення знань про атаки.

Переваги підходів, заснованих на знаннях, полягають у їх здатності мати низький рівень помилкових тривог та детально аналізувати контекст, що полегшує розуміння проблеми службою безпеки та вжиття профілактичних або коригувальних заходів. Недоліками цих методів є труднощі у зборі необхідної інформації про відомі атаки та підтриманні її в актуальному стані щодо нових вразливостей та середовищ. Підтримка бази знань системи виявлення вторгнень вимагає ретельного аналізу кожної вразливості, що є трудомістким завданням.

Методи, засновані на знаннях, також стикаються з проблемою узагальнення, оскільки знання про атаки сильно залежать від операційної системи, версії, платформи та програми.

Отже, отримана система виявлення вторгнень тісно пов’язана із заданим середовищем.

Крім того, виявлення інсайдерських атак, пов’язаних із зловживанням привілеями, є більш складним, оскільки зловмисник фактично не використовує жодної вразливості [10].

1.2.1 Експертні системи

Системи виявлення вторгнень, що базуються на знаннях, зазвичай використовують експертні системи для виявлення атак. Експертні системи складаються з набору правил, що описують різні види атак. Події інтерпретуються як факти, що мають певне семантичне значення у системі, і механізм виведення робить висновки, використовуючи ці правила та факти. Такий підхід підвищує рівень абстракції даних аудиту, додаючи до них смисловий контекст.

Мови, засновані на правилах, є природним засобом для моделювання знань, накопичених експертами щодо атак. Вони дозволяють систематично переглядати журнали подій у пошуках доказів використання відомих вразливостей і також використовуються для перевірки дотримання політики безпеки організації. Моделювання, що базується на моделі поведінки, яке також використовує експертні системи, але додає додаткові властивості, було розроблено Гарві та Лунтом. Воно описує поведінку зловмисника через його цілі, дії для досягнення цих цілей та використання системи, що іноді виявляє певний рівень параної. Інструмент сканує наявність доказів цих дій і переходів.

Такий підхід виявив ряд обмежень у кількох ключових областях. Однією з основних проблем є інженерія знань: важко витягувати знання про атаки і ще складніше перетворювати ці знання у правила, використовуючи дані аудиту як

вхідні дані. Часом необхідна інформація відсутня під час перевірок. Крім того, існує багато способів використання однієї вразливості, що призводить до необхідності створення великої кількості правил. Ще однією проблемою є швидкість обробки: використання оболонки експертної системи вимагає імпортування всіх перевірок як фактів до оболонки, після чого відбувається процес міркування. Незважаючи на те, що деякі інструменти експертної системи дозволяють оптимізувати правила, загальна продуктивність інструменту часто залишається низькою. Через проблеми швидкості обробки оболонки експертних систем використовуються переважно у прототипах, тоді як комерційні продукти обрали більш ефективні підходи.

1.2.2 Аналіз сигнатур

Аналіз сигнатур дотримується принципу, подібного до того, який використовують експертні системи, проте знання застосовуються інакше. Семантичний опис атак перетворюється на інформацію, яку можна безпосередньо знайти в аудиторському сліді. Наприклад, сценарії нападу можуть бути трансформовані в послідовності аудиторських подій або шаблони даних, які можуть бути використані в аудиторському сліді, сформованому системою. Такий підхід знижує семантичний рівень описів атак, дозволяючи дуже ефективно реалізувати систему виявлення вторгнень, що широко застосовується в комерційних продуктах виявлення вторгнень .

Основним недоліком цієї техніки, загальним для всіх підходів, що базуються на знаннях, є необхідність частого оновлення, щоб не відставати від потоку нових вразливостей. Ця ситуація ускладнюється вимогою представляти всі можливі аспекти атак сигнатурами, що призводить до того, що кожна атака повинна бути представлена кількома сигнатурами для різних операційних систем, на які розгорнуто систему виявлення вторгнень [9].

Експертні системи в основному використовуються на основі знань методами виявлення вторгнень. Експертна система містить набір правил, що

описують атаки. Події перекладаються у факти, що мають їх семантичне значення в експертній системі, і механізм висновків робить висновки, використовуючи ці правила та факти. Цей метод підвищує рівень абстракції даних аудиту, додаючи до них семантику.

Мови, засновані на правилах, є природним інструментом для моделювання знань, які експерти зібрали про атаки. Такий підхід дозволяє систематично переглядати аудиторський слід у пошуках доказів спроб використання відомих вразливих місць. Вони також використовуються для перевірки належного застосування політики безпеки організації.

Моделювання на основі моделей, яке також використовує експертні системи, але має додаткові властивості, було представлено Гарві та Лунтом. Знання про поведінку зловмисника описуються цілями зловмисника, діями, які він робить для досягнення цих цілей, та його використанням системи, яка іноді виявляє певний рівень параної. Потім інструмент сканує перевірки на наявність доказів цих дій та переходів .

Такий підхід до використання мов, заснованих на правилах, продемонстрував обмеження в наступних областях: інженерія знань, швидкість обробки. Інженерія знань включає складнощі з витягуванням знань про атаки та переведенням їх у правила, використовуючи аудит як вхідні дані. Часто необхідна інформація відсутня під час перевірок, що ускладнює процес. Крім того, може бути багато способів використати дану вразливість, що призводить до необхідності створення численних правил. Щодо швидкості обробки, використання оболонки експертної системи вимагає імпортування всіх перевірок як фактів до оболонки, після чого можливе міркування.

Незважаючи на те, що деякі інструменти експертної системи дозволяють складати правила, загальна продуктивність інструменту часто залишається низькою. Через проблеми швидкості обробки, оболонки експертних систем використовуються лише в прототипах, тоді як комерційні продукти обрали більш ефективні підходи.

1.3 Огляд методів виявлення вторгнень, заснованих на поведінці

Методи виявлення вторгнень, засновані на поведінці, використовуються для визначення загроз шляхом спостереження за відхиленнями від звичної або очікуваної поведінки системи або користувачів. Для цього спочатку формується модель нормальної поведінки на основі зібраних даних за допомогою різних методів. Потім система виявлення вторгнень порівнює поточну активність з цією моделлю і генерує сигнал тривоги при виявленні аномалій. Все, що не відповідає раніше визначеним нормам, розглядається як потенційна загроза. Хоча цей підхід дозволяє виявляти різноманітні загрози, він стикається з проблемами точності.

Однією з головних переваг поведінкових методів є їх здатність виявляти нові та непередбачувані вразливості, що дозволяє автоматично ідентифікувати нові типи атак. Ці методи менш залежні від механізмів, специфічних для конкретних операційних систем, що робить їх більш універсальними. Проте, високий рівень помилкових тривог залишається значною проблемою, оскільки не всі аспекти поведінки можуть бути враховані на етапі навчання.

Крім того, поведінка користувачів і систем змінюється з часом, що вимагає періодичного оновлення моделей поведінки. Це може призвести до тимчасової недоступності системи виявлення вторгнень або збільшення кількості помилкових тривог. Зрештою, методи виявлення вторгнень, засновані на поведінці, є потужним інструментом для ідентифікації загроз, проте вони потребують ретельного налаштування та постійного оновлення для забезпечення високої точності.

1.3.1 Статистичний метод

Одним із найпопулярніших методів для побудови систем виявлення вторгнень, заснованих на поведінці, є використання статистичних методів. Поведінка користувача або системи оцінюється за допомогою різних

параметрів, які вимірюються протягом певного часу [10]. Наприклад, до таких параметрів належать час входу та виходу з системи під час кожного сеансу, тривалість використання ресурсів, а також кількість ресурсів процесора, пам'яті та диска, які споживаються під час сеансу. Інтервали вибірки можуть варіюватися від кількох хвилин до кількох місяців. Початкова модель зберігає середні значення всіх цих змінних і визначає, чи перевищені порогові значення на основі стандартного відхилення змінної. Однак, така проста модель часто не здатна точно представляти дані. Навіть порівняння параметрів окремих користувачів із загальною груповою статистикою не завжди призводить до суттєвого покращення.

Для вирішення цієї проблеми була розроблена більш складна модель, яка порівнює профілі довгострокової та короткострокової активності користувачів. Ці профілі регулярно оновлюються, щоб відображати зміни в поведінці користувачів. Така складніша статистична модель наразі використовується в багатьох системах виявлення вторгнень та їх прототипах. Вона дозволяє точніше виявляти аномалії, оскільки враховує як короткострокові, так і довгострокові зміни в поведінці, що значно підвищує ефективність системи виявлення вторгнень.

1.3.2 Методи глибокого навчання

Ефективність традиційних методів машинного навчання у виявленні аномальних значень є обмеженою при роботі з послідовними наборами даних, оскільки вони не можуть повністю захопити складні структури даних. Крім того, зростання обсягів даних, наприклад, до гігабайтних масштабів, робить традиційні методи практично непридатними для аналізу таких великих масивів даних, ускладнюючи виявлення аномалій.

Методи глибокого виявлення аномалій використовують ієрархічні ознаки для дискримінаційного аналізу даних. Ця здатність до автоматичного виділення характеристик знімає необхідність у ручному налаштуванні

параметрів експертами у відповідній області. Методи глибокого навчання, що найчастіше застосовуються у виявленні аномалій, можна умовно поділити на три основні категорії [11], як показано на рисунку 1.1:

– генеративні методи, такі як DCA, SAE, RBM, DBN, CVAE;
– гібридні методи, такі як GAN ;
– дискримінаційні методи, такі як RNN, LSTM, CNN.

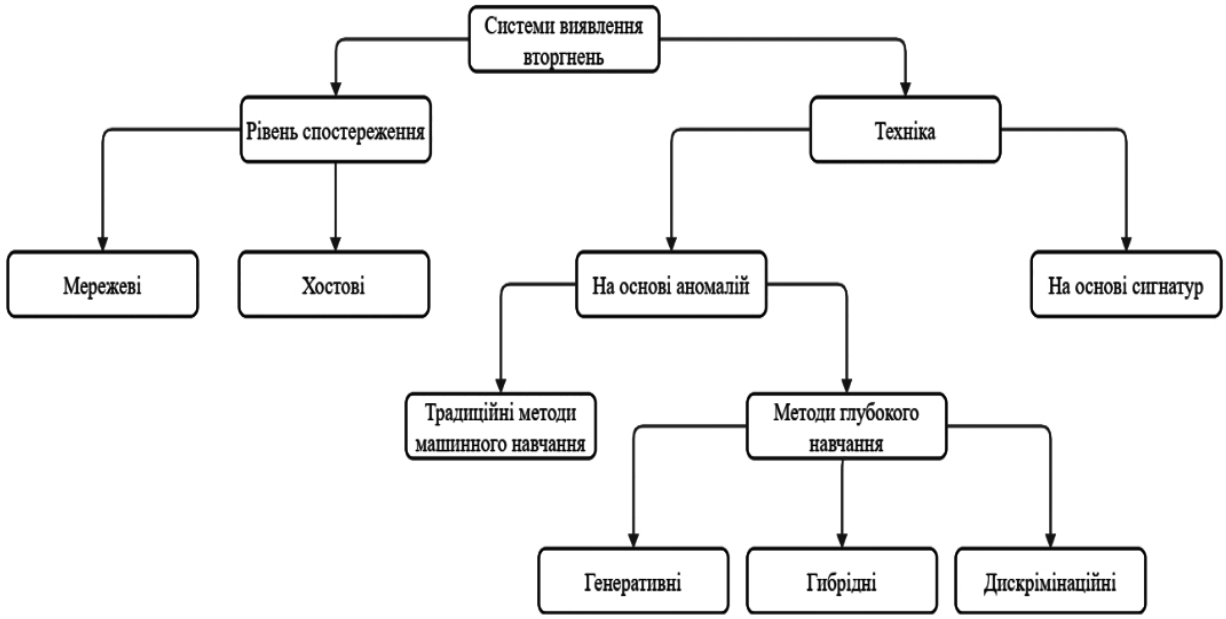

Рисунок 1.1 – Класифікація методів глибокого навчання для виявлення вторгнень

Розглянемо дискримінаційні методи глибокого навчання, оскільки вони безпосередньо спрямовані на вирішення завдання виявлення вторгнень.

Згорткова нейронна мережа є спеціалізованим типом нейронної мережі, призначеним для обробки даних з матричною топологією (рис. 1.2). Прикладами таких даних можуть бути тимчасові ряди, які можна розглядати як одновимірну сітку даних, відібрану через регулярні інтервали часу, або зображення, що представляють собою двовимірну сітку пікселів.

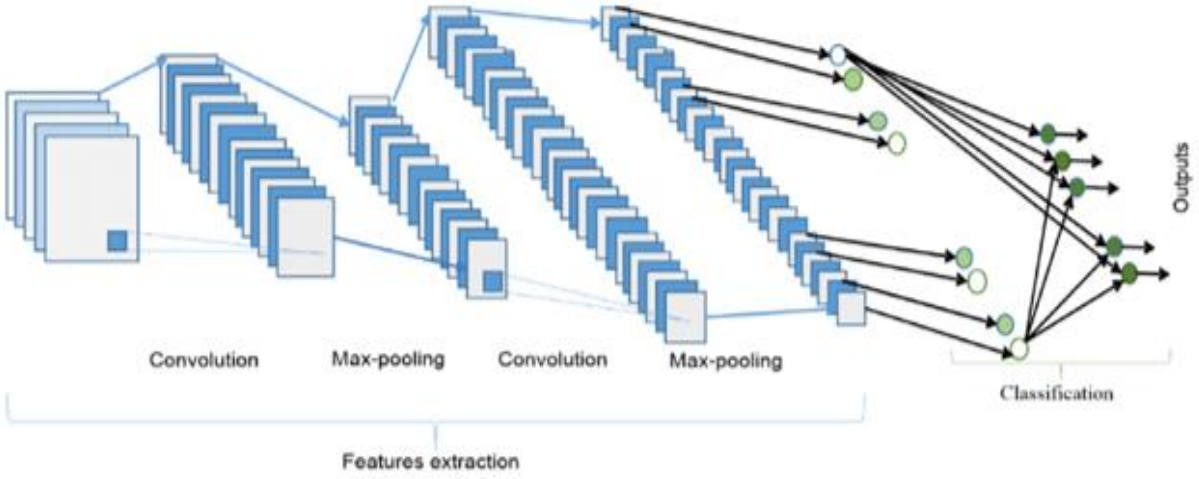

Рисунок 1.2 – Архітектура згорткової нейронної мережі

CNN особливо добре підходять для аналізу зображень і відео, а також можуть бути ефективно використані для обробки послідовних даних і тимчасових рядів. Завдяки своїй здатності автоматично виявляти важливі характеристики з даних, згорткові нейронні мережі стали популярним інструментом у багатьох завданнях, пов'язаних з аналізом даних і виявленням аномалій.

В основі згорткових нейронних мереж лежать три ключові механізми, які використовуються для досягнення інваріантності до переносу, масштабування та незначних змін.

Перший механізм – локальний витяг ознак, коли кожен нейрон отримує вхідний сигнал від локального рецептивного поля в попередньому шарі, що дозволяє виділяти локальні ознаки незалежно від їх точного розташування.

Другий механізм – формування шарів у вигляді набору карт ознак, де кожен шар складається з багатьох карт, на яких всі нейрони використовують одні й ті ж синаптичні ваги. Це дозволяє досягти інваріантності до зміщення та зменшити кількість вільних параметрів.

Третій механізм – підвибірка, де кожен шар згортки супроводжується обчислювальним шаром, що виконує локальне усереднення і підвибірку, зменшуючи роздільну здатність карт ознак і знижуючи чутливість до незначних змін.

Послідовне застосування згортки та підвибірки призводить до підвищення рівня ознак: перші шари витягують локальні ознаки з окремих областей зображення, а наступні шари витягують загальні ознаки, які називаються ознаками високого порядку. З іншого боку, рекурентні нейронні мережі є спеціалізованими для обробки послідовних даних (рис. 1.3). На відміну від згорткових мереж, які обробляють сітки значень, RNN призначені для обробки послідовностей значень x(1), ..., x(τ). RNN добре масштабуються для обробки довгих послідовностей даних, що було б практично неможливо для стандартних нейронних мереж, і можуть обробляти послідовності змінної довжини. Всі рекурентні мережі мають форму ланцюжка повторюваних модулів нейронної мережі, де в стандартних RNN цей модуль має просту структуру, наприклад, один шар з гіперболічним тангенсом [13].

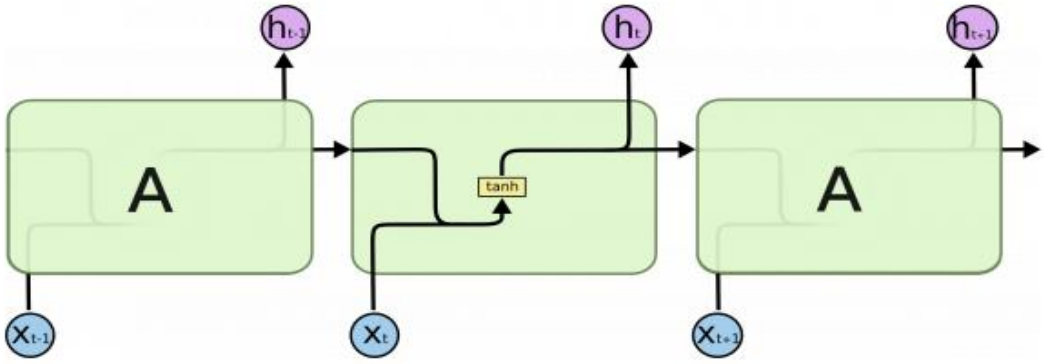

Рисунок 1.3 – Архітектура RNN мережі

Основною проблемою використання рекурентних нейронних мереж є математична складність навчання довгостроковим залежностям: градієнти, що поширюються через багато шарів, можуть або зникати, або вибухово зростати. Навіть якщо рекурентна мережа є стабільною за заданими параметрами, тобто здатна зберігати спогади без вибухового зростання градієнтів, все одно існує проблема призначення довгостроковим залежностям експоненціально менших ваг порівняно з короткостроковими.

Довга короткострокова пам'ять LSTM – це тип рекурентної нейронної мережі, що може навчатися довгостроковим залежностям. LSTM спеціально розроблені для подолання проблеми довгострокової залежності. Вони здатні запам'ятовувати інформацію протягом тривалих періодів, що значно зменшує потребу у додатковому навчанні. Ключовою особливістю LSTM-мережі є те, що кожен блок мережі має пам'ять Ct, яка зберігає інформацію, отриману в попередній момент часу. Ці блоки містять декілька вентильних функцій, які визначають, чи нова інформація є важливою для задачі прогнозування і чи залишається стара інформація актуальною (рис. 1.4).

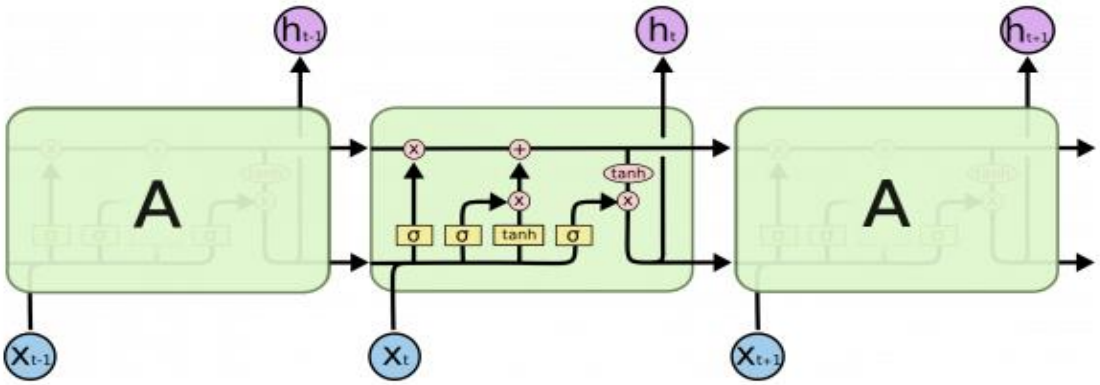

Рисунок 1.4 – Ланцюжок модулів рекурентної нейронної мережі з довгою короткостроковою пам’яттю

Це дозволяє зберігати інформацію, отриману на набагато раніших етапах послідовності, протягом всієї обробки. Це є головною перевагою архітектури LSTM у порівнянні з іншими архітектурами RNN.

1.4 Набори даних для досліджень IDS

Для створення ефективної моделі системи виявлення вторгнень важливо мати доступ до великого обсягу даних для навчання та тестування. Якість цих даних є вирішальною для результативності моделі IDS. Низькоякісна або

нерелевантна інформація може негативно вплинути на результати, тому її необхідно вилучати після збору статистичних характеристик з доступних атрибутів та елементів.

Однак, дані можуть бути недостатніми, неповними, незбалансованими, багатовимірними або дуже великими. Тому важливо провести глибокий аналіз наявних наборів даних перед використанням їх у дослідженнях IDS.

Набори даних KDD Cup 99 та UNSW-NB15 є двома з найвідоміших наборів даних для IDS. KDD Cup 99, який містить понад п'ять мільйонів екземплярів даних про 27 типів атак, є одним із найпоширеніших. Пізніше цей набір був покращений шляхом видалення надмірних записів, що призвело до створення NSL-KDD у 2009 році.

Однак, KDD Cup 99 вважається застарілим, оскільки мережевий трафік, на якому він базується, був створений у 1998 році і не відображає сучасні мережеві структури та динаміку атак [14].

Набір даних UNSW-NB15 був створений за допомогою інструменту IXIA PerfectStorm у лабораторії Cyber Range Австралійського центру кібербезпеки. Цей набір даних поєднує реалістичну сучасну діяльність з синтетичною поведінкою атак у мережевому трафіку.

UNSW-NB15 був розроблений для стандартизованої оцінки систем виявлення вторгнень (NIDS) і покликаний замінити застарілі набори даних KDD Cup 99 та NSL-KDD, які вже не відповідають сучасним моделям мережевих атак [15].

Для збору 100 ГБ сирого мережевого трафіку використовувався інструмент tcpdump (рис. 1.5). Для вилучення атрибутів були застосовані інструменти Argus, Bro-IDS та розроблені 12 алгоритмів.

Він містить 2540000 наборів даних записів мережевого підключення. 49 функцій у кожному записі відповідають за ідентифікацію атак і класифікацію їх за одним із дев’яти типів: Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worm.

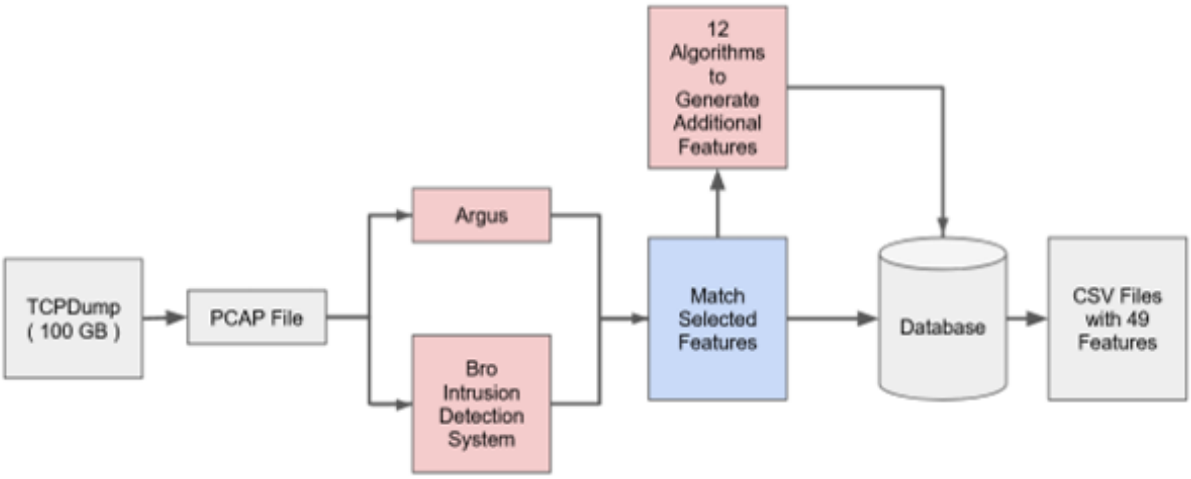

Рисунок 1.5 – Схема процесу створення набору даних UNSW-NB15

Найпопулярніші набори даних включають KDD Cup 99, NSL-KDD, CAIDA, DEFCON, ISCX, DARPA-2009 і UNSW-NB15 у таблиці 1.2.

Таблиця 1.2 – Порівняння найпопулярніших наборів даних

Набори даних	Реалі-стична конфігу-рація мережі	Реалі-стичний мережевий трафік	Мічені спостере-женнями	Повне захоп-лення взаємодії	Повне захоп-лення пакетів	Багато зловми-сних сценаріїв
KDD99	+	-	+	+	+	+
NSL-KDD	+	-	+	+	+	-
CAIDA	+	+	+	+	-	+
DEFCON	+	+	+	+	+	+
ISCX	+	+	+	+	_-	+
DARPA-2009	+	+	+	+	+	+
UNSW-NB15	+	+	+	+	+	+

Вибраний набір даних має включати функції, які базуються на реалістичній конфігурації мережі, мережевому трафіку, позначених спостереженнях, повному захопленні взаємодії та повному захопленні пакетів,

а також різноманітних шкідливих сценаріях. Було прийнято рішення використовувати набір даних UNSW-NB15 [16] для навчання та тестування моделі, яка має бути побудована.

1.5 Висновки та уточнення задач

На основі розглянутих вище методів виявлення та запобігання атак, система виявлення вторгнень у комп'ютерній мережі повинна відповідати наступним вимогам:

– система має виявляти атаки в режимі реального часу;

– виявляти загрози, застосовуючи методи аналізу аномалій;

– мати низьку ймовірність помилково-позитивних та помилково-негативних результатів;

– вміти розпізнавати нові види атак;

– відображати результати аналізу мережевого трафіку.

Для реалізації такої системи необхідно виконати наступні завдання:

– розробити модуль сенсору, який збирає дані про пакети мережевого трафіку в режимі реального часу;

– створити модуль обробки даних;

– реалізувати модуль аналізу, який базується на двох нейронних мережах;

– впровадити модуль сповіщення;

– навчити нейронні мережі за допомогою набору даних UNSW-NB15;

– провести тестування нейронних мереж на тестовій вибірці;

– провести аналіз результатів роботи системи виявлення атак.

2 ПРОЕКТУВАННЯ СИСТЕМИ ВИЯВЛЕННЯ АТАК

2.1 Архітектура системи NIDS

Класична архітектура системи виявлення мережевих вторгнень складається з чотирьох основних компонентів: сенсорного модуля, модуля обробки, модуля аналізу (що включає підмодулі для виявлення та класифікації атак), та модуля сповіщення, як показано на рисунку 2.1.

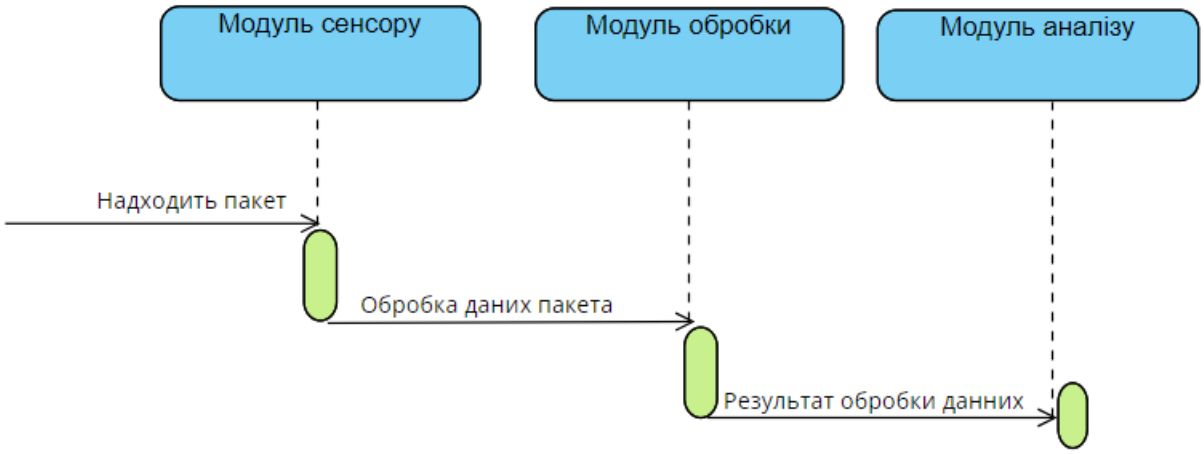

Рисунок 2.1 – Діаграма послідовності NIDS

Модуль сенсору функціонує як декодер пакетів, який отримує частини необробленого мережевого трафіку за допомогою інструментів збору даних аудиту, таких як Tcpdump і Libpcap. Ці інструменти передають кожну частину трафіку в препроцесор для подальшої обробки та аналізу.

Модуль обробки витягує набір атрибутів з необроблених даних аудиту, які потім використовуються в детекторі вторгнень.

Типовий препроцесор, широко застосовуваний у мережевих IDS, це TCP-препроцесор, що аналізує TCP-протоколи в контексті сеансових потоків.

Прикладами таких інструментів є Netflow, Bro-IDS та Argus, які перевіряють різні протоколи, такі як HTTP, DNS, SMTP та UDP.

Модуль аналізу отримує ці атрибути від модуля обробки і використовує модель, побудовану на основі каскадної структури нейронних мереж, яка дозволяє відрізняти атаки від нормальних пакетів. У разі виявлення атаки, він передає тривожне повідомлення модулю сповіщення, який піднімає тривогу.

Модуль сповіщення відповідає за генерацію попереджень, отриманих від модуля аналізу, та їх передачу адміністратору безпеки для подальшого реагування та вжиття заходів.

2.2 Модуль обробки даних мережі

Обробка даних є критично важливим етапом у створенні системи виявлення вторгнень, оскільки зібрані дані часто містять нерелевантну або дубльовану інформацію, яка може негативно вплинути на ефективність аналізу. Як і при зборі даних, який часто слабо контролюється, мережеві дані, отримані з мережевого трафіку, можуть включати зайві або нерелевантні значення.

Попередня обробка фільтрує ці дані, видаляючи зайву інформацію, що підвищує продуктивність системи виявлення вторгнень. Цей процес включає створення, скорочення, перетворення та нормалізацію атрибутів, як описано нижче.

Мережеві атрибути витягуються з необроблених мережевих пакетів за допомогою інструментів Argus, Bro-IDS, Netflow, Tcptrace та Netmate. Неможливо ефективно керувати NIDS без попереднього вилучення набору атрибутів, які часто називаються базовими атрибутами, як у наборах даних KDD99 і UNSW-NB15.

Додаткові атрибути встановлюються на основі ідентифікаторів транзакційного потоку, наприклад, IP-адреси джерела та пункту призначення, порти джерела та пункту призначення, тип протоколу, а також часу транзакційного з'єднання, наприклад, 10 або 100 з'єднань в секунду, для визначення потенційних характеристик мережевої поведінки. Ці атрибути є важливими для виявлення зловмисників, які можуть сканувати мережу нерегулярно, наприклад, одне сканування в хвилину або годину.

Щоб надати більш придатні дані для класифікатора нейронної мережі, набір даних проходить через кілька етапів попередньої обробки.

Видалення інформації про сокет: оскільки вихідний набір даних включає IP-адресу та номери портів хостів джерела та призначення в мережі, важливо видалити цю інформацію, щоб забезпечити неупереджене виявлення. Використання такої інформації може призвести до надмірної підготовки до інформації про сокет. Натомість класифікатор має навчатися на характеристиках самих пакетів, щоб будь-який хост з подібною інформацією про пакет фільтровувався незалежно від інформації про його сокети.

Кодування міток: мітки різних класів у наборі даних містять імена атак, які є строковими значеннями. Тому важливо закодувати ці значення в числові, щоб класифікатор міг розпізнавати номер класу, до якого належить кожен кортеж.

Нормалізація даних: числові дані в наборі даних мають різний діапазон, що створює проблеми для класифікатора під час навчання.

Таким чином, важливо нормалізувати значення в кожному атрибуті так, щоб мінімальне значення було нулем, а максимальне – одиницею. Це забезпечує більш однорідні значення класифікатора, зберігаючи відносність між значеннями кожного атрибута.

Головна перевага нормалізації полягає в усуненні зміщення з необроблених даних без зміни їхніх статистичних властивостей. Зазвичай використовується лінійне перетворення, яке описується рівнянням (2.1).

$$X_{normalised} = (X - min(X))/(max(X) - min(X)) \qquad (2.1)$$

В результаті виходить вектор з 176 атрибутами.

2.3 Модуль аналізу виявлення вторгнень

Модуль аналізу в системі виявлення вторгнень включає два основні компоненти: модуль виявлення аномалій та модуль класифікації. В обох

модулях використовуються багатошарові нейронні мережі, які натреновані на вирішення конкретних задач: виявлення аномалій або класифікації атак.

Для першої задачі, всі типи атак обробляються як наявність атаки, що спрощує задачу класифікації, перетворюючи її в задачу прогнозування. Для цього використовується гібридна архітектура нейронної мережі, яка поєднує шари згорткової та рекурентної нейронної мережі.

Друга задача передбачає класифікацію атак. Оскільки перший модуль вирішує питання наявності атаки, а нейронні мережі можуть давати хибні спрацьовування, друга мережа додатково перевіряє результати першої на наявність або відсутність атаки. Це дає можливість знизити ймовірність хибнопозитивних результатів. Для цієї задачі обрана згорткова нейронна мережа, яка виявилася ефективнішою для класифікації з багатьма класами. На рисунку 2.1 представлена діаграма послідовності NIDS.

Архітектура CNN, яка використовується в обох мережах, складається з вхідного шару, що приймає вектор з 176 атрибутів, двох послідовних комбінацій шару згортки та шару об’єднання і повного шару з'єднання. У першій комбінації шар згортки з невеликим ядром використовується для вилучення локальних особливостей мережевого трафіку, таких як IP та порт. Друга комбінація з великим ядром аналізує взаємозв'язки між віддаленими бітами. На шарі згортки застосовується функція активації ReLU, яка має дві основні переваги: простоту обчислень і сприяння вираженню складних ознак, див. вираз (2.2).

$$ReLU(x) = \begin{cases} 0, \text{для } x < 0 \\ 1, \text{для } x \geq 0 \end{cases} \quad (2.2)$$

Для гібридної архітектури після шарів згорткової мережі використовується рекурентна нейронна мережа з довгою короткостроковою пам’яттю (LSTM). Спочатку «шар фільтра забування» визначає, яку інформацію можна забути або залишити, див. вираз (2.3).

$$f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f) \quad (2.3)$$

Значення попереднього виходу і поточного входу пропускаються через сигмоїдальний шар, значення якого знаходяться в діапазоні [0; 1]. Далі визначається, яка нова інформація буде зберігатися в стані осередка, з використанням двох частин, див. вирази (2.4, 2.5). Сигмоїдальний шар визначає, які значення слід оновити, а tanh-шар будує вектор нових значень-кандидатів, які можна додати в стан осередка.

$$i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i) \quad (2.4)$$

$$\tilde{C}_t = tanh(W_C \times [h_{t-1}, x_t] + b_C) \quad (2.5)$$

Для заміни старого стану осередка на новий необхідно виконати операції, що описані у виразі (2.6).

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \quad (2.6)$$

На останньому етапі визначається, яка інформація буде отримана на виході, з використанням фільтрів, див. вираз (2.7, 2.8).

$$o_t = \sigma(W_o \times [h_{t-1}, x_t] + b_o) \quad (2.7)$$

$$h_t = o_t \times tanh(C_t) \quad (2.8)$$

Для налаштування параметрів моделі використовується алгоритм зворотного розповсюдження помилки. Як функція витрат застосовується категоріальна перехресна ентропія.

Для задачі виявлення атаки використовується бінарна перехресна ентропія. На виході нейронної мережі модуля виявлення атаки застосовується сигмоїдальна функція, див. вираз (2.9), яка видає значення від 0 до 1. На виході

нейронної мережі модуля класифікації атак застосовується функція Softmax, див. вираз (2.10), що формує вектор з ймовірностями належності пакету до певного класу: Fuzzers, Analysis, Backdoor, DoS, Exploits, Generic, Reconnaissance, Shellcode, Worm або Normal.

$$Sigmoid(x) = \frac{1}{1 + e^{x}} \tag{2.9}$$

$$Softmax(z_j) = \frac{e^{z_j}}{\sum_{k=1}^{K} \dots e^{z_k}} \tag{2.10}$$

Таким чином, побудовано дві нейронні мережі: одна для виявлення атак, а друга для їх класифікації, що разом утворюють ефективну каскадну систему нейронних мереж.

2.4 Набір даних для навчання нейронних мереж

Аналіз мережевого трафіку для визначення актуальних загроз базується на зборі даних з існуючих мережевих з’єднань. Ці дані формують характеристичний вектор перевірюваного зв’язку, який передається до модуля виявлення атак. Збір даних здійснюється за протоколами TCP, UDP та ICMP.

Очікуваний вихідний вектор модуля має формат UNSW-NB15, що включає різні атрибути, які поділяються на п'ять груп (табл. 2.1):

– поточні атрибути: включають ідентифікатори між хостами, наприклад, клієнт-сервер або сервер-клієнт;

– базові атрибути: представляють з'єднання протоколів;

– атрибути вмісту: інкапсулюють атрибути TCP/IP та http-послуг;

– атрибути часу: містять час прибуття між пакетами, час початку/закінчення пакету та час зворотного зв’язку протоколу TCP;

– додаткові генеровані атрибути: поділяються на атрибути загального призначення та атрибути підключення.

Таблиця 2.1 – Групи атрибутів, які має вихідний вектор

Номер	Назва	Опис
Поточні атрибути		
1	srcip	IP-адреса джерела
2	sport	Номер порту джерела
3	dstip	IP-адреса призначення
4	dsport	Номер порту призначення
5	proto	Тип протоколу (наприклад, TCP, UDP)
Базові атрибути		
6	state	Показує стан і залежний від нього протокол (ACC, CLO)
7	dur	Запис загальної тривалості
8	sbytes	Байтів від джерела до призначення
9	dbytes	Байтів від призначення до джерела
10	sttl	Час життя пакета даних від джерела до призначення
11	dttl	Час життя пакета даних від призначення до джерела
12	sloss	Джерельні пакети повторно передані або скинуті
13	dloss	Пакети призначення повторно передані або скинуті
14	service	Такі як http, ftp, smtp, ssh, dns та ftp-дані
15	sload	Біти джерела в секунду
16	dload	Біти призначення в секунду
17	spkts	Кількість пакетів джерела до призначення
18	dpkts	Кількість пакетів призначення до джерела

Продовження таблиці 2.1

Номер	Назва	Опис
Атрибути вмісту		
19	swin	Значення вікна TCP джерела
20	dwin	Значення вікна TCP призначення
21	stcpb	Базовий порядковий номер TCP джерела
22	dtcpb	Базовий порядковий номер TCP призначення
23	smeansz	Середнє значення розміру потоку пакетів, переданого джерелом
24	dmeansz	Середнє значення розміру потоку пакетів, переданого призначенням
25	trans_depth	Представляє конвеєрну глибину підключення транзакції запиту / відповіді http
26	res_bdy_len	Фактичний розмір нестисненого вмісту даних, переданих із серверної служби http
Атрибути часу		
27	sjit	Джиттер джерела (мСек)
28	djit	Джиттер призначення (мСек)
29	stime	Запис часу початку
30	ltime	Запис часу кінця
31	sintpkt	Час прибуття між пакетами джерела (мСек)
32	dintpkt	Час прибуття між пакетами призначення (мСек)
33	tcprtt	Час налаштування з'єднання TCP, сума 'synack' і 'ackdat'
34	synack	Час встановлення з'єднання TCP, час між пакетами SYN та SYN_ACK
35	ackdat	Час налаштування підключення TCP.

Продовження таблиці 2.1

Номер	Назва	Опис
Додаткові генеровані атрибути		
36	is_sm_ips_ports	Якщо srcip (1) дорівнює dstip (3), а sport (2) дорівнює dsport (4), ця змінна присвоює 1, інакше 0
37	ct_state_ttl	Номер для кожного стану (6) відповідно до конкретного діапазону значень sttl (10) та dttl (11)
38	ct_flw_http_mthd	Кількість потоків, які мають такі методи, як Get та Post в службі http
39	is_ftp_login	Якщо доступ до сеансу ftp здійснюється за допомогою користувача та пароля, тоді 1 ще 0
40	ct_ftp_cmd	Кількість потоків, яка має команду в сеансі ftp
41	ct_srv_src	Кількість записів, що містять одну і ту ж службу (14) та srcip (1), у 100 записах відповідно до ltime (30)
42	ct_srv_dst	Кількість записів, що містять ту саму послугу (14) та dstip (3) у 100 записах відповідно до ltime
43	ct_dst_ltm	Кількість записів того самого dstip (3) у 100 записах відповідно до ltime (30)
44	ct_src_ltm	Кількість записів того самого srcip (1) у 100 записах відповідно до ltime (30)
45	ct_src_dport_ltm	Кількість записів того самого srcip (1) та dsport (4) у 100 записах відповідно до ltime (30)
46	ct_dst_sport_ltm	Кількість записів одного і того ж dstip (3) та виду sport (2) у 100 записах відповідно до ltime (30)
47	ct_dst_src_ltm	Кількість записів того самого srcip (1) та dstip (3) у 100 записах відповідно до ltime (30)

Виявлення аномалій – це процес ідентифікації відхилень від нормального профілю системи. Для цього використовується бінарна класифікація, де критерієм є аномальність з'єднання (0 – нормальний профіль, 1 – аномалія). Набір даних UNSW-NB15 включає різноманітні спостереження за нормальними та аномальними поведінками [17].

Події безпеки в цьому наборі поділяються на 9 типів:

1) DoS: атака, що блокує доступ до комп'ютерних ресурсів, надсилаючи непотрібні запити, які перевантажують систему. Відмінність між DoS та DDoS полягає в кількості з’єднань: DoS використовує одне, а DDoS – багато;

2) Exploits: використовують вразливості у системі для отримання контролю над нею або для реалізації інших атак. Включають атаки нульового дня, які експлуатують ще невідомі вразливості;

3) Generic: атаки, спрямовані на всі блок-шифри для створення колізій, незалежно від їх конфігурації. Включають атаки на день народження, що використовують теорію ймовірностей;

4) Fuzzers: використовують генератори трафіку для виявлення вразливостей безпеки шляхом подання випадкових даних, що можуть спричинити збої у системі. Використовується для тестування проникнень і виявлення помилок, таких як переповнення буфера та SQL-ін’єкції;

5) Analysis: включає методи порушення Інтернет-програм через сканування портів, спам та веб-сценарії. Сканування портів дозволяє визначити відкриті порти і послуги на цільовій системі, тоді як спам поширює небажані повідомлення;

6) Backdoor: забезпечує несанкціонований віддалений доступ до системи, обходячи звичайну аутентифікацію. Після доступу зловмисник може викрадати дані або встановлювати шкідливе ПЗ;

7) Reconnaissance: збирають інформацію про мережу, використовуючи пінг-запити для виявлення активних IP-адрес і служб. Зловмисники можуть дізнатися про програмне забезпечення та операційні системи жертви;

8) Shellcode: малі фрагменти коду, що використовуються для запуску

командної оболонки та надання доступу зловмиснику до системи жертви. Зазвичай вводяться через вразливості переповнення буфера;

9) Worm: шкідливі програми, що самостійно розмножуються в мережі та використовують вразливості системи для поширення. Можуть викрадати дані або видаляти файли, перевантажуючи ресурси мережі.

Таким чином, процес виявлення аномалій у мережевому трафіку базується на детальному аналізі атрибутів і класифікації подій для ефективного виявлення та запобігання загрозам.

2.5 Обґрунтування обраних засобів програмної розробки

Для розробки системи вибрано мову програмування Python через кілька ключових причин. По-перше, Python має потужний набір бібліотек для штучного інтелекту та машинного навчання, таких як TensorFlow, Scikit-Learn та Keras, які полегшують реалізацію технічних рішень. Для візуалізації даних використовуються бібліотеки Scipy, Seaborn, Pandas та NumPy, що робить Python надзвичайно корисним для роботи з даними.

Python також є платформно-незалежною мовою програмування, що дозволяє розробникам створювати програмні продукти на одному пристрої та використовувати той самий код на будь-якій іншій платформі без необхідності вносити зміни. Це робить Python ідеальним для розробки додатків, які можуть працювати на різних операційних системах, таких як Windows, Mac та Linux. Крім того, Python дозволяє використовувати графічні процесори для пришвидшення навчання моделей машинного навчання.

Для реалізації нейронних мереж використовується бібліотека Keras, яка є частиною TensorFlow. TensorFlow, розроблена компанією Google, є відкритою бібліотекою для машинного навчання, яка призначена для побудови і тренування нейронних мереж, що автоматично знаходять та класифікують образи, досягаючи рівня якості людського сприйняття. TensorFlow широко застосовується у комерційних продуктах Google.

Keras, написана на Python, є високорівневою бібліотекою для роботи з нейронними мережами, яка спрощує процес їх створення та навчання. Вона є надбудовою над такими фреймворками, як Deeplearning4j, TensorFlow та Theano. Keras була створена для забезпечення швидкої і ефективної роботи з мережами глибокого навчання [18]. Основним розробником і підтримувачем Keras є Франсуа Шолле, інженер Google. Порівняння Keras з іншими популярними бібліотеками для побудови нейронних мереж, такими як TensorFlow та Pytorch, наведено в таблиці 2.2.

Таблиця 2.2 – Порівняння найбільш популярних бібліотек для формування нейронних мереж

	Keras	Pytorch	TensorFlow
API Рівень	високий	низький	високий та низький
Архітектура	простий, стислий, читабельний	складний, менш читабельний	непростий у використанні
Відладка	проста побудова мережі, тому відладка часто не потрібна	хороші можливості відладки	важко провести відладку
Популярність	найбільш популярний	третій за популярністю	другий за популярністю
Швидкість	повільний, низька ефективність	швидкий, висока ефективність	щвидкий, висока ефективність
Засоби реалізації	Python	Lua	C++, CUDA, Python

Проект використовує базу даних у форматі CSV, яка містить усі навчальні вектори UNSW-NB15, що значно прискорює процес навчання нейронних мереж.

Для захоплення та аналізу мережевих пакетів використовується консольна утиліта TShark, що входить до складу Wireshark. Вона дозволяє

виконувати захоплення пакетів у мережі, зберігати їх у файлі та переглядати раніше збережені дані. TShark використовує формат libpcap, який також використовується в tcpdump та інших інструментах для аналізу мережевих пакетів [19].

Argus є інструментом для аудиту з відкритим кодом, розробленим Картером Буллардом для аналізу необроблених мережевих пакетів та створення з них мережевих функцій. Argus складається з сервера, який отримує пакети, захоплені мережевими інтерфейсами пристрою, та перетворює їх у двійковий формат, і клієнтів, які можуть читати ці двійкові потоки і зберігати їх у базі даних [20].

Таким чином, поєднання Python та його бібліотек, разом з інструментами для збору та аналізу даних, забезпечує ефективну платформу для розробки системи виявлення вторгнень.

3 РОЗРОБКА СИСТЕМИ ВИЯВЛЕННЯ АТАК

Модулі системи виявлення аномалій розроблена на мові Python. Для реалізації нейронних мереж обрані такі бібліотеки як Tensorflow та Keras.

3.1 Модуль сенсору

Модуль сенсору розроблений саме для зв'язка та діалогом з утилітою Tshark. Програма має наступну задумку – при її запуску користувач має обрати інтерфейс на якому буде проходити аналіз мережевих даних, наступним кроком є вибір інтервал часу протягом якого буде перехоплювати трафік. Перехоплення трафіку відбувається у фоновому режимі, а результат зберікається у файл з розширенням pcap. Ця процедура відбувається кожній заданий період часу, який вказав користувач. Реалізація модулю сенсору наведена у додатку А.

3.2 Модуль обробки

Модуль обробки виконує функцію підготовки та обробки даних перед передачею її на наступний модуль – модуль аналізу. Файли , де зберігаються дані пакетів, формату pcap та argus, з якого буде вилучено корисні дані, система створює та зберігає у директорії за замовчуванням. Наступним кроком системи буде збереження файлу csv формат для подання його на вхід обученій нейронній мережі. Реалізація модуля обробки наведена у додатку А.

Обробка даних відбувається у чотири етапу: генерація додаткових атрибутів; зменшення наявного набору даних; кодування символьних даних; нормалізація даних.

Генерація додаткових атрибутів включає в себе генерування таких атрибутів як: is_sm_ips_ports, ct_state_ttl, ct_dst_ltm, ct_src_ltm, ct_src_dport_ltm, ct_dst_sport_ltm, ct_dst_src_ltm. На першому кроку знакходится генерація

атрибута is_sm_ips_ports: якщо srcip (1) дорівнює dstip (3), а sport (2) дорівнює dsport (4), ця змінна присвоює 1, інакше 0. Наступним – ct_state_ttl. Для ct_state_ttl номер для кожного state (6) відповідно до конкретного діапазону значень sttl (10) та dttl (11). Останнім етапом в генерації атрибутів є генерація залишившися атрибутів:

– ct_dst_ltm –записи dstip (3) у 100 записах ltime (30);

– ct_src_ltm – кількість записів того самого srcip (1) у 100 записах ltime (30);

– ct_src_dport_ltm – кількість записів того самого srcip (1) та dsport (4) у 100 записах до ltime (30);

– ct_dst_sport_ltm – кількість записів одного і того ж dstip (3) та виду sport (2) у 100 записах ltime (30);

– ct_dst_src_ltm – кількість записів того самого srcip (1) та dstip (3) у 100 записах ltime (30).

Наступним етапом функціонування модулю являє собою зменшення набору даних методом вибору тільки тих характеристик, які будуть використані у модулі виявлення вторгнень.

Кодування символьних даних відбувається з використанням моделі OneHotEncoder який теж використовувався на начальному наборі даних.

Останнім кроком є нормалізація данних. На даному етапі використовують мінімальні та максимальні значення отриманих на начальному наборі даних атрибутів.

3.3 Модуль аналізу

Для побудови та реалізації цього модуля реалізовані дві нейронні мережи, які є незалежними одна від одної. Ці мережі включають в себе мережу яка заточена під класифікацію атаки та під її прогнозування, схеми мереж надані у додатку Б та додатку В відповідно.

Обробка даних з набору даних UNSW-NB15 для навчання обох моделей була практично схожою. Перший етап являє собою поміщення даних до csv

файлу в об’єкт який має тип DataFrame. Наступним кроком є відокремлення цільового атрибуту від інших атрибутів з його набору та видалення усіх інших атрибутів, які не будуть брати участь в навчанні нейронної мережі. Для двох мереж відокремлюється lable та attack_cat відповідно. Реалізація нейронних мереж наведена у додатку Г.

Кодування в числові атрибути шляхом використовування OneHotEncoder для атрибутів state і proto є наступним етапом перед навчанням мереж. Після чого модель кодувальника зберігається для подальшого використання для перетворення даних.

Наступним кроком є нормалізація даних. Варто зауважити, що для подальшого процесу нормалізації даних мінімальні та максимальні значення кожного з атрибутів зберігатимуться в окремому файлі. Саме це зроблено для повторення процесу реалізації над даними з реальних пакетів.

Далі відбувається на розділення на тестову та навчальну, а також відокремлення від навчальної вибірки частину даних у нову вибірку для обчислення помилки під час навчання.

Перша модель нейронної мережі, що виконує завдання на виявлення у мережі аномалій. Ця модель використовує функцію витрат nadam, яка використовується у завданнях двійкової класифікації.

Друга модель нейронної мережі зосереджена на завдання класифікації атаки. Для її навчання використано оптімізатор Adam вже з категоріальною кроссентропією в якості функції витрат.

3.4 Навчання та тестування нейронних мереж

Ефективність виявлення вторгнень (IDS) залежить від правильної побудови матриці неточностей (табл. 3.1), яка застосовується для будь-якого завдання класифікації. Розмір матриці визначається кількістю класів у наборі даних.

Мета матриці полягає в порівнянні фактичних і прогнозованих міток.

Якщо в завданні виявлення вторгнень є два класи – нормальні записи і записи атак, матриця має розмір 2 на 2.

Таблиця 3.1 – Матриця неточностей

		Фактичні	
		Негативні	Позитивні
Предсказані	Негативні	TN	FP
	Позитивні	FN	TP

Побудована для першої нейронної мережі матриця неточностей зображена на рисунку 3.1.

TP (істинно позитивний) та TN (істинно негативний) позначають корректно класифіковані умови, у той час як FP (хибно позитивний) та FN (хибно негативний) – неправильні. TP і TN відповідають правильно класифікованим записам атак і нормальних записів, відповідно, а FP і FN – неправильно класифікованим нормальним записам і атакам, відповідно [21]. Ці терміни використовуються для створення показників оцінки роботи IDS.

Точність (Accuracy) – це показник, що відображає загальний відсоток правильних виявлень і помилкових тривог, який відображає загальну ймовірність успіху IDS, див. вираз (3.1).

$$Accuracy = (TN + TP)/(TP + FP + TN + FN) \quad (3.1)$$

Значення показників точності та функції потері на нейронній мережі модуля виявлення атаки під час тренування наведені на рисунках 3.2 і 3.3.

Рівень виявлення (DR), також відомий як істинно позитивний коефіцієнт (TPR), визначає частку правильно класифікованих шкідливих записів від загального числа шкідливих записів, див. вираз (3.2):

$$DR = TP/(FN + TP) \quad (3.2)$$

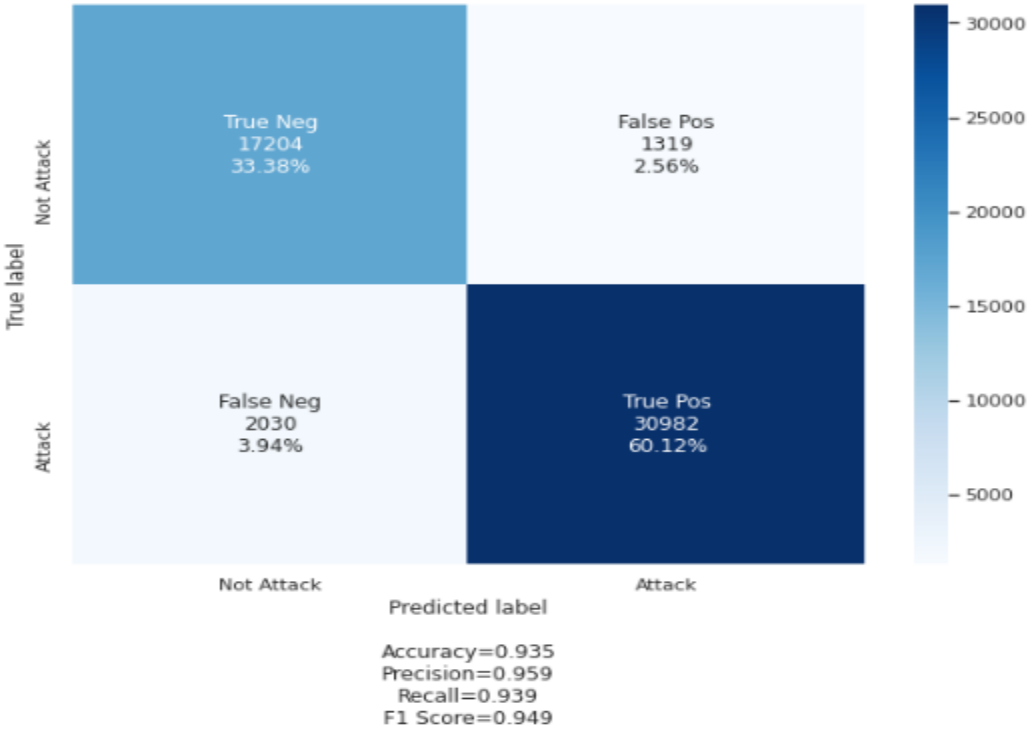

Рисунок 3.1 – Матриця неточностей нейронної мережі модуля виявлення атак

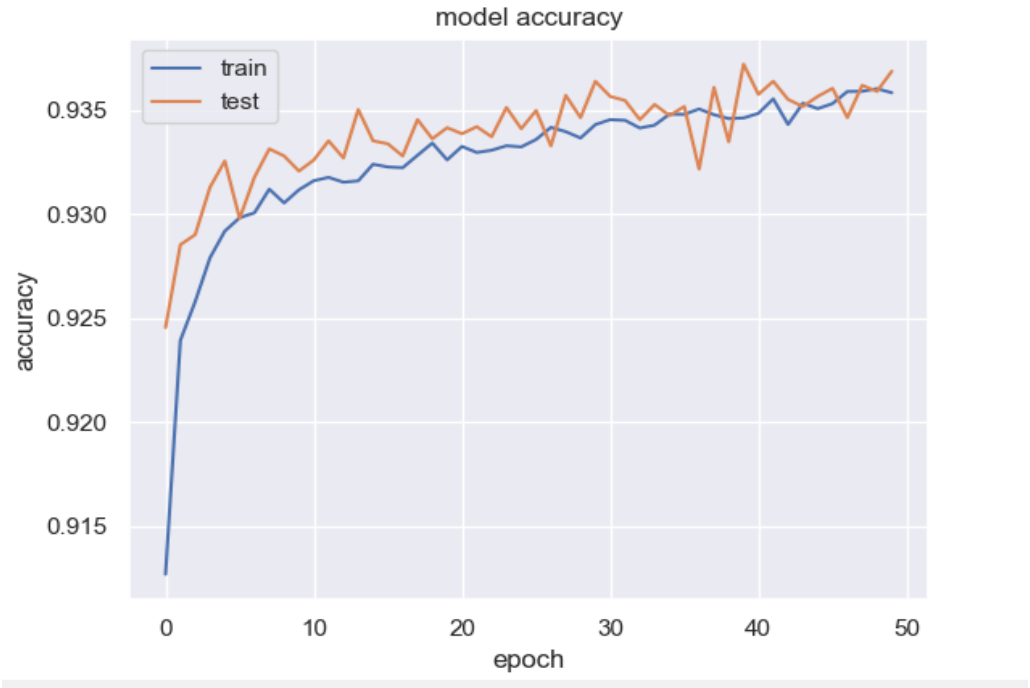

Рисунок 3.2 – Значення показників точності Accuracy під час тренування нейроної мережі модуля виявлення атаки

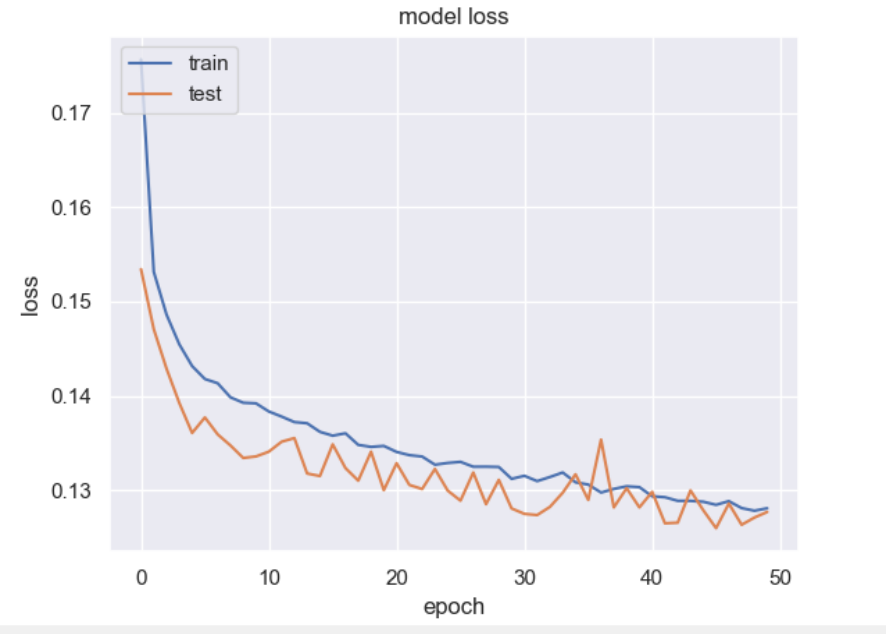

Рисунок 3.3 – Значення функції втрат Loss під час тренування нейроної мережі модуля виявлення атаки

Значення показників точності та функції потері на нейронній мережі модуля класифікації атаки під час тренування наведені на рисунках 3.4 і 3.5.

Рисунок 3.4 – Значення показників точності Accuracy під час тренування нейроної мережі модуля класифікації атаки

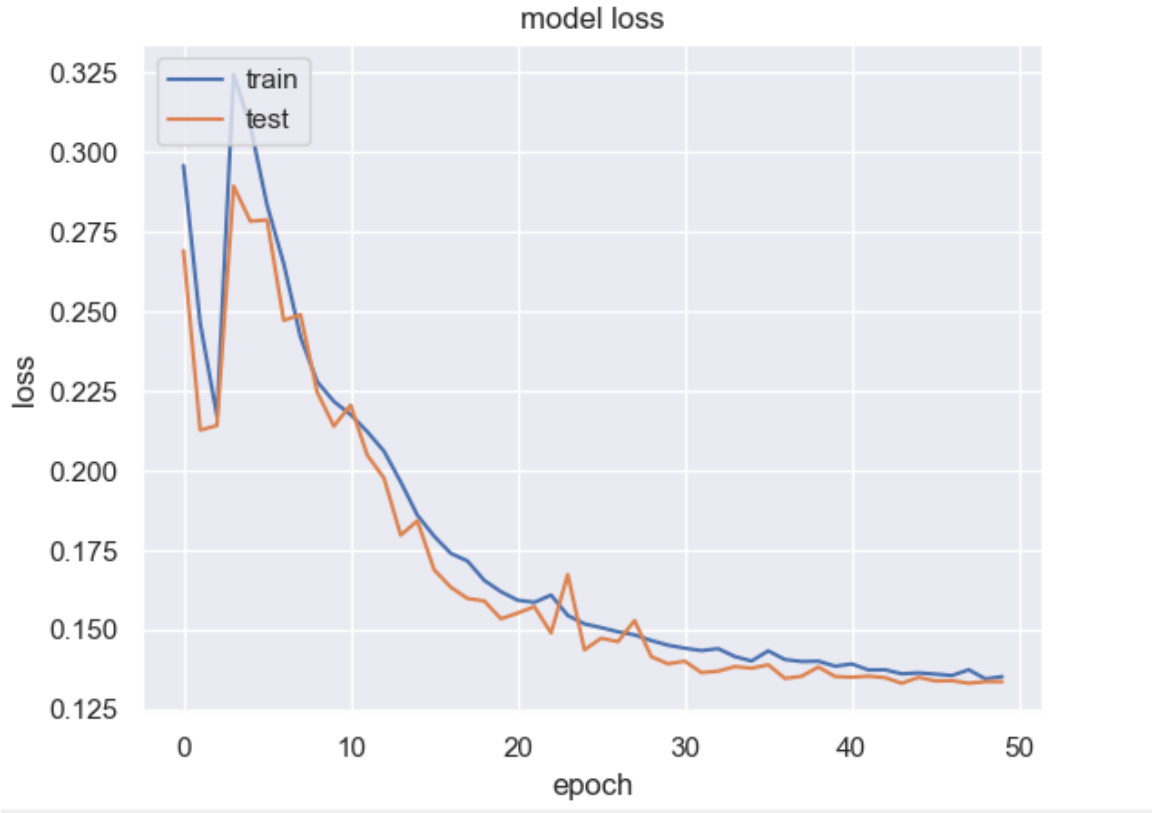

Рисунок 3.5 – Значення показників функції втрат під час тренування нейроної мережі модуля класифікації атаки

Істинно негативний коефіцієнт (TNR), або специфічність, визначає частку правильно класифікованих нормальних записів від загального числа нормальних записів, див. вираз (3.3).

$$TNR = TN/(TN + FP) \tag{3.3}$$

Коефіцієнт помилкових спрацьовувань (FPR) визначає частку нормальних записів, неправильно класифікованих як атаки, див. вираз (3.4).

$$FPR = FP / (FP + TN) \tag{3.4}$$

Хибно негативний коефіцієнт (FNR) визначає частку неправильно класифікованих атак від загального числа атак, див. вираз (3.5).

$$FNR = FN / (FN + TP) \quad (3.5)$$

Робота IDS оцінюється за допомогою показників TPR-FPR або специфічності-чутливості, щоб оцінити їхню точність у виявленні шкідливих дій [22]. Значення показників роботи IDS наведені в таблиці 3.2.

Таблиця 3.2 – Значення коефіцієнтів модуля виявлення атаки

	DR	TNR	FPR	FNR
Нейронна мережа модуля виявлення атаки	90,86%	89,78%	9,21%	8,17%

Критерій F-міри є найкращою мірою оцінки IDS. Це гармонійне середнє значення Precision і Recall [23], див. вираз (3.6).

$$F - мipa = 2 * (Precision * Recall) / (Precision + Recall), \quad (3.6)$$

де Precision – це частка передбачених позитивних значень, які насправді є позитивними;

Recall – частка правильно виявлених позитивних значень.

$$Precision = TP / (TP + FP) \quad (3.7)$$

$$Recall = TP / (TP + FN) \quad (3.8)$$

Значення критерія F-міри нейроної мережі модуля виявлення атаки та нейроної мережі модуля класифікації атаки під час тренування зображені відповідно на рисунках 3.6 і 3.7.

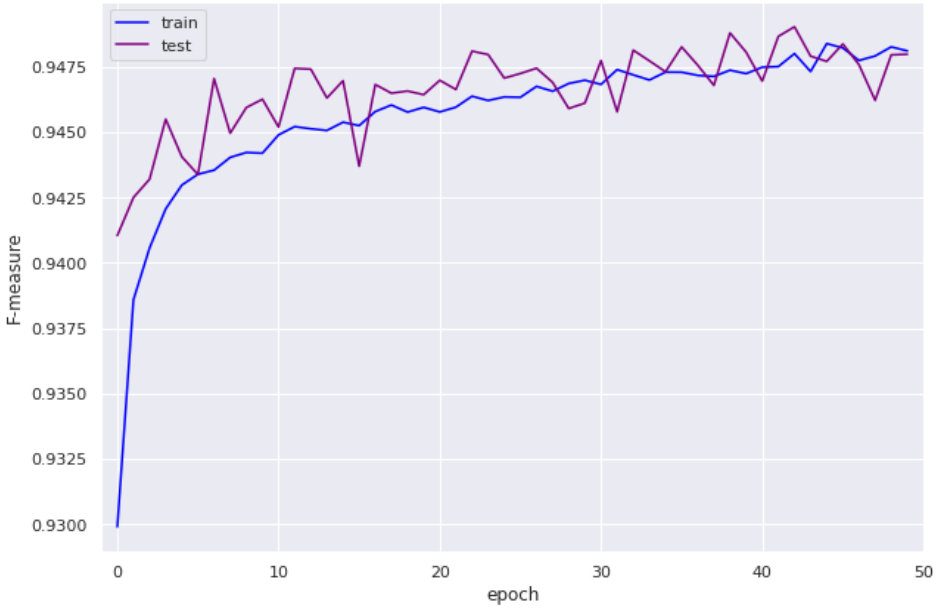

Рисунок 3.6 – Значення критерія F-міри під час тренування нейроної мережі модуля виявлення атаки

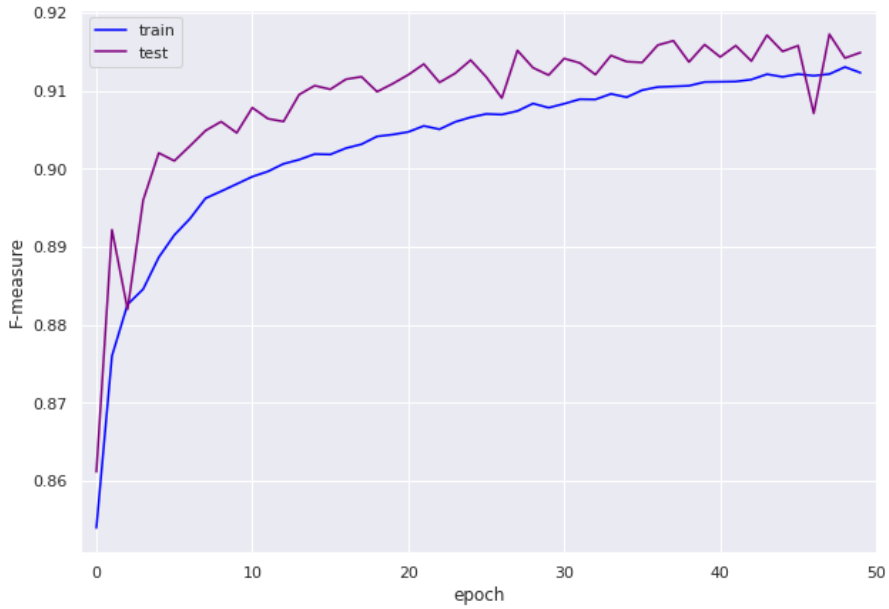

Рисунок 3.7 – Значення критерія F-міри під час тренування нейроної мережі модуля класифікації атаки

ВИСНОВКИ

У ході виконання кваліфікаційної роботи спроектована система, призначена для аналізу мережевого трафіку з метою виявлення атак. Для цього проведений аналіз існуючих систем виявлення вторгнень та методів виявлення атак, що дозволило зробити висновок про доцільність реалізації даного проекту для вирішення основних проблем існуючих систем.

В роботі розроблени три основні модулі: сенсорний модуль, модуль обробки даних, модуль аналізу. Для виявлення атак використовувався метод виявлення аномальної поведінки за допомогою глибокої нейронної мережі.

Модуль аналізу включає дві частини: модуль виявлення атаки та модуль класифікації атаки. Модель використовує дві нейронні мережі: перша перевіряє наявність атаки, а друга класифікує атаку у разі її виявлення. Для виявлення аномалій використана гібридна нейронна мережа, що складається з шарів згорткової нейронної мережі та рекурентної нейронної мережі з довготривалою короткочасною пам'яттю. Для класифікації атак використовувалася згорткова нейронна мережа, оскільки вона показала високу ефективність у задачах класифікації.

Для навчання нейронних мереж використовувався набір даних UNSW-NB15, який містить мережевий трафік, класифікований у 9 різних категорій атак. Для визначення ефективності конфігурації використовувалися спеціалізовані метрики, що застосовуються для тестування сучасних систем виявлення вторгнень (IDS).

Нейро протестована на тестовій вибірці з набору UNSW-NB15, що включає більше 50 тисяч векторів.

Подальший розвиток системи може здійснюватися в кількох напрямках. Перш за все, інтеграція системи виявлення атак у систему запобігання атак. Крім того, варто працювати над вдосконаленням системи, наприклад, додати можливість збереження результатів аналізу трафіку у файл або базу даних.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Названа кількість кібератак в Україні за минулий рік. [Електронний ресурс] – Режим доступу: https://www.slovoidilo.ua/2024/01/31/novyna/suspilstvo/nazvana-kilkist-kiberatak-ukrayini-mynulyj-rik
2. Колодчак О.М., 2012. [Електронний ресурс] – Режим доступу: https://science.lpnu.ua/sites/default/files/journal-paper/2017/nov/6726/16-98-104.pdf
3. Soumyadeep Hore,Jalal Ghadermazi,Ankit Shah,Nathaniel D. Bastian Deep PackGen: A Deep Reinforcement Learning Framework for Adversarial Network Packet Generation [Електронний ресурс] – Режим доступу: https://doi.org/10.48550/arXiv.2305.11039
4. A sequential deep learning framework for a robust and resilient network intrusion detection system. vol. 1 - 2024 - [Електронний ресурс] – Режим доступу: https://www.sciencedirect.com/science/article/pii/S0167404824002311
5. Stephen Cooper -2021-2024 [Електронний ресурс] – Режим доступу:https://securitymedia.org/analytics/nids.html
6. Anatomy of a Thread Module (TmModule) [Електронний ресурс] – Режим доступу: https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Thread_Modules
7. Zeek Documentation [Електронний ресурс] – Режим доступу: https://docs.zeek.org/en/current/
8. OSSEC Documentation [Електронний ресурс] – Режим доступу: https://www.ossec.net/docs/ (Daniel Cid et al., OSSEC Documentation, 2021)
9. Snort 3 Rule Writing Guide [Електронний ресурс] – Режим доступу: https://docs.snort.org/ (Cisco Systems, Inc., Snort 3 Rule Writing Guide, 2020)
10. Khraisat, A., Gondal, I., Vamplew, P., & Kamruzzaman, J. (2018). Survey of intrusion detection systems: techniques, datasets and challenges. Cybersecurity. [Електронний ресурс] – Режим доступу: https://cybersecurity. springeropen.com/articles/10.1186/s42400-019-0038-7

11. Hu, W., & Gao, J. (2019). Anomaly-based intrusion detection using statistical techniques. Journal of Cyber Security Technology, 3(2), C.123-145. doi:10.1080/23742917.2018.
12. Farady, I., Kuo, C., Ng, H., & Lin, C. (2023). Hierarchical Image Transformation and Multi-Level Features for Anomaly Defect Detection. Sensors, 23(2), C.988. [Електронний ресурс] – Режим доступу: https://www.mdpi.com/1424-8220/23/2/988
13. Wang, Y., Zhang, J., & Zhang, Q. (2021). Image/Video Deep Anomaly Detection: A Survey. arXiv preprint. [Електронний ресурс] – Режим доступу: https://arxiv.org/abs/2103.01739
14. Wu, Y., & Prasad, S. (2017). Convolutional Recurrent Neural Networks for Hyperspectral Data Classification. Remote Sensing. [Електронний ресурс] – Режим доступу: https://www.mdpi.com/2072-4292/9/3/298
15. Sahu, S.K. A Detail Analysis on Intrusion Detection Datasets. In Proceedings of the 2014 IEEE International Advance Computing Conference (IACC). / Sahu, S.K.; Sarangi, S.; Jena, S.K. – 2014. – C. 1343.
16. Tavallaee M. A detailed analysis of the KDD CUP 99 data set, IEEE Symposium on Computational Intelligence for Security and Defense Applications, CISDA./ Tavallaee M., Bagheri E., Lu W., Ghorbani A. A. – 2009. – C. 1–6.
17. Moustafa, Nour. Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic. Diss. University of New South Wales, Canberra, Australia, 2017. [Електронний ресурс] – Режим доступу: https://research.unsw.edu.au/projects/unsw-nb15-dataset
18. Moustafa, Nour. Designing an online and reliable statistical anomaly detection framework for dealing with large high-speed network traffic. Diss. University of New South Wales, Canberra, Australia, 2017. [Електронний ресурс] – Режим доступу: https://unsworks.unsw.edu.au/entities/dataset/4dc0e35c-6196-4c9d-945a-c50b981e5955
19. Keras: The high-level API for TensorFlow [Електронний ресурс] – Режим доступу: https://www.tensorflow.org/guide/keras?hl=ru (François Chollet, Keras: The high-level API for TensorFlow, 2021)

20. Using TShark to watch and inspect network traffic – [Електронний ресурс] – Режим доступу: https://www.linuxjournal.com/content/using-tshark-watch-and-inspect-network-traffic (Linux Journal, Using TShark to watch and inspect network traffic, 2020).
21. Bullard, C. (2021). Argus Project Official Website. [Електронний ресурс] – Режим доступу: https://openargus.org/
22. Fairuz A. Evaluation of machine learning classifiers for mobile malware detection. Soft Computing. / Fairuz A., Feizollah A., Nor B., and Gani A. – 2016. – pp. 343–357.

ДОДАТОК А

Код модуля сенсору та модуля обробки

```
class SnifferModule:
     def init(self, _network_interface, _timeout,_pcap_filename):
self._network_interface = _network_interface self._timeout =
_timeout self._pcap_filename = _pcap_filename
     def run(self):
          self.capture_to_pcap()
     def capture_to_pcap(self):
          capture=pyshark.LiveCapture(interface=self._network_in
     terface, output_file=self._pcap_filename)
     capture.sniff(timeout=self._timeout) packets = [pkt for pkt
     in capture._packets] capture.close() capture.clear()
```

Лістинг А.1 – SnifferModule

```
class PreprocessingModule:
    def init(self,_pcap_filename,_argus_filename, _csv_filename,
_encoder_filename, _norm_filename):
        self._pcap_filename = _pcap_filename
        self._argus_filename = _argus_filename
        self._csv_filename = _csv_filename
        self._encoder_filename = _encoder_filename
        self._norm_filename = _norm_filename
    def run(self):
        self.from_pcap_to_argus()
        self.from_argus_to_csv()
        df = pd.read_csv(self._csv_filename)
        df = self.filter_csv(df)
        pd.set_option('display.max_columns', 30)
        pd.set_option('display.width', 1000)
        print(df.head())
```

Лістинг А.2 – Основні функції PreprocessingModule

```
    def from_argus_to_csv(self):
    command = subprocess.run(['ra',   '-L', '-1', '-c',',','-r',
self._argus_filename,'s', 'stime ltime '  + 'saddr daddr '  +
'sport dport '  + 'spkts dpkts '  + 'sbytes dbytes '  + 'rate
dur '  + 'proto state ' +'sttl dttl ' +    'sload dload ' +
'sloss dloss ' +   'sintpkt dintpkt ' +     'sjit djit ' + 'swin
dwin ' +   'stcpb dtcpb ' + 'tcprtt ' +    'synack ackdat ' +
'smeansz dmeansz'], capture_output=True )
    with open(self._csv_filename, 'w') as f:
    f.write('stime,ltime,'  + 'saddr,daddr,'  +
    'sport,dport,'  +    'spkts,dpkts,'  +   'sbytes,dbytes,'  +
    'rate,dur,'  + 'proto,state,' +   'sttl,dttl,' +
    'sload,dload,' +   'sloss,dloss,' +   'sinpkt,dinpkt,' +
    'sjit,djit,' +  'swin,dwin,' + 'stcpb,dtcpb,' + 'tcprtt,' +
    'synack,ackdat,' +
    'smean,dmean\n')f.write(command.stdout.decode('utf-8')
```

Лістинг А.2, аркуш 2

ДОДАТОК Б

Схема нейронної мережі, яка використовується для виявлення атаки

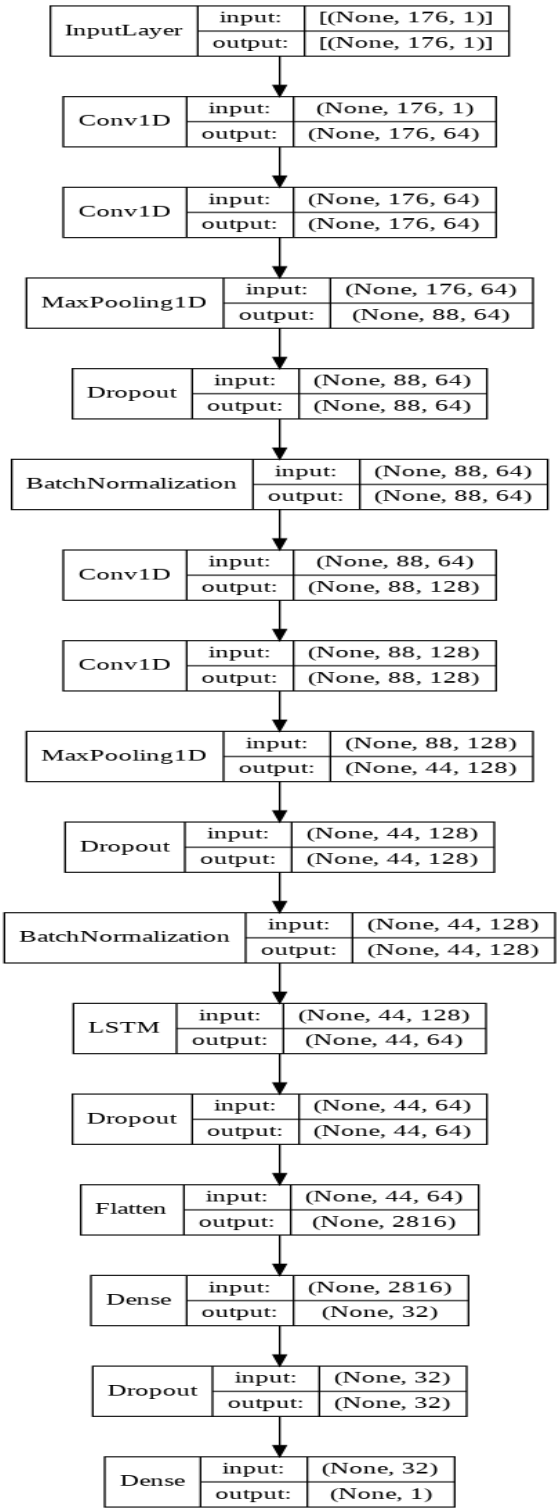

Рисунок Б.1 – Схема нейронної мережі, яка використовується в задачі виявлення атаки

ДОДАТОК В

Схема нейронної мережі, яка використовується для класифікації атаки

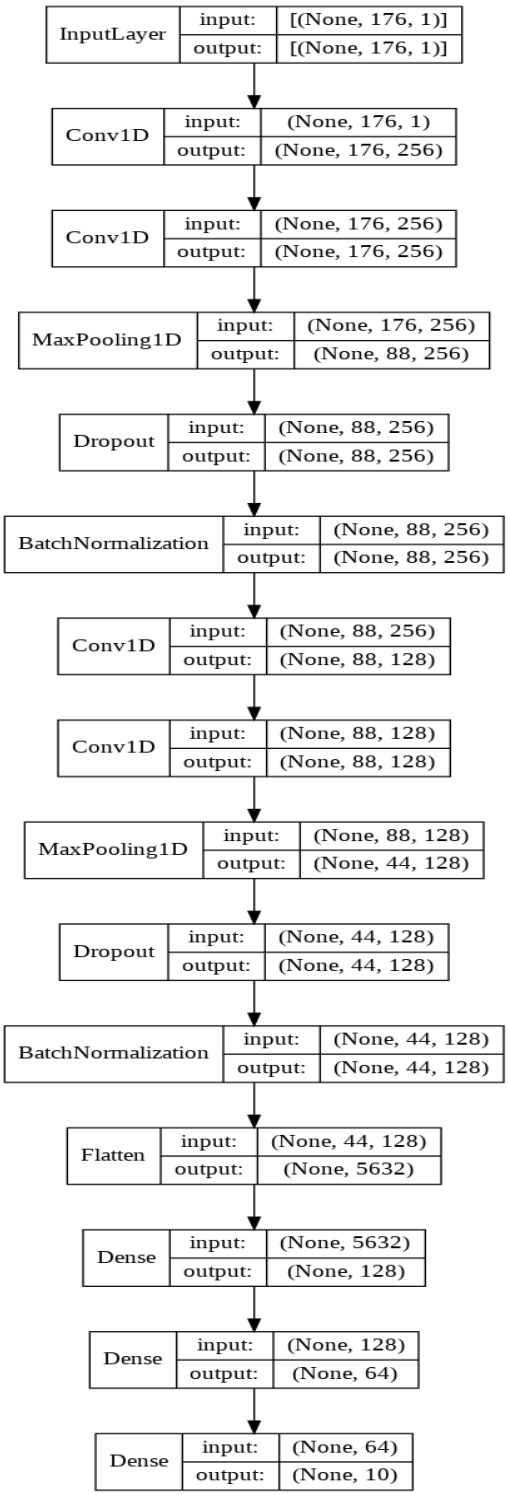

Рисунок В.1 – Схема нейронної мережі, яка використовується в задачі класифікації атаки

ДОДАТОК Г

Програмний код модуля аналізу

```
        def init_model(self):
            self.model = Sequential()
    self.model.add(Conv1D(64,2,activation='relu',padding='same',
kernel_initializer='he_uniform', input_shape=(176,1))
self.model.add(Conv1D(64,2,activation='relu',padding='same',kern
el_initializer='he_uniform'))
            self.model.add(MaxPooling1D(pool_size=2,strides=2))
            self.model.add(Dropout(0.2))
            self.model.add(BatchNormalization())
self.model.add(Conv1D(128,2,activation='relu',padding='same',ker
nel_initializer='he_uniform'))
self.model.add(Conv1D(128,2,activation='relu',padding='same',ker
nel_initializer='he_uniform'))
            self.model.add(MaxPooling1D(pool_size=2,strides=2))
    self.model.add(Dropout(0.2))
            self.model.add(BatchNormalization())
            self.model.add(Bidirectional(LSTM(units = 64,
return_sequences=True)))
            self.model.add(Dropout(0.2))
            self.model.add(Flatten())
            self.model.add(Dense(32, activation='relu'))
            self.model.add(Dropout(0.2))
            self.model.add(Dense(1, activation='sigmoid'))
            nadam = Nadam(lr=0.008, beta_1=0.9, beta_2=0.999,
epsilon=1e-08, schedule_decay=0.004)
            self.model.compile(loss =
"binary_crossentropy",optimizer = nadam, metrics = ["accuracy"])
        def load_weights(self):
            self.model.load_weights(self._weight_filename)
```

Лістинг Г.1 – Модель CNN для виявлення атак

```
    def init_model(self):
            self.model = Sequential()
self.model.add(Conv1D(256,2,activation='relu',padding='same',ker
nel_initializer='he_uniform', input_shape=(176,1)))
self.model.add(Conv1D(256,2,activation='relu',padding='same',ker
nel_initializer='he_uniform'))
            self.model.add(MaxPooling1D(pool_size=2,strides=2))
            self.model.add(Dropout(0.2))
            self.model.add(BatchNormalization())
self.model.add(Conv1D(128,2,activation='relu',padding='same',ker
nel_initializer='he_uniform'))
self.model.add(Conv1D(128,2,activation='relu',padding='same',ker
nel_initializer='he_uniform'))
            self.model.add(MaxPooling1D(pool_size=2,strides=2))
            self.model.add(Dropout(0.2))
            self.model.add(BatchNormalization())
            self.model.add(Flatten())
            self.model.add(Dense(128,activation='relu'))
            self.model.add(Dense(64,activation='relu'))
            self.model.add(Dense(10,activation='softmax'))
            self.model.compile(
            loss=tf.keras.losses.categorical_crossentropy,
                    optimizer=tf.keras.optimizers.Adam(),
                    metrics = ["accuracy"])
```

Лістинг Г.2 – Модель CNN для класифікації атак