Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

Пояснювальна записка

до магістерської кваліфікаційної роботи

на тему: «Інтелектуальна інформаційна технологія виявлення DDoS-атак»

08-20.МКР.009.00.000 ПЗ

Виконав: студент 2 курсу, групи 1 БС-18м
Спеціальності 125 – Кібербезпека
ОПП Безпека інформаційних та
комунікаційних систем

_____ Кульчицький Б. В.

Керівник: к. т. н., доц. каф. ЗІ

_____ Куперштейн Л. М.

Рецензент к. т. н., доц., доц. каф. ОТ

_____ Крупельницький Л. В.

Вінниця - 2019 року

Вінницький національний технічний університет

Факультет Інформаційних технологій та комп'ютерної інженерії
Кафедра Захисту інформації
Освітньо-кваліфікаційний рівень магістр
Спеціальність – 125 Кібербезпека

ЗАТВЕРДЖУЮ
Завідувач кафедри ЗІ, д.т.н., проф.
_____В. А. Лужецький
"____" _____2019 року

З А В Д А Н Н Я

НА КОМПЛЕКСНУ МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кульчицькому Богдану Володимировичу

1. Тема роботи: «Інтелектуальна інформаційна технологія виявлення DDoS-атак» керівник роботи: Куперштейн Леонід Михайлович, к. т. н., доц. каф. ЗІ, затверджені наказом № 254 ректора ВНТУ від 02.10.2019 р.

2. Строк подання студентом роботи 12 грудня 2019 р.

3. Вихідні дані до роботи:

- операційна система – Windows від 7 до Windows 10;
- для 32-х та 64-х бітних систем;
- різні типи DDoS-атак;
- обробка нейронними мережами;
- мова програмування – Python 3.7.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) Вступ. 1. Аналіз предметної області. 2. Розробка технічного проекту. 3. Розробка робочого проекту. 4. Економічна частина. Висновки. Список використаних джерел. Додатки.

5. Перелік ілюстративного матеріалу: типи DDoS-атак (плакат, А4); Складові інформаційної технології виявлення DDoS-атак (плакат, А4); Схема процесів інформаційної технології (плакат, А4); Схема формування набору даних (плакат, А4); Схема алгоритму перехоплення та аналізу пакетів (плакат, А4); Модуль інтелектуального аналізу (плакат, А4); матриця похибок рекурентної мережі LSTM (плакат, А4); фрагмент інтерфейсу додатку (плакат, А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Куперштейн Л.М., к.т.н., доц. каф.ЗІ		
2	Куперштейн Л.М., к.т.н., доц. каф.ЗІ		
3	Куперштейн Л.М., к.т.н., доц. каф.ЗІ		
4	Мацкевічус С. С. ст. вик. каф. ЕПВМ		

7. Дата видачі завдання _____ 2019 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2019 – 04.09.2019	
2	Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	05.09.2019 – 15.09.2019	
3	Науково-технічне обґрунтування	16.09.2019 – 22.09.2019	
4	Розробка технічного завдання	23.09.2019 – 29.09.2019	
5	Розробка рішень	30.09.2019 – 12.10.2019	
6	Практична реалізація, моделювання, експериментування, результати	14.10.2019 – 10.11.2019	
7	Розробка розділу економічного обґрунтування доцільності розробки	11.11.2019 – 17.11.2019	
8	Аналіз виконання ТЗ, висновки	18.11.2019 – 24.11.2019	
9	Оформлення пояснювальної записки	25.11.2019 – 30.11.2019	
10	Попередній захист та доопрацювання МКР	28.11.2019 – 01.12.2019	
11	Перевірка магістерської роботи на наявність плагіату	02.12.2019 – 10.12.2019	
12	Представлення МКР до захисту	11.12.2019 – 14.12.2019	
13	Захист МКР	16.12.2019 – 20.12.2019	

Студент _____ Кульчицький Б.В.
(підпис)

Керівник роботи _____ Куперштейн Л.М.
(підпис)

АНОТАЦІЯ

У роботі було проведено аналіз існуючих типів DDoS-атак. Проаналізовано підходи до захисту від DDoS-атак та інформаційного забезпечення, яке дозволяє реалізувати такого роду атаки. На основі цього аналізу було визначено необхідність розробки інтелектуальної інформаційної технології виявлення DDoS-атак та виконано постановку задачі дослідження. Запропоновано складові інформаційної технології які розділені на процеси. Визначено теоретичні оцінки підвищення стійкості даних методів. Представлено метод формування набору даних для інтелектуального аналізу. Досліджено існуючі роботи у цьому напрямку. Інформаційну технологію представлено у вигляді програмного опису мовою Python. За допомогою модулю TensorFlow змодельовано різні архітектури нейронних мереж та оцінено їх точність. Розроблено інтерфейс та проведено тестування програмного засобу. Визначено термін окупності програмного засобу.

ABSTRACT

The work analyzes the existing types of DDoS-attacks. The approaches to protection from DDoS-attacks and information software that allows you to implement this kind of attack. On the basis of this analysis the necessity of development of intelligent information technology for detecting DDoS attacks was determined and the research task was formulated. The components of information technology which are divided into processes are offered. Theoretical estimations of increase of stability of these methods are determined. The method of forming a data set for mining is presented. Existing works in this direction are investigated. Information technology is presented in the form of a software description in Python. The TensorFlow module simulated different neural network architectures and evaluated their accuracy. Interface developed and software tested. The payback period of the software has been determined.

ЗМІСТ

ВСТУП

З розвитком технологій збільшується і кількість кібератак. Особисті та корпоративні комп'ютерні системи можуть зіткнутися з різними типами загроз, які реалізуються зловмисники за допомогою інформації, як от використання її проти когось, крадіжка чи попросту її недоступність для звичайних користувачів. До останнього і відноситься мережева розподілена атака на відмова в обслуговуванні (DDoS) [1]. Реалізація якої може створити матеріальну та моральну шкоду як для людей, а ще більшу для компаній які надаються доступ певної інформації.

Одним з перспективних напрямів забезпечення безпеки мережі є використання засобів виявлення атак, побудованих на основі інтелектуальних технологій, а саме штучних нейронних мереж (НМ). На сьогодні ШНМ дозволяють створити ефективну адаптивну систему з високою точністю виявлення мережевих вторгнень і забезпечити надійний рівень захисту комп'ютерних систем від зовнішніх атак.

Всякий раз, коли дослідник пропонує будь-який новий метод виявлення або захисту запропонований підхід повинен бути реалізований у вигляді експерименту. Для його перевірки та оцінки необхідні набори даних (dataset), які можуть бути попередньо зібрані чи згенеровані програмними засобами [2]. DDoS-атаки еволюціонують та стають більш складними для визначення системами виявлення вторгнень (IDS), через недостатність відомостей про актуальні атаки. Доступні та згенеровані набори трафіку стають все більш важливими при розробці нової IDS.

Актуальність. Сучасні кібератаки, які завдають великого збитку компаніям, вважаються віддаленими, серед яких найбільш небезпечною є атака на відмову в обслуговуванні, а особливо – їх розподілена реалізація – DDoS-атака (Distributed Denial of Service). Ця атака популярна своєю простотою, численними відкритими відомостями про її реалізації і нечисленними обчислювальними ресурсами. DDoS-атаки є активним інструментом конкурентної боротьби , а також інформаційних воєн.

Об'єктом дослідження є процеси виявлення DDoS-атак.

Предметом дослідження є методи та засоби виявлення DDoS-атак.

Метою магістерської кваліфікаційної роботи є підвищення точності виявлення DDoS-атак на основі нейромережевого підходу. Для досягнення мети необхідно розв'язати такі задачі:

– проаналізувати сучасні типи DDoS-атак та інтелектуальні методи захисту;

– розробити інформаційну технологію;

– згенерувати необхідний набір даних;

– виконати моделювання різних архітектур нейронних мереж;

– розробити програмний засіб;

– провести тестування розробленого засобу.

Методи дослідження. Для реалізації поставлених задач були використані методи теорії штучного інтелекту для проектування нейронної мережі; статистичні методи для підготовки вхідної та вихідної інформації для моделювання нейронної мережі, методи проектування програмного забезпечення для розробки та верифікації інтелектуальної системи.

Наукова новизна. Запропонована інтелектуальна інформаційна технологія для виявлення DDoS-атак, яка полягає у аналізі мережевого трафіку штучною нейронною мережею на основі емпірично підібраних параметрів, що дозволяє підвищити точність виявлення 9 видів атак до 97%.

Практична цінність. Розроблено програмний засіб для виявлення DDoS-атак на основі технологій штучних нейронних мереж, ефект від якої полягає в автоматизації підтримки прийняття рішень системного адміністратора, що базується на використанні інтелектуальних технологій, що дозволяє оперативно та з високою точністю виявляти факти мережевого вторгнення, відповідно, вчасно застосовувати контрзаходи.

За результатами науково-дослідної роботи оформлено та відіслано заявку на отримання авторського свідоцтва на твір «Комп'ютерна програма. Інтелектуальний програмний засіб для виявлення DDoS-атак. «NeuroSoft for DDoS detection».

Результати наукової роботи доповідалися на міжнародній науково-практичній конференції «Фотоніка-ОДС-2018» та опубліковані у збірнику тез.

Також подано до друку статтю «DDoS-attack detection using artificial neural networks with Matlab» у співавторстві до видання “Proceedings of SPIE”, що входить до наукометричної бази Scopus.

До проблеми формування набору даних для дослідження DDoS-атак. Конференція ВНТУ електронні наукові видання, XLVIII Науково-технічна конференція факультету інформаційних технологій та комп’ютерної інженерії / Б. В. Кульчицький, Л. М. Куперштейн – ВНТУ, 2019.

Аналіз інструментальних засобів здійснення DDoS-атак. Конференція ВНТУ електронні наукові видання, Молодь в науці: дослідження, проблеми, перспективи / Б. В. Кульчицький, Л. М. Куперштейн – ВНТУ, 2019.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Науково-технічне обґрунтування

Зловмисники з кожним роком знаходять все нові вразливі місця та розробляють техніки для реалізації атак. Останнім часом проводяться не прямі, а саме amplification DDoS-атаки, які використовуються сервери для збільшення коефіцієнта атаки. Так у 2019 році хакери активно використовували вразливі місця в DNS, NTP, SSDP, Chargen, SNMP та Memcached, щоб максимально збільшити масштаб своїх атак [3]. Крім того, спостерігається значне збільшення експлуатації пристроїв IoT для генерації великого потоку пакетів та атак на рівні додатків. Найбільша атака, відбулася в березні 2018 року на GitHub. Обрушилася потужна DDoS-атака в історії, яка становила 841 Гбіт/с [4]. На рис. 1.1 наведено динаміку потужності DDoS-атак [5].

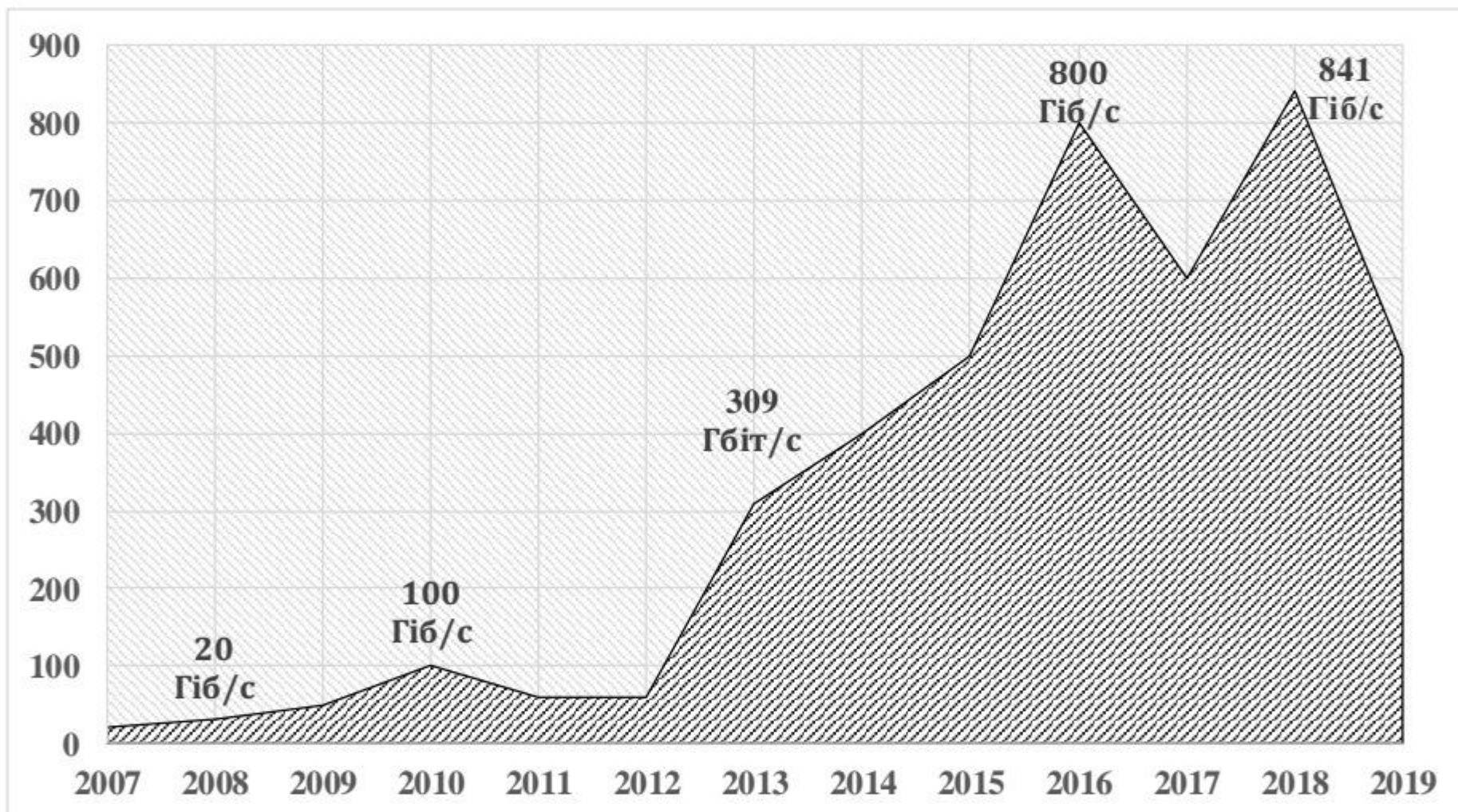


Рисунок 1.1 – Статистика потужності DDoS-атак

Середні збитки недоступності послуг після певної DDoS-атаки компаніям становить від 1000 до 5000 доларів за хвилину [3]. Однак близько 5 відсотків збитки обходиться від 10 тис. до 20 тис. доларів, що в два рази перевищує попередній рік. Це вказує або на те, що вартість DDoS-атаки значно зросла, або що більше компаній зараз знають про справжній вплив на їхній бізнес.

Як видно з рис. 1.2 втрати від DDoS-атак можуть бути як незначними та і дуже критичними в залежності від цілі (атакуючого ресурсу).

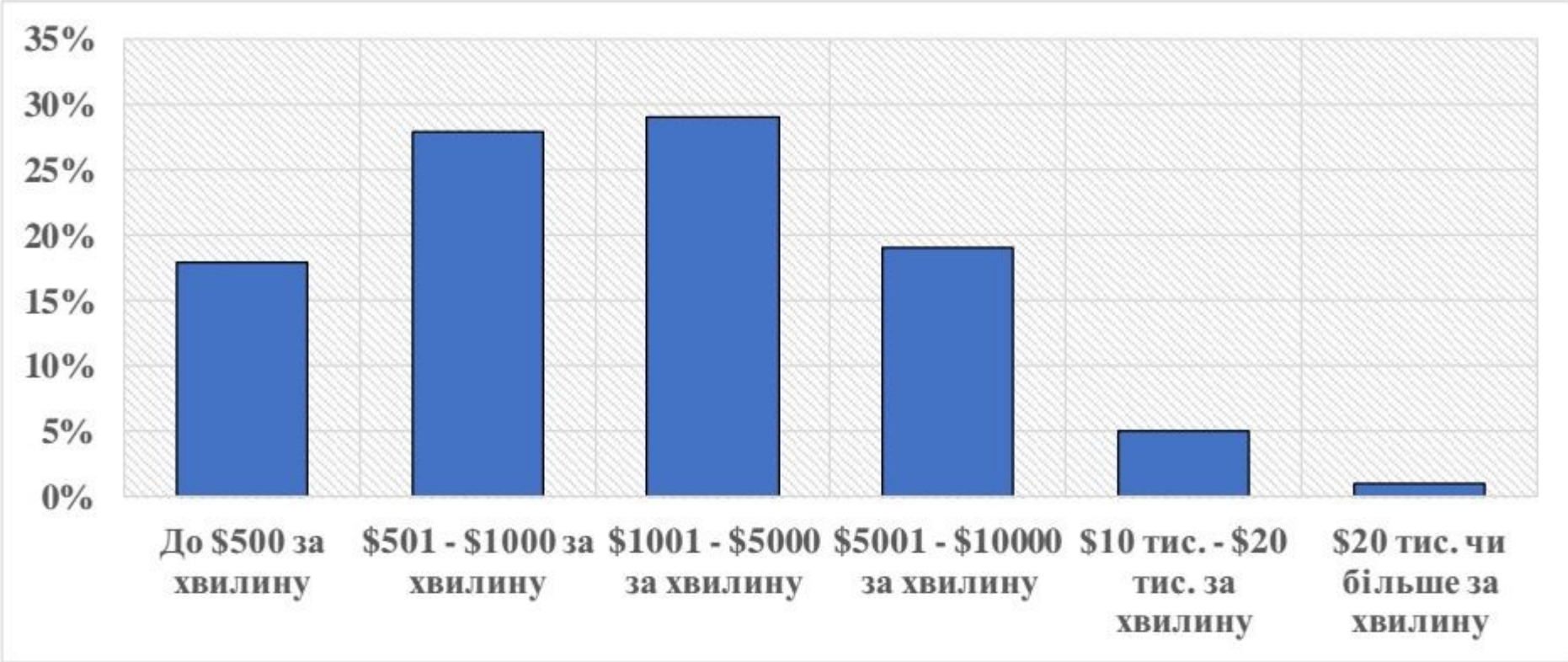


Рисунок 1.2 – Статистика втрат компаній від DDoS-атак

Тому, дуже актуальним на сьогодні стоїть питання розробки інформаційної технології для оперативного, високоточного виявлення та попередження DDoS-атак.

1.2 Аналіз DDoS-атак

DDoS-атака – це критична загроза безпеці мережі, яка є широко поширеною для дослідників в сфері кібербезки [6]. Складність у визначенні схеми атак, різноманітність інструментів генерації трафіку для атаки, складність відстеження підробленої адреси джерела атаки сприяє тому що DDoS-атаку може здійснити навіть користувач не маючи відповідного досвіду. Основна мета DDoS-атаки – порушити доступ до послуг законним користувачам, що в свою чергу, заподіює фінансові та репутаційні втрати для цільових компаній.

DDoS-атаки можуть використовувати вразливості протоколу чи програми для надсилання неправомірних пакетів жертві, що називається протокольною атакою. В іншому типі атаки, яка називається DDoS-flood, жертва перевантажується повідомленнями, що надсилаються їй. Тому жертва не може надавати послуги законним користувачам. За допомогою скомпрометованих хостів або ботнетів широкомасштабна DDoS-атака може бути запущена за лічені хвилини.

Ранні типи DDoS-атак – це переважно ручні атаки, де зловмисники переважно повинні виконати кілька кроків, включаючи виявлення скомпрометованих машин для генерації ботів в Інтернеті, сканування портів та розгортання зловмисного програмного забезпечення до початку остаточної атаки. В даний час інструменти DDoS-атаки стали автоматизованими та досконалими, завдяки чому зловмисники могли виконати всі або кілька кроків автоматично з мінімальними зусиллями людини. Зловмисники також можуть налаштувати параметри, характерні для цілі, тоді як рештою можна керувати за допомогою автоматизованих інструментів. Ці автоматизовані засоби атаки включають Trinoo, Tribe Flood Network (TFN), TFN2K, Trinity, Knight та Stacheldraht, більшість з яких працює над Internet Relay Chat (IRC), в якому скомпрометовані машини та боти можуть спілкуватися опосередковано, не розкриваючи свою особу [7]. Інші інструменти для атаки здебільшого агентські засоби, в яких боти та обробники спілкуються безпосередньо, обмінюючись інформацією між собою.

Існує ряд досліджень, які пропонують класифікацію щодо DDoS-атак [8]. Але з кожним роком зловмисники знаходять нові методи та вразливості в протоколах для реалізації атаки. Тому, для реалізації актуального засобу захисту, необхідно проаналізували актуальні типи атак, які широко використовують атакуючі за допомогою протоколів, заснованих на TCP/UDP прикладного рівня.

1.2.1 Посилюючі DDoS-атаки

Посилюючі атаки – це види мережевих атак, в яких особистість зловмисника залишається прихованою, використовуючи законні сторонні компоненти, де пакети відправляються на відбиваючі сервери зловмисниками з вихідною IP-адресою, встановленою для цільової IP-адреси жертви, щоб переповнити жертву пакетами відповідей. Ці атаки можуть бути здійснені через протоколи прикладного рівня, використовуючи протоколи транспортного рівня, тобто протокол керування передачею (TCP), протокол дейтаграм користувача (UDP) або за допомогою комбінації обох. Атаки на основі TCP включають MSSQL, SSDP тоді як атаки на основі UDP включають TFTP та NTP. Існують певні атаки, які можна здійснити за

допомогою будь-якого з протоколів TCP або UDP, такі як DNS, LDAP та SNMP [9]. Схема посилюючої DDoS-атаки зображено на рис. 1.3.

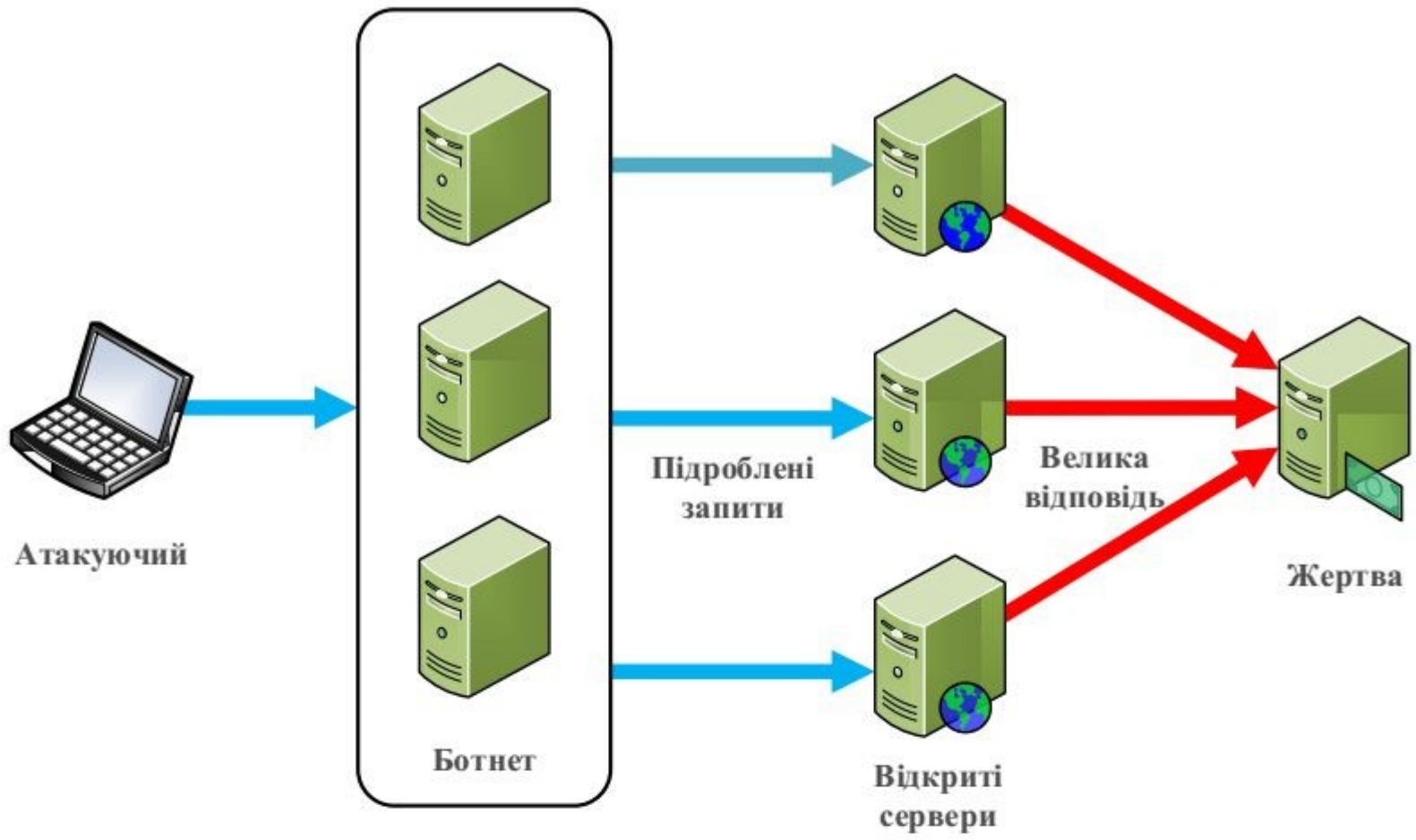


Рисунок 1.3 – Схема посилюючі DDoS-атаки

MSSQL. Тип атаки який базується на використанні в роботі протоколу Microsoft SQL з цілю запуску атаки відбиття, для реалізації DDoS-атаки [10]. Атака відбувається, коли Microsoft SQL Server відповідає на запит, вимагає використовувати протокол дозволу Microsoft SQL Server (MC-SQLR), прослуховуючи UDP-порт 1434. Протокол дозволу SQL використовується щоразу, коли клієнт вимагає отримати інформацію від сервера MS SQL. Підключившись до серверів бази даних, клієнт отримує відповідь у списку екземплярів бази даних. При цьому використовується протокол MC-SQLR, який допомагає визначити, з якими екземплярами бази даних задаються встановити зв’язок. Зловмисники можуть використовувати сервери SQL, виконуючи скриптові запити з використанням фіктивної IP-адреси, створюючи враження, що він надходить від жертви. Кількість фактично існуючих екземплярів бази даних, наявних на уявних серверах SQL, визначає силу або коефіцієнт посилюючих DDoS-атаки.

SSDP. Simple Service Discovery Protocol (SSDP) тип атака який використовує Universal Plug and Play (UpnP) мережеві протоколи, щоб направити посилений

обсяг трафіку на цільову жертву, переважаючи інфраструктура цілі та вимкнення їх веб-ресурсів [11]:

1) Спочатку зловмисник проводить сканування, шукаючи пристрої підключення та відтворення, які можна використовувати як коефіцієнти посилення.

2) Коли зловмисник виявляє мережеві пристрої, вони створюють список усіх пристроїв, які реагують.

3) Зловмисник створює пакет UDP з підробленою IP-адресою цільової жертви.

4) Зловмисник використовує ботнет для надсилання підробленого пакета виявлення на кожен пристрій підключення із запитом якомога більше даних, встановивши певні прапори, зокрема ssdp: rootdevice або ssdp: all.

5) У результаті кожен пристрій надішле відповідь націленій жертві, кількість даних приблизно в 30 разів більша, ніж запит зловмисника.

6) Ціль отримує великий об'єм трафіку з усіх пристроїв і стає перевантаженим, що є результатом DDoS-атаки для законного трафіку.

TFTP. Тривіальний протокол передачі файлів Trivial File Transfer Protocol назва якого і описує його функціонал, не підтримує автентифікацію та не містить будь яких механізмів безпеки. Цілі атаки типу TFTP заповнюються RRQ (запит на читання) DATA відповідями. Атака робить запит за замовчуванням для файлу "/x" у цьому випадку з сервера TFTP. Потерпілий сервер TFTP повертає дані запитуваному цільовому хосту, в результаті цього запиту незалежно від невідповідності імені файлу [12].

NTP. Тип атаки в якій зловмисник використовує функціональність NTP (Network Time Protocol) сервера, щоб переповнити цільову мережу або сервер з посиленою кількістю трафіку UDP, роблячи ціль та оточуючу її інфраструктуру недоступними для регулярного руху. Усі посилюючі атаки використовують невідповідність вартості пропускної здатності між зловмисником та цільовим веб-ресурсом. Коли розбіжність у вартості збільшується в багатьох запитах, отриманий обсяг трафіку може порушити мережеву інфраструктуру. Надсилаючи невеликі запити, які призводять до великих відповідей, зловмисний користувач може

отримати більше від меншого. При множенні цього збільшення за рахунок того, що кожен бот у ботнеті робить подібні запити, зловмисник одночасно перешкоджає виявленню та отримання переваг сильно збільшеного трафіку [13].

DNS. Тип атаки в якій зловмисник використовує функціональність відкритих DNS розпізнавача для того, щоб переповнити цільовий сервер або мережу з посиленою кількістю трафіку, роблячи сервера та її оточуючу інфраструктуру недоступною. У результаті, кожний бот звертається з проханням відкрити DNS розпізнавач зі спустошеною IP-адресою, яка була змінена на реальну вихідну IP-адресу цільової жертви, ціль потім отримує відповідь від DNS розпізнавача. Щоб створити велику кількість трафіку, зловмисник структурує запит таким чином, що генерує якомога більшу відповідь від DNS розпізнавача [14].

LDAP. Тип атаки при якому атакуючі використовують вразливі сервери з підтримкою CLDAP (Connection-less Lightweight Directory Access Protocol) для перенаправлення мусорного трафіка на свої цілі. Атакуючі відправляють багато запитів, обраних до порту LDAP, з заміною IP-адресою відправника на адресу жертви. В результаті, сервери завалили жертву відповідями, розмір яких значно перевищує розмір запитів [15].

SNMP. Тип атаки який має на меті ініціювати багаторазове збільшення відповідей на SNMP-запит [16]. GetBulkRequest, реалізований в SNMP версії 2, призначений для можливості запросити великий обсяг даних, чим і користуються атакуючі, задіюючи неправильно налаштовані сервери в інтернеті. Суть вразливості полягає, як правило не в налаштуванні кількості значень, що віддаються на один GetBulkRequest, а в тому, що значення SNMP community встановлено за замовчуванням: public – read-only або що ще гірше, private – readwrite. Протокол SNMP версій 1 і 2 заснований на UDP, використовується для моніторингу та управління, а в якості аутентифікаційного параметра доступу до керованого обладнання, використовує значення community, яке може бути поставлено лише для читання (read-only) або з можливістю запису (read-write) . Найчастіше в системах при активації сервісу SNMP встановлюється значення за замовчуванням – public для read-only і private для read-write. Практично безмежний

привілейований доступ з правами адміністратора до пристрою дає read-write community. Навіть якщо не будуть проводитися шкідливі зміни, інтенсивні запити з використанням протоколу SNMP можуть викликати значне навантаження на обчислювальні ресурси опитуваного сервера, чим впливає на якість надавання їм сервісів.

1.2.2 Експлуатаційні DDoS-атаки

Експлуатаційні атаки – це види атак, в яких особистість зловмисника може залишається прихованою, використовуючи законні сторонні компоненти та спрямовані на створення великої кількості напівз'єднання. Ці атаки також використовують протоколи транспортного рівня TCP та UDP. Атаки на основі TCP включають SYN-flood та атаки на основі UDP відповідно UDP-flood [17].

Зловмисники створюють власні ботнети, скануючи Інтернет на вразливі пристрої, а потім скомпроментовувають за допомогою шкідливого програмного забезпеченням, яке дало змогу зловмисникам віддалено керувати ботами. На жаль, зловмисникам більше не потрібно будувати ботнети, вони можуть орендувати сформовані ботнети для використання у операторів, які стягують дуже мало грошей за короткочасні (але ефективні) атаки.

Схема експлуатаційної DDoS-атаки зображено на рис. 1.4.

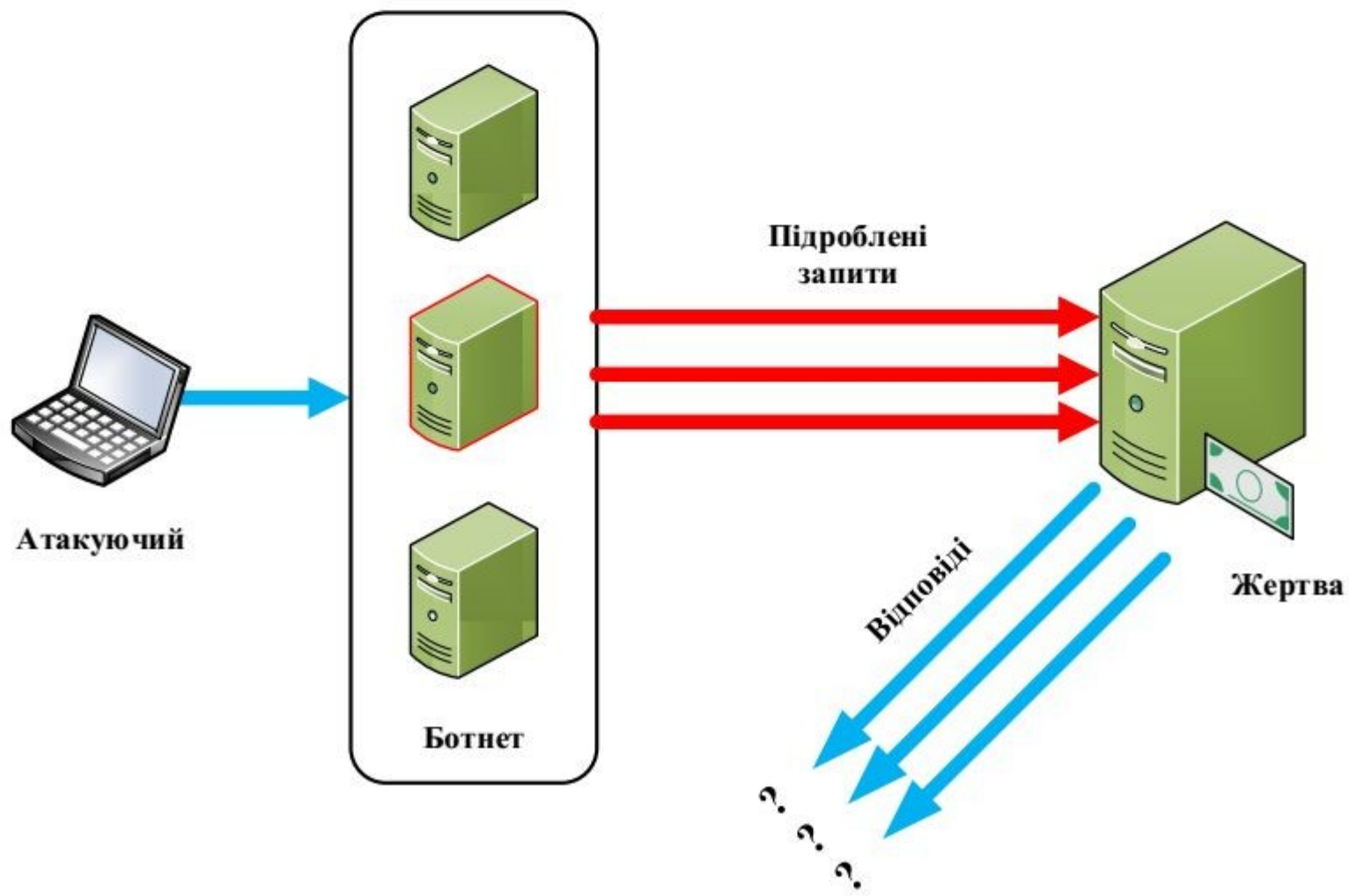


Рисунок 1.4 – Схема експлуатаційної DDoS-атаки

SYN-flood. Тип атаки який має на меті зробити сервер недоступним для легального трафіку, використовуючи всі наявні ресурси сервера [18]. Неодноразово надсилаючи початкові пакети запитів на з'єднання (SYN), зловмисник в змозі перекрити всі доступні порти на цільовій серверній машині, внаслідок чого цільовий пристрій повільно реагує на законний трафік або зовсім не працює. SYN-flood атаки працюють, використовуючи процес рукостискання з'єднання TCP. Для реалізації DDoS-атаки зловмисник використовує той факт, що після отримання початкового пакету SYN сервер відповість назад одним або декількома пакетами SYN/ACK і чекатиме останнього кроку в рукостисканні.

UDP-flood. Тип атаки при якому велика кількість UDP пакетів надсилається на цільовий сервер з метою перекрити можливість пристрою обробляти та реагувати [19]. Міжмережевий екран, що захищає цільовий сервер, також може бути вичерпаний внаслідок UDP-flood, що призводить до відмови в наданні послуг у законному трафіку. UDP-flood працює головним чином, використовуючи кроки, які виконує сервер, коли він відповідає на пакет UDP, що надсилається в один з його портів. Оскільки кожен новий пакет UDP отримує сервер, він проходить кроки для обробки запиту, використовуючи серверні ресурси в процесі. Коли передаються пакети UDP, кожен пакет буде містити IP-адресу вихідного пристрою. В результаті використання цільового сервера, ресурси цілі можуть швидко вичерпатися, коли надійде велика кількість пакетів UDP, що призводить до відмови в обслуговуванні до нормального трафіку.

Проаналізовані типи DDoS-атак зображено на рис. 1.5.

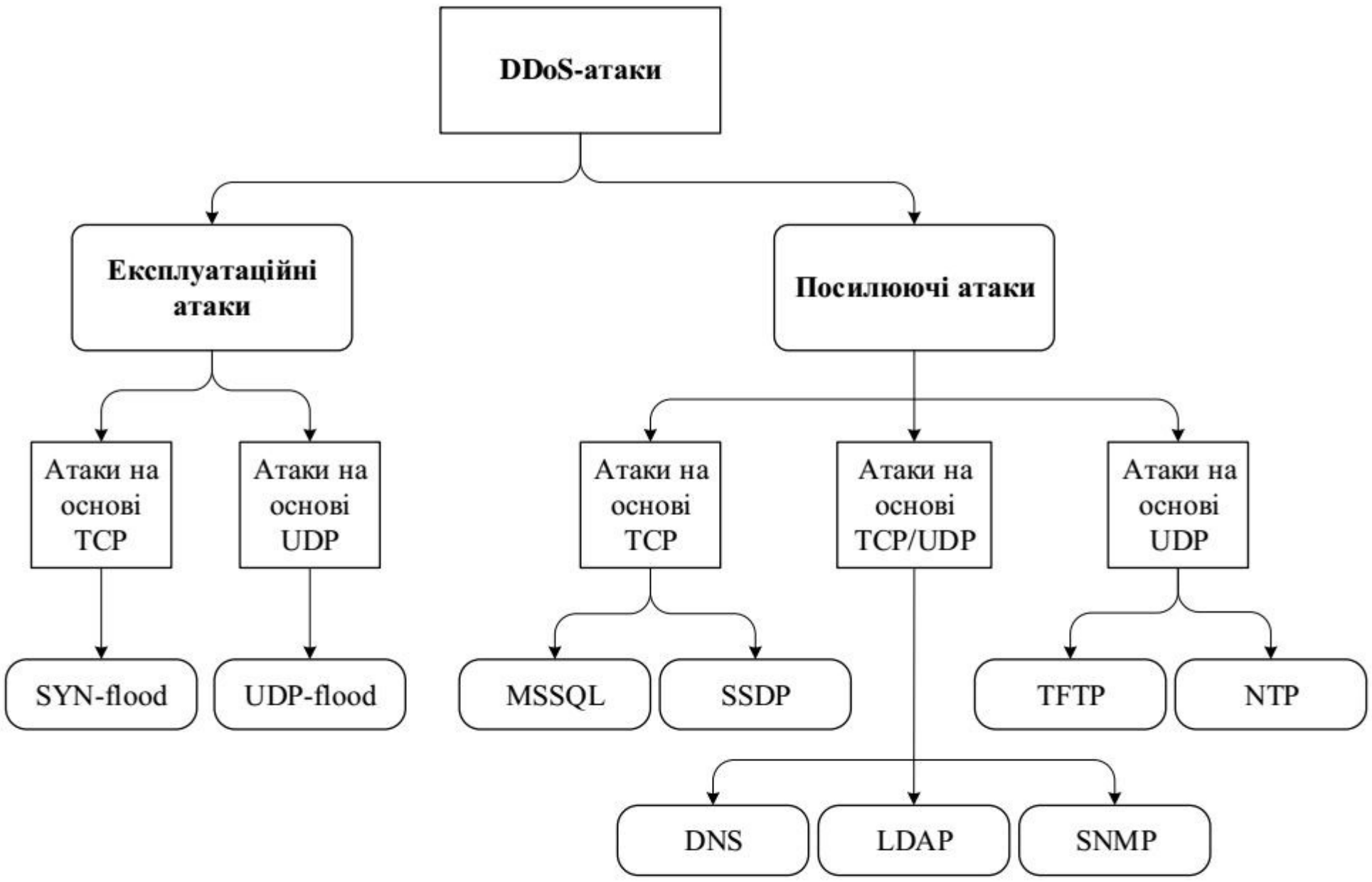


Рисунок 1.5 – Типи DDoS-атак

DDoS-атаки еволюціонують та стають більш складними для визначення системами виявлення вторгнень (IDS), через недостатність відомостей про актуальні атаки. Доступні та згенеровані набори трафіку стають все більш важливими при розробці нової IDS.

1.3 Інтелектуальні підходи захисту від DDoS-атак

DDoS-атаки не мають загальних атрибутів, за допомогою яких їх можна виявити. Крім того, розподілений характер DDoS-атак робить атаки надзвичайно важкими, щоб їм протистояти або відстежувати, і автоматизовані програмні засоби, що розгортають DDoS-атаки, можуть бути легко отримані простим користувачем. Зловмисники також можуть використовувати підробку IP-адресу для приховування своєї ідентичності і таким чином роблять виявлення DDoS-атак все більш складним. Нарешті, машини підключені до Інтернету, мають недостатній рівень безпеки, а веб-хости містять кілька вразливостей у безпеці.

Найбільш популярними та універсальними підходами на сьогодні є інтелектуальні технології в яких використовується штучний інтелект. Система

виявлення, розроблена на таких підходах, може модифікувати свій процес виконання відповідно до нещодавно зібраних даних. Система може підвищити свою ефективність у деяких тестових випадках на основі попередніх результатів. Системи, які побудовані на основі штучного інтелекту (ШІ) зосереджуються на отриманні правил, що створюють нові дані [20]. Вони надають такі ознаки, як паралелізм, стійкість, допустимість помилок, неточності та невизначеності.

До технологій штучного інтелекту можна віднести: Байєсові мережі, нечітка логіка, генетичні алгоритми, алгоритм К-Найближчих сусідів, нейронні мережі, програмні агенти та опорно-векторні машини (рис. 1.6).

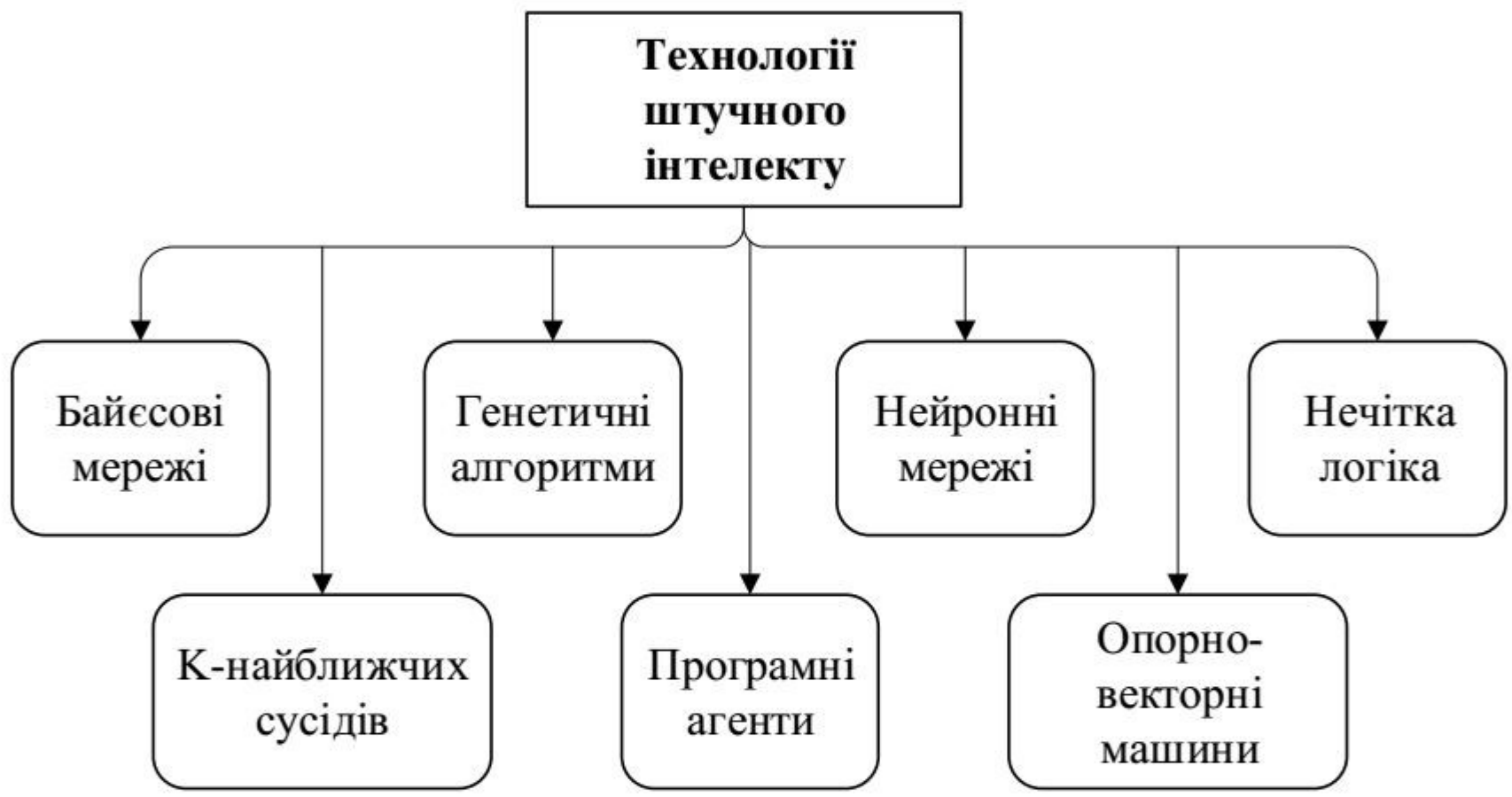


Рисунок 1.6 – Системи штучного інтелекту

Байєсові мережі (Bayesian networks). Визначає ймовірні асоціації серед змінних, що цікавлять [21]. Ця методика зазвичай використовується для виявлення атак разом зі статистичними схемами, які дають безліч переваг, таких як можливість кодування взаємозалежностей між змінними, прогнозування подій, включаючи попередні дані та знання.

Нечітка логіка (Fuzzy logic). Поняття нечіткості використовується поряд із методами ідентифікації DDoS-атаки, щоб більше акценту було зроблено на мережевих аномаліях або атак [22]. Основа теорії нечітких множин використовується для наближення міркувань, а не для того, щоб бути точно впевненим за допомогою традиційної логіки. Використання нечітких наборів, а

також їх правил, застосовуються коли велика кількість вхідних параметрів, таких як час використання процесора, швидкість активності та тривалість з'єднання, може бути неоднозначною при обробці неповних наборів даних.

Генетичні алгоритми (Genetic algorithms). Класифікується як евристичні прийоми, які ґрунтуються на гіпотетичних думках про природний відбір та інструментах, що використовуються в євгеніці для отримання досить точних рішень або встановлення загадок оптимізації [23]. Цей алгоритм застосовує еволюційні статистичні прийоми, такі як селекція, кросовер (рекомбінація), успадкування, мутація та елітаризм. Обираються тонкі тестові зразки, як інформаційний набір даних і зменшує підроблені позитивні масштаби, коли людський ввід використовується в циклі зворотного зв'язку.

K-найближчих сусідів (K-nearest neighbor). Підхід який генерує прогнози та визначає їх порівнюючи найближчий елемент графу [24]. Ввід можна класифікувати на групи, що використовують цей найближчий елемент, а сусідні місця можна визначити в режимі реального часу, використовуючи цей параметр географічно. Спочатку K-NN повинен записувати IP-адреси, отримані на сервері. Пізніше слід записати їх у файл та створити графік із довготою та широтою як осі. Дуже висока щільність, продемонстрована графом у певній географічній області, може свідчити про потенційний DDoS.

Нейронні мережі (Neural networks). Представлені як заміна статистичних методик, які класифікують параметри відповідно до ряду попередніх вхідних даних які було надано [25]. Для ефективної класифікації нейрону мережу слід правильно натренувати, щоб навчання було успішним. Нейронні мережі є обчислювально "важкими" і потребують підготовки великого набору даних для правильної класифікації. Штучним нейронним мережам потрібен великий набір даних для навчання, щоб вони могли побудувати свою модель, що складається з входів, прихованих шарів і виходів, а також їх зв'язків між собою.

Програмні агенти (Software Agents). Агент – це програмне забезпечення або суміш програмних та апаратних об'єктів, які можуть виконуватися паралельно від імені своїх користувачів. Він включає багато корисних функцій, таких як здатність

до навчання, співпраця, реактивність та ефективність. Котенко та Уланов [26] визначають агентурну методологію та програмне середовище, розроблені для моделювання розподілених методів захисту, які можна встановити в Інтернеті для нейтралізації мережевих атак. У рекомендованому підході кібернетична нейтралізація “malicious guys” та механізмів безпеки характеризується взаємодією різних команд агентів.

Опорно-векторні машини (Support vector machines). Методика навчання, яка використовується для побудови навчальних векторів у просторі віднесеного простору, класифікує їх відповідно [27]. SVM розглядає класифікацію як проблему квадратичної оптимізації. Інтегрує управління узагальненням для запобігання “прокляття розмірності” встановлюючи верхню межу на грані між різними класами, тим самим надаючи практичні засоби для великих і постійно змінюваних наборів даних.

Використання нейронних мереж, які займаються ідентифікацією та розпізнавання образів, можливо вирішити клас задач до яких і відноситься попередження та виявлення DDoS-атак. НМ навчають за допомогою набору даних з відомими відповідями. Ця технологія носить як сигнатурний характер, так і може бути само адаптованою до нових видів атак. Перевагами використання нейромережевого підходу при виявленні DDoS-атаки є гнучкість та апроксимаційну універсальність. Нейромережа здатна аналізувати неповні або перекручені дані, одержувані з мережевого трафіку. Здатність обробляти дані від великої кількості джерел є особливо важливою при розгляді DDoS-атак, проведених проти мережі скоординованими численними атакуючими.

1.4 Аналіз інформаційного забезпечення для дослідження DDoS-атак

Для отримання набору даних DDoS-атак дослідники можуть: використовувати спеціалізовані програмні засоби для отримання необхідної їм типу атаки, так і використання вже зібраних даних, які містять як реальний так і згенерований трафік. Дані можуть бути використані для формування нових систем виявлення вторгнень, які здатні передбачити різні типи DDoS-атак.

Існують загальнодоступні набори даних, які дослідники використовують для тестування своєї методики та ефективності алгоритмів. Деякі набори даних отримуються з реальних атак, тоді як інші об'єднання імітаційних атак. Тим не менш, є необхідність відзначити, що статистичні дані, представлені в наборі даних, як правило, відрізняються від реального мережевого трафіку.

Надійність вибраного набору даних для перевірки підходу залишається відкритим питанням. Дуже важливо вибрати відповідний набір даних для перевірки будь-якої запропонованої методики виявлення DDoS-атаки. В захопленому мережевому трафіку повинно містити суміш нормального та атакуючого трафіку у відповідній пропорції, і не повинно бути зміщень до певного типу трафіку. Але, дуже важко забезпечити відповідну суміш нормального і атакуючого трафіку в реальному наборі експериментів, оскільки не існує відомої формули для правильного моделювання мережевого трафіку. У дослідженні DDoS-атак широко використовуються такі набори даних як CAIDA, DARPA і TUIDS інші набори даних використовуються рідко, тому що вони досить застарілі або обмежені для використання. Так в популярному наборі даних KDD Cup 1999 дуже важко розрізняти трафік атаки з нормального трафіку, оскільки набір даних не належно позначений [28].

Також можна використати доступні програми для генерування трафіку та побудови власної мережі. Спосіб реалізації моделювання або емуляції набору даних залежить від вимог до генерації типу трафіку і мети виконання експерименту. Застосовуючи моделювання та емуляцію мережі, імітований трафік може бути близьким до реального трафіку DDoS-атаки, але проблеми полягають не

тільки в витратах на придбання середовища емуляції чи змодельованої мережі, але й на додаткових професійних знаннях, необхідних для створення, розгортання та експлуатування комплектів або системи керування [29]. Тому, дослідники обирають вже реалізовані програмні засоби для проведення DDoS-атак. Інструменти, що використовуються при моделюванні чи реалізації DDoS-атаки включають: Ddosflowgen, OMNET++, Shaft, Tribe Flood, Network (TFN), LOIC, Trininty v3, Knight, WinCap and JpCap [30-34]. В табл. 1.1 наведено порівняння різних інструментів DDOS-атаки з відповідними можливостями.

Таблиця 1.1 – Інструменти моделювання DDoS-атак

Інструмент моделювання	Протокол	Атака	Можливості
Ddosflowgen	UDP, TCP	UDP flood, TCP requests, Mirai scans	– Можливість визначення кількості атакуючих мереж і налаштування параметрів таких як: коефіцієнт посилення, вектори атаки та кількість джерел мережевих атак. – Створює набори синтетичних даних трафіку з N переглядів.
OMNET++	UDP, TCP, ICMP	Атаки транспортного ріння	– Керована форма веб-серверу. – Здатний до моделювання TCP/IP.
Shaft	ICMP, UDP, TCP	TCP flood, UDP flood, ICMP flood	– Обробники та агенти спілкуються через UDP. – Рандомізує вихідний порт і IP-адреси в пакетах. – Фіксований розмір пакета під час атаки. – Перемикає керування основними серверами і портами в режимі реального часу тим самим ускладнюючи засоби виявлення вторгнень.
Tribe Flood Network (TFN)	TCP, UDP, ICMP	TCP SYN, ICMP flood, smurf	– Використовується для зменшення пропускної здатності та ресурсів. – Використовує інтерфейс командного рядка для атакуючого.
LOIC	TCP, UDP, HTTP	UDP, TCP, HTTP flood	– Інструмент анонімної атаки на основі IRC. – Існує як двійкова, так і веб-версія
Trinity v3	UDP, TCP	TCP floods, RST packet floods,	– TCP flood зроблені шляхом рандомізації всіх 32-бітів IP-адреси джерела. – Пакети Flood генерується за допомогою випадкових прапорів управління.
Knight	TCP, UDP	PUSH and flood TCP, SYN, UDP flooding	– Використовує Back Orifice, троянські програми для встановлення цільового хоста. – Містить генератор контрольної суми. – Дуже легкий, але потужний інструмент атаки на IRC.
WinCap and JpCap	TCP, UDP, ICMP	TCP dump, UDP ICMP dump	– Програма на базі Windows для передачі мережевого трафіку і процесу стека протоколів
Saddam та інші скрипти	TCP, UDP	Атаки транспортного ріння	– Модулі які містять реалізації певних атак і автоматизовані

Більшість з перелічених програм доступні безкоштовно. Серед популярних Ddosflowgen та LOIC які виконують DDoS-атаку та генерують синтетичний набори даних трафіку. Середовище OMNET++ дає змогу змоделювати власну топологію мережі. Використання скриптів дає змогу здійснити атаку з мінімальною кількістю інструментів та не знаючи алгоритму атаки.

1.5 Формалізація вимог та постановка задачі

Проаналізувавши вищевикладені дані можна зробити висновок, що DDoS-атаки не втрачають своїх потужностей, а збитки від них стають все більшими. Хоча дослідникам і відомі топології реалізації атак, їхнє своєчасне виявлення та запобігання залишається актуальною проблемою. Все більше набувають методи захисту основані на штучному інтелекту, а саме нейронні мережі, що є хорошими апроксиматорами в задачах класифікації. Але отримання даних, наближених до реальних атак, є також важливою задачею при навчанні НМ. Існує велика кількість інструментів для моделювання чи реалізації DDoS-атак, а також простих скриптів, що реалізують атаку в декілька натискань.

Розробка інформаційної технології включатиме декілька етапів:

1) Перехоплення трафіку для аналізу та попереднього навчання.
2) Моделювання нейронної мережі, що передбачає:
   – обробка даних і представлення її для навчання нейронної мережі;
   – вибір архітектури НМ;
   – навчання НМ та корегування параметрів;
   – тестування НМ для визначення її кількості помилок.
3) Реакція на атаку у вигляді блокування чи попередження.

В результаті виконання перерахованих етапів буде сформовано інформаційну технологію для виявлення DDoS-атак, яка дозволить виявляти сучасні типи атак з підвищеною точність виявлення.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

2.1 Інформаційна технологія виявлення DDoS-атак

Інформаційна технологія включає в себе комплекс методів та засобів, за допомогою яких реалізується певна задача. Інформаційна технологія складається з процесів, які є незалежними частинами, що дає змогу виявляти проблеми на ранніх стадіях проектування та швидко їх усунути. Складові інформаційної технології представлено на рис. 2.1.

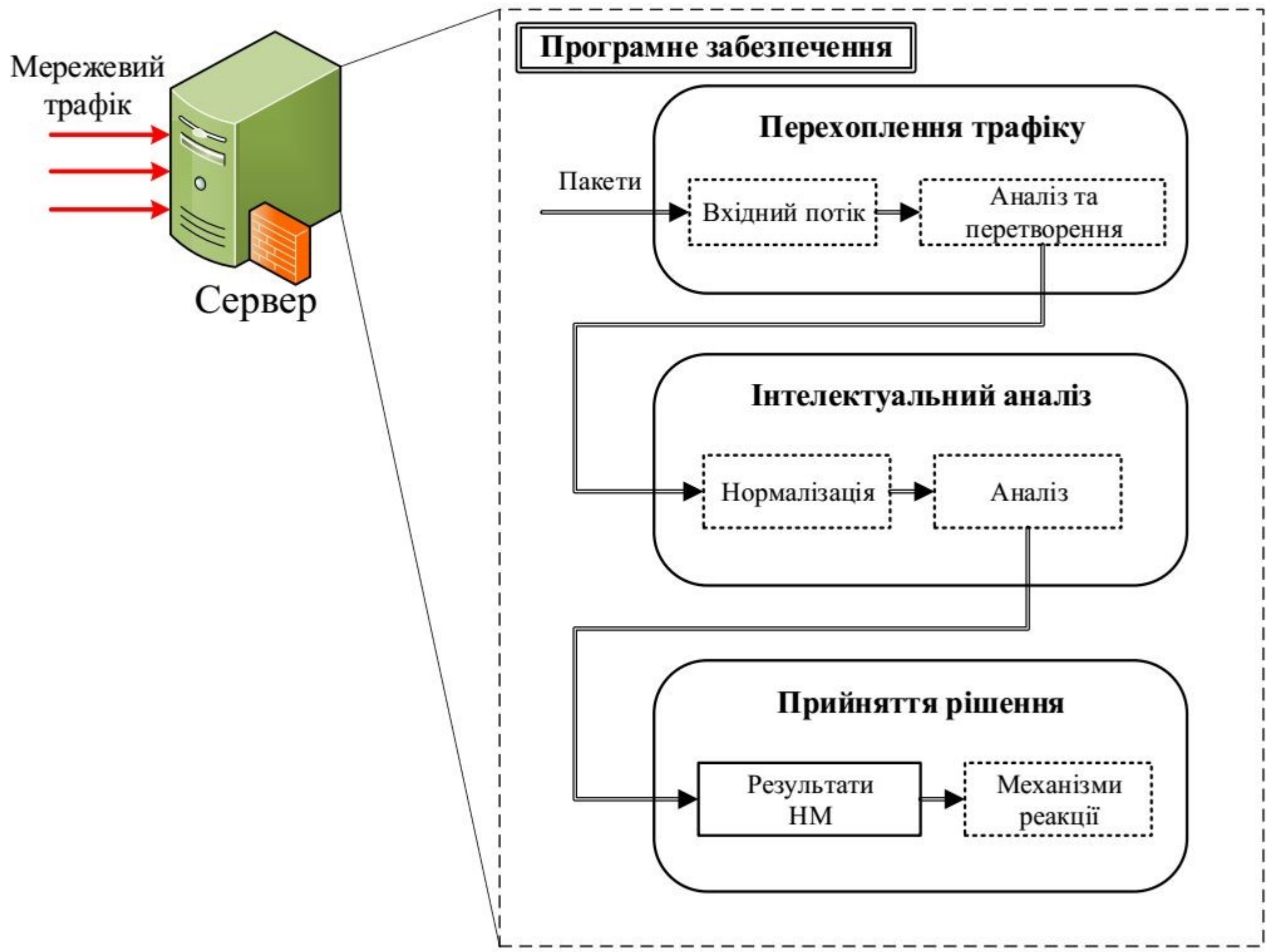


Рисунок 2.1 – Складові інформаційної технології виявлення DDoS-атак

Інформаційна технологія містить у собі три процеси:

– перехоплення трафіку – зчитування вхідного мережевого трафіку, вибір відповідних параметрів, аналіз та перетворення;

– інтелектуальний аналіз – нормалізація результатів попереднього модуля на вхід НМ та їх аналіз;

– прийняття рішення – отримання результату НМ, їх аналіз для застосування відповідного механізму реакції.

Кожен з процесів спроектований для виконання конкретної під задачі. В свою чергу процес, має етапи в яких подаються певні дані на вхід, обробляються та надсилаються далі.

Процес перехоплення трафіку містить у собі два етапи. На вхід першому надходить мережевий трафік, які перехоплюється та перетворюється у файл зі структурою pcap. Отриманий файл надходить на другий етап, де відбувається аналіз даних, вибір необхідних полі та формування статистичних параметрів. Результат об'єднується та надсилається до наступного процесу (рис. 2.2).

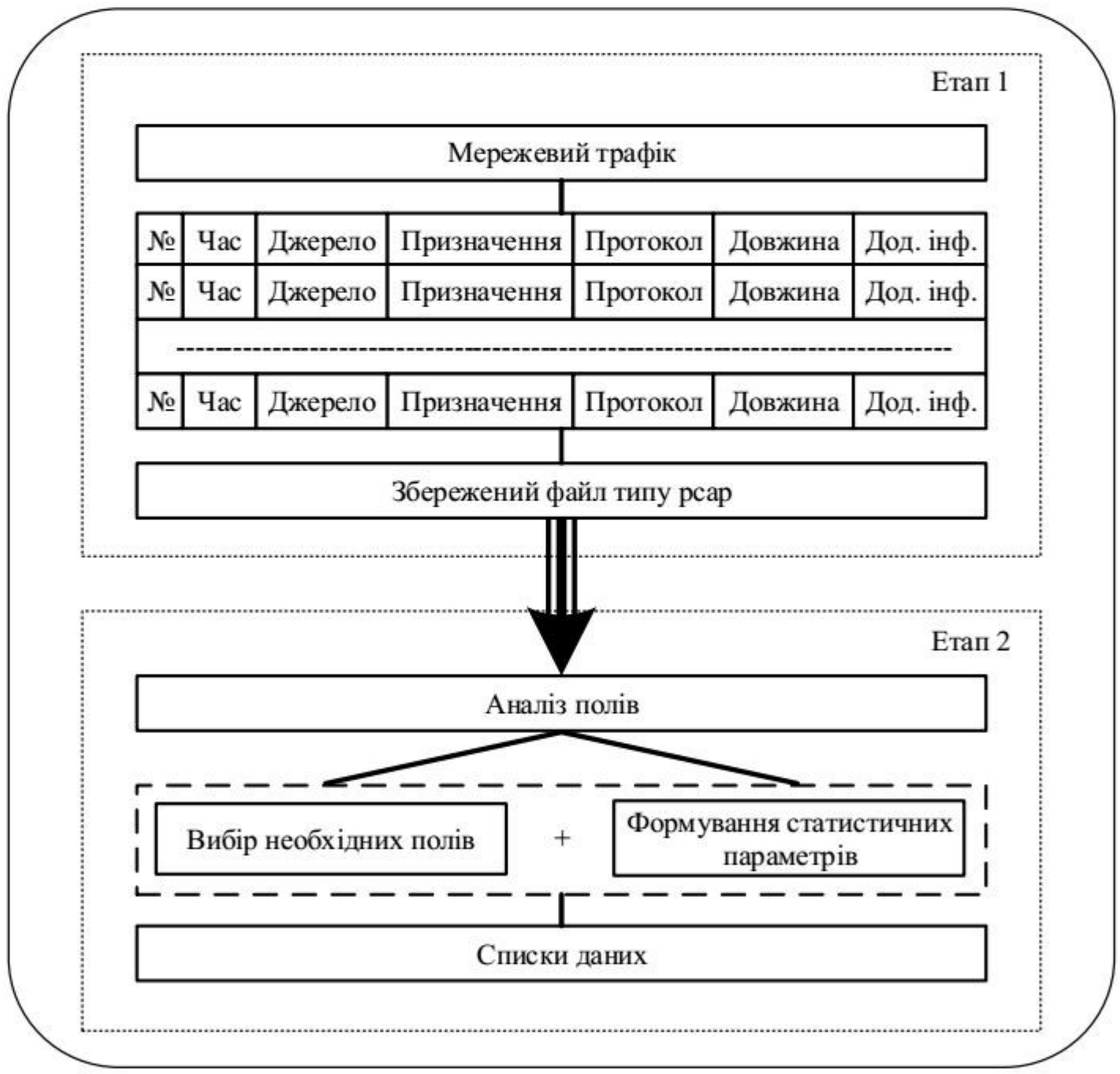


Рисунок 2.2 – Схема процесу перехоплення трафіку

Процес інтелектуального аналізу також складається з двох етапів. Вхідні списки даних необхідно перетворити, тобто символьні значення перевести в числа та нормалізувати, привести до певного вигляду. В основі інтелектуального аналізу

лежить НМ, яка попередньо навчається, після чого вона і зможе аналізувати, класифікувати відповідні атаки (рис. 2.3).

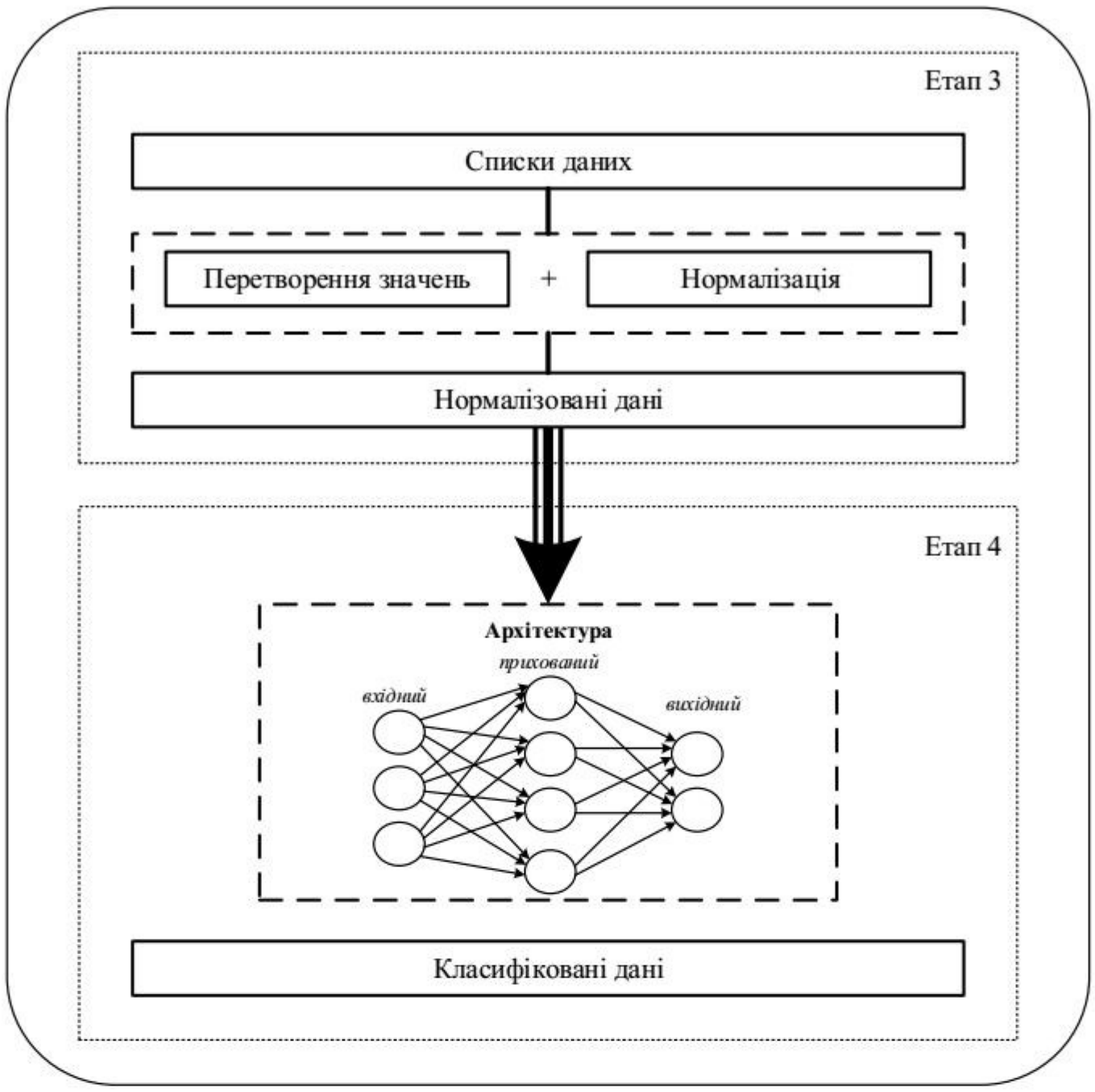


Рисунок 2.3 – Схема процесу інтелектуального аналізу

Процес прийняття рішення має лише один етап. Класифіковані дані НМ надходять на механізм реакцію, в якому і буде прийматися рішення, щодо реакції на атаку. Серед них може бути, як попередження користувача, так і блокування вхідного трафіку (рис. 2.4).

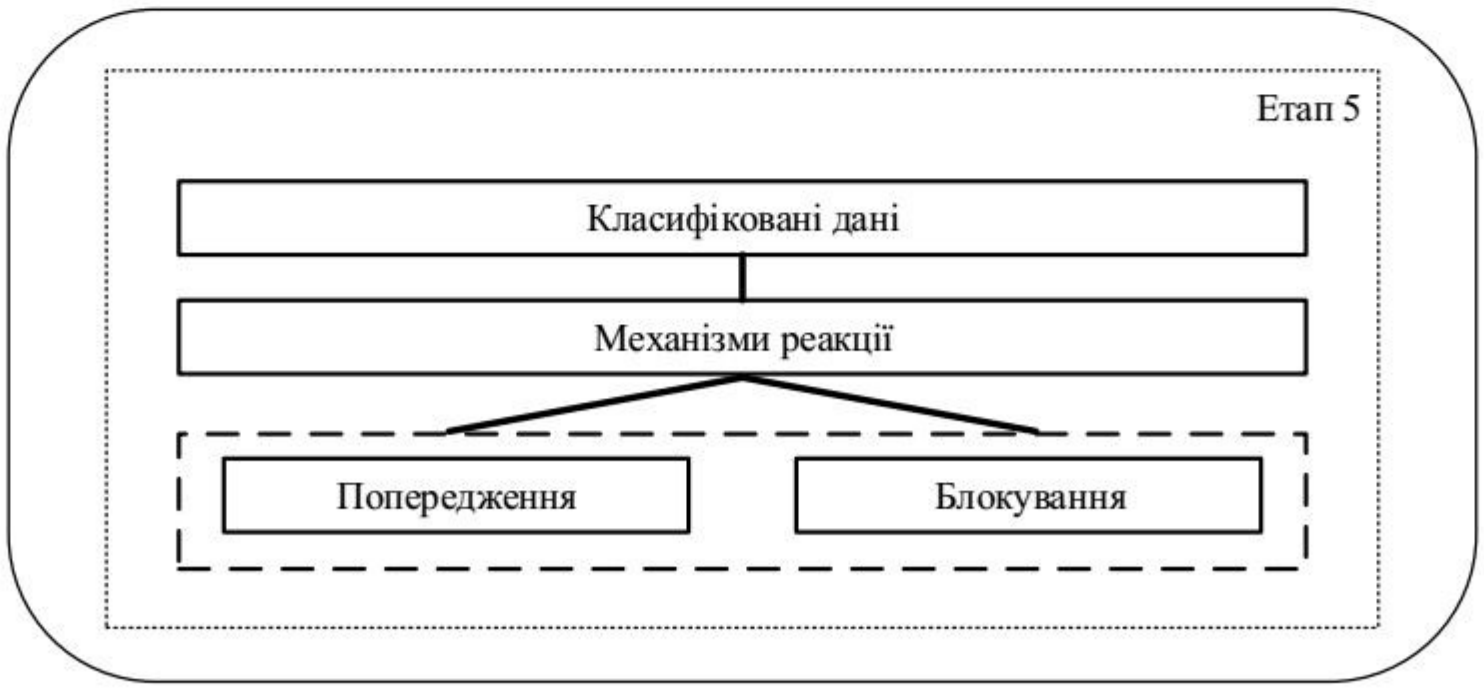


Рисунок 2.4 – Схема процесу прийняття рішення

Отже, результатом має бути інформаційна технологія, яка моніторить мережевий трафік, перетворює отримані дані для НМ, а вона в свою чергу класифікує чи це є атакою, якщо так то якого виду та виконує відповідні механізми реакції.

2.2 Формування набору даних для інтелектуального аналізу

Формування атакуючого трафіку буде отримуватись за допомогою симуляції наближеної до реальної. Для цього буде використано: порт сканер MASSCAN [35], скрипти на python для реалізації атак та модуль перехоплення трафіку в основі якого лежить tshark сканер. Структура атаки буде подібна до тієї, що розглядалась в першому розділі див. рис. 1.3. Схему формування набору даних зображено на рис. 2.5

Рисунок 2.5 – Схема формування набору даних

Посилюючі DDoS-атаки мають спільну структуру. Мета яких відправляти запити, з отримувачем жерти на сервера, які в свою чергу «підсилюють», збільшують розмір пакета.

Для реалізації такого роду атак необхідні сервери з відповідними відкрити портами. Для цього і буде використано MASSCAN, сканер який дає змогу сканувати Інтернет на відкриті порти (рис. 2.6).

```
root@toor:/media/sf_Shared# masscan --help
MASSCAN is a fast port scanner. The primary input parameters are the
IP addresses/ranges you want to scan, and the port numbers. An example
is the following, which scans the 10.x.x.x network for web servers:
 masscan 10.0.0.0/8 -p80
The program auto-detects network interface/adapter settings. If this
fails, you'll have to set these manually. The following is an
example of all the parameters that are needed:
 --adapter-ip 192.168.10.123
 --adapter-mac 00-11-22-33-44-55
 --router-mac 66-55-44-33-22-11
```

Рисунок 2.6 – MASSCAN в терміналі Kali Linux

Типовий шаблон команди MASSCAN може бути: masscan -p[номер портів] [проміжок адрес]. Після якого і буде виконуватись сканування проміжку адрес на відкритість відповідних портів.

До відбиваючих DDoS-атак можна віднести, протокол та порт: DNS-53, LDAP-389, SNMP-161, CharGen-19, NTP-123, MSSQL-1434 та SSDP-1900. Використовуючи сканер MASSCAN буде відбуватись сканування хостів в Інтернеті на відкритість перелічених портів. Прикладом є NTP-123, з командою masscan -p123 --max-rate 1200 170.0.0.0-185.255.255.255. Результатом має бути список портів які доступні (рис. 2.7).

```
root@toor:/media/sf_Shared# masscan -p123 --max-rate 1200 170.0.0.0-185.255.255.255

Starting masscan 1.0.4 (http://bit.ly/14GZzcT) at 2019-11-10 20:40:21 GMT
 -- forced options: -sS -Pn -n --randomize-hosts -v --send-eth
Initiating SYN Stealth Scan
Scanning 268435456 hosts [1 port/host]
Discovered open port 123/tcp on 185.91.121.85
Discovered open port 123/tcp on 177.234.228.209
Discovered open port 123/tcp on 183.60.234.66
Discovered open port 123/tcp on 171.21.123.40
Discovered open port 123/tcp on 185.203.230.67
Discovered open port 123/tcp on 178.232.50.195
Discovered open port 123/tcp on 183.245.146.203
Discovered open port 123/tcp on 185.240.232.105
Discovered open port 123/tcp on 181.200.233.126
Discovered open port 123/tcp on 183.60.234.115
Discovered open port 123/tcp on 181.200.5.133
Discovered open port 123/tcp on 178.232.64.63
```

Рисунок 2.7 – Результати сканування MASSCAN

Для кожного з перерахованих типів атаки необхідно провести сканування та зберегти отримані адреси в файл.

Перед реалізацію атак необхідно відфільтрувати отримані адреси на показник коефіцієнта відбиття. Тобто, у скільки разів буде збільшуватись відправлений запит. Для цього буде використовуватись написані скрипти на python, які містять перевірку та реалізовані атаки, на вхід лише потрібно файл з IP-адресами. Одним з таких є Saddam, та його модифікації, в якому містяться перевірка коефіцієнта amplification для обраного файлу та типу DDoS-атаки. Щоб перевірити отриманні IP-адреси необхідно ввести команду: python Saddam.py benchmark --ntp=NTP.txt.

```
https://github.com/OffensivePython/Saddam
https://twitter.com/OffensivePython

Protocol|  IP  Address  |     Amplification     |     D
-------------------------------------------------------
 ntp    | 152.4.151.235 |     0x (8B -> 0B)     |N/A
 ntp    |164.114.115.242|     0x (8B -> 0B)     |N/A
 ntp    |178.232.244.12 |    6x (8B -> 48B)     |N/A
 ntp    | 163.16.6.241  |     1x (8B -> 8B)     |N/A
 ntp    | 163.16.6.178  |     1x (8B -> 8B)     |N/A
 ntp    | 163.16.199.43 |     1x (8B -> 8B)     |N/A
 ntp    |168.255.58.194 |     0x (8B -> 0B)     |N/A
 ntp    |168.255.155.38 |     0x (8B -> 0B)     |N/A
 ntp    |168.184.225.23 |     0x (8B -> 0B)     |N/A
 ntp    | 168.255.32.28 |     0x (8B -> 0B)     |N/A
 ntp    |156.239.247.165|    6x (8B -> 48B)     |N/A
 ntp    | 171.21.46.52  |     0x (8B -> 0B)     |N/A
 ntp    | 154.83.17.204 |   21x (8B -> 168B)    |N/A
```

Рисунок 2.8 – Перевірка IP-адрес в Saddam

Після чого обираються IP-адреси в яких найбільші коефіцієнти відбиття. За допомогою їх відповідей і буде виконуватись атака з використанням. В скрипті модифікуються заголовок пакету для відповідного протоколу. Команда для виконання атаки: python Saddam.py 192.168.1.50 --ntp=NTP_0.txt.

```
root@toor:/media/sf_Shared# python Saddam.py 192.168.1.50 --ntp=NTP_0.txt

https://github.com/OffensivePython/Saddam
https://twitter.com/OffensivePython

     Sent      |    Traffic    |    Packet/s    |     Bit/s
-----------------------------------------------------------------
    61.28K     |   120.55MB    |   10.90Kpps    |   13.44Mbps
```

Рисунок 2.9 – NTP DDoS-атака за допомогою скрипту

Атакуючий трафік пересилається на локальну мережу та перехоплювався з використанням Wireshark. Отриманий трафік Wireshark дає змогу відфільтрувати по декільком різним параметрам, результати зберігаються у вайл. Перехоплений трафік у Wireshark зображено на рис. 2.10.

ip.dst == 192.168.1.50

No.	Time	Source	Destination	Protocol	Length	Info
168488	1.057479	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168489	1.057523	172.16.0.5	192.168.1.50	UDP	482	643 → 16459 Len=440
168490	1.057524	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168491	1.057525	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168492	1.057525	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168493	1.057526	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168494	1.057527	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168495	1.057527	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168496	1.057528	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440
168497	1.057529	172.16.0.5	192.168.1.50	UDP	482	639 → 20147 Len=440

Рисунок 2.10 – Перехоплений трафік в Wireshark

Отриманий файл необхідно проаналізувати, обрати поля та сформувати на їх основі вхідні параметри для НМ.

Для решти типів атак алгоритм дій такий самий. Лише для експлуатаційних не потрібно було сканувати адреси, використовувались скрипти Memcrashed-DDoS-Exploit та aSYNcron які вже мали внутрішні списки [36, 37].

Основі параметри які беруться для вибірки: IP-джерела, порти відправника та отримувача, протоколи та довжина пакету. IP-адреса джерела ідентифікує атакуючого, а саме сервер через якого відбулася відбиваюча атака для подальшого його блокування. Порти джерела та отримувача на які часто звертаються. Протоколи, серед яких TCP та UDP. Довжина пакету – заголовок та дані, які

зазвичай повторюються. Час з першого фрейму та з попереднього, дуже малі значення які також можуть свідчити про атаку. Сформовані параметри на основі статистики, певного проміжку пакетів. Загальна кількість пакетів з відповідного IP-джерела. Загальна кількість байтів для відповідного IP-адресу, сума довжини всіх пакетів IP-адресу, швидкість передачі у бітах за секунду та тривалість потоку IP-адресу.

Таблиця 2.1 – Фрагмент необроблених даних

<table>
<tr><td colspan="5">Основні параметри</td></tr>
<tr><td rowspan="2">IP-джерела</td><td colspan="2">Порт</td><td rowspan="2">Протокол</td><td rowspan="2">Довжина пакету, байт</td><td>Час</td></tr>
<tr><td>Відправника / отримувача</td><td>З першого / попереднього фрейму, с</td></tr>
<tr><td>185.37.100.107</td><td>634/29594</td><td>TCP/UDP</td><td>482</td><td>$4 \times 10^{-6}/1 \times 10^{-6}$</td></tr>
<tr><td colspan="5">Сформовані параметри</td></tr>
<tr><td>Кількість пакетів</td><td colspan="2">Загальна кількість байт</td><td>Швидкість передачі, бітів/с</td><td>Тривалість потоку, с</td></tr>
<tr><td>82</td><td colspan="2">38524</td><td>25×10^{6}</td><td>0.0125</td></tr>
</table>

Сформовані поля об’єднуються в одні дані та будуть подаватись на вхід нейронній мережі. Вихідні параметри включають нормальний трафік та 9 типів DDoS-атак: SYN-flood, UDP-flood, MSSQL, SSDP, DNS, LDAP, SNMP, TFTP, NTP.

2.3 Дослідження нейронних мереж

Алгоритми нейронної мережі вважаються найкращими при класифікації шаблонів DDoS-атак на основі статистичних особливостей та його здатності розпізнавати шаблони непідготовлених даних, а також здатність працювати з неточними та неповними даними. Типи нейронних мереж, які використовуються в задачі класифікації DDoS-атак відносять: прямого поширення, згорткова, рекурентна та модульна.

Нейромережа прямого поширення (Feedforward Neural Network) – найпростіші типи штучних нейронних мереж. У нейронній мережі, що подається, дані проходять через різні вхідні вузли до тих пір, поки вони не досягнуть вихідного вузла. Дані рухаються лише в одному напрямку від першого рівня поки не досягнуть вихідного вузла. Це також відоме як передня хвиля, що

розповсюджується, що зазвичай досягається за допомогою класифікаційної функції активації. На відміну від більш складних типів нейронних мереж в мережі прямого поширення відсутній метод зворотного поширення помилки і дані рухаються лише в одному напрямку.

Згорткова нейронна мережа (Convolutional Neural Network) використовує варіацію багатошарових перцептронів. CNN містить один або декілька згорткових шарів. Ці шари можуть бути повністю пов'язані між собою або об'єднані. Перш ніж передати результат наступному шару, згортковий шар використовує згорнуту операцію на вході. Завдяки цій згортковій роботі мережа може бути набагато глибшою, але зі значно меншими параметрами. Згорткові нейронні мережі також показують чудові результати в семантичному розборі та виявленні парафрази. Вони також застосовуються при обробці сигналів та класифікації зображень.

Рекурентна нейронна мережа (Recurrent Neural Network) – це тип штучної нейронної мережі в якій вихід певного шару зберігається і повертається на вхід. Перший шар формується так само, як і в мережі прямого поширення. Тобто з добутком суми ваг та ознак. Однак у наступних шарах починається повторюваний процес нейронної мережі. Від кожного часового кроку до наступного кожен нейрон запам'ятовує певну інформацію, яку він мав у попередньому часовому кроці. Іншими словами, кожен нейрон виконує функцію комірки пам'яті під час обчислення та виконання операцій. Якщо прогноз неправильний, система самостійно навчається та працює над правильним прогнозуванням під час розмноження. LSTM – це варіація рекурентної мережі. LSTM-мережі складаються з повторюваних елементів. Кожен такий елемент містить чотири шари і відрізняється тим, що має комірку довгої короткочасної пам'яті. Ідея, покладена в основу концепції полягає в тому, щоб використовувати одну LSTM-мережу для читання вхідної послідовності крок за кроком, щоб отримати векторний простір даних фіксованої розмірності, а потім використовувати іншу LSTM-мережу для формування вихідної послідовності з цього вектора.

Модульна нейронна мережа (Modular Neural Network) має ряд різних мереж, які функціонують незалежно та виконують підзадачі. Різні мережі насправді не

взаємодіють між собою та не передають сигнал під час обчислення. Вони працюють незалежно щодо досягнення результату. В результаті великий і складний обчислювальний процес можна зробити значно швидше, розбивши його на незалежні компоненти. Швидкість обчислення зростає, оскільки мережі не взаємодіють і навіть не з'єднуються між собою.

Різні дослідники запропонували декілька технік та механізмів захисту, щоб допомогти виявити DDoS-атаку.

Кім та інші [38], запропонували використовували Cisco Systems NetFlow та два різних методи обміну даними для виявлення різних типів DDoS-атак. NetFlow надав сім унікальних та корисних функцій для кожного трафіку даних, що надходить у мережу. Сюди входили: IP-адреса джерела, IP-адреса призначення, порт джерела, порт призначення, тип протоколу 3 рівня, байт TOS (DSCP) та логічний інтерфейс введення (в індексі). Вони використовували дерева прийняття рішень для автоматичного вибору різноманітних можливостей наданих NetFlow для моделювання трафік-шаблонів різних типів DDoS-атак. До недоліків можна віднести вибір функції NetFlow, виконаний маршрутизатором cisco. Це відбувається тому, що вибір функції маршрутизатора cisco доведеться регулярно оновлювати, щоб він міг виявляти нові функції, які залежать від DDoS-атаки. Запропонована методологія залежить від набору даних для оновлення бази знань алгоритму штучної нейронної мережі (ANN). Вони зазначили, що їхній підхід був зосереджений на аналізі та класифікації єдиних даних про трафік.

Лукас та інші [39], запропонували методику для виявлення DDoS-атак, яка ґрунтувалася на трьох основних факторах: вивчення статистичних особливостей вхідного трафіку DDoS-атаки, байєсівський класифікатор для оцінки і прогнозування ймовірності нападу та використання випадкових рекурентних нейронних мереж (r-RNN), яка об'єднує всю зібрану інформацію про вхідний трафік для прийняття рішення про виявлення. Одним з головних недоліків цієї методики є те, що результати були протестовані лише на автономному застарілому наборі даних, який не мав поточних функцій DDoS-атак. Крім того, ця техніка не є самонавчальною технікою, якій потрібно постійно навчатись із нещодавно

відкритими пакетами DDoS-атак, щоб забезпечити оновлення її стану за допомогою нововиявлених функцій DDoS-атаки.

Юан та інші [40], визначали та порівнювали продуктивність різних моделей нейронної мережі, таких як: CNN, LSTM та GAT з методом Random Forrest. Їх метою було вивчити зразки з послідовностей мережевого трафіку та відстежувати діяльність мережевої атаки. Найкраща точність виявлення за їх моделлю LSTM призвела до точності виявлення 97%. Вони також свідчать про те, що рекурентні нейронні мережі можуть вчитись набагато довші історичні особливості, ніж звичайні методи машинного навчання. Але при цьому вони не визначили, які параметри оцінювання їх моделі впливають на точність виявлення. Крім того, вони не згадали, скільки різних типів атак здатна виявити змодельована нейронна мережа.

Сабріна та інші [41], запропонував процедуру виявлення, яка базується на використанні спостереження та штучного інтелекту (ансамбль RNN) для виявлення DDoS-атаки на клієнта та проміжні вузли. Основна причина вибору цього підходу полягає в тому, що DDoS-атака має різну поведінку як на клієнтському, так і на проміжному вузлах, і згідно з дослідженням авторів, ансамбль зможе допомогти виявити різні стани DDoS-атаки. Недолік такого підходу є етап спостереження, коли передбачається, що збільшення кількості запитів про відхилення та збільшення системних ресурсів вказує на те, що жертва зазнає нападу. Це тому, що такі випадки можуть траплятися і в результаті звичайного використання користувача. Коли це станеться, то всі прогнози, зроблені RNN, будуть помилковими через неправильні функції введення, що подаються в нього.

Існуючі підходи до виявлення DDoS-атак, які було наведено вище, підсумовано та наведено в таблиці 2.2.

Таблиця 2.2 – Існуючі підходи виявлення DDoS-атак

Дослідники	Ціль	Методика виявлення	Опис	Недоліки
Кім та інші	Мережеве виявлення	Дерева прийняття рішень та НМ	Дерева прийняття рішень для автоматичного вибору функцій пакету, а потім НМ для класифікації пакетів атак на основі вибраних функцій.	Використання NetFlow, що потребує оновлення

Лукас та інші	Захист потерпілого вузла	Байєсовий класифікатор та r-RNN	Байєсівські класифікатори для оцінки ймовірності DDoS-атаки, а потім зібрану інформацію як вхід до r-RNN для прийняття рішень щодо виявлення	Застарілий набір даних
Юан та інші	Мережеве виявлення	CNN, LSTM, GRU	Система DeepDefense для виявлення шаблонів DDoS-атаки в мережі	Не було вказано параметри та типи атак
Сабріна та інші	Клієнтська частина та проміжні вузли	Ансамбль RNN	Ансамбль RNN для виявлення різної поведінки DDoS-атак	Лише статистичні ознаки

Основна відмініть у дослідженнях полягає в тому, що жодна стаття не була написана, використовуючи модель LSTM рекурентної нейронної мережі та бібліотеку Tensorflow для розробки мережі для виявлення DDoS-атак. Дослідники, що використовували будь-яку з них у своїй роботі, але не обидві, не мали також актуального набору даних. Більшість робіт також використовували звичайну рекурентну НМ, яка має проблему зникаючого градієнту, яка вирішується з використанням рекурентної НМ на основі LSTM.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

3.1 Обґрунтування вибору засобів програмної реалізації інформаційної технології

Вибір мови програмування є важливим кроком при реалізації будь-якої технології. До популярних та широко використовуваних відносяться: Java, Pyhton та C# [42].

Python – інтерпретована, об'єктно-орієнтована мова програмування високого рівня з динамічною семантикою [43]. Високо рівневі вбудовані структури даних у поєднанні з динамічним набором тексту та динамічним зв'язуванням, Використовується як мова сценаріїв, або для з'єднання існуючих компонентів разом. Простий, легкий у вивченні синтаксис Python підкреслює читабельність і, таким чином, зменшує витрати на обслуговування програми. Python підтримує модулі та пакети, що заохочує модульність програми та повторне використання коду. Інтерпретатор Python та широка стандартна бібліотека доступні у вихідному чи двійковому вигляді безкоштовно для всіх основних платформ і можуть вільно поширюватися.

Java – основана на класах, об'єктно-орієнтована та спеціально спроектована, щоб запускатися на будь-якій системі [44]. Java покликана дозволити розробникам додатків «писати один раз, запускати куди завгодно» (WORA), тобто компільований код Java може працювати на всіх платформах, які підтримують Java без необхідності перекомпіляції. Є безліч додатків і веб-сайтів, які не працюватимуть, якщо у вас не встановлена Java, і більше створюються щодня. Розробникам легко писати програми, які використовують популярні моделі дизайну програмного забезпечення та найкращі практики, використовуючи різні компоненти, знайдені в Java EE. Значну частину екосистеми Java становить велика різноманітність проектів з відкритим кодом, який побудованими спільнотою, програмними платформами та API.

C# – це проста, сучасна та об'єктно-орієнтована мова, яка надає сучасним розробникам гнучкість та можливості для створення програмного забезпечення,

яке не тільки працюватиме сьогодні, але буде застосовано роками в майбутньому [45]. Метою C # було розробити мову програмування, яку не тільки легко вивчити, але й підтримувати сучасний функціонал для всіх видів розробки програмного забезпечення. C# підтримує розробку WEB, мобільних пристроїв та додатків. Деякі з сучасних функцій мови програмування C# підтримують: типи var, автоматичну ініціалізацію типів і колекцій, лямбда-вирази, динамічне програмування, асинхронне програмування, кортежі, відповідність шаблонів, вдосконалене налагодження та обробку винятків тощо.

Також при моделюванні нейронних мереж використовують MATLAB. MATLAB – це платформа програмування, розроблена спеціально для інженерів та вчених [46]. Серцем MATLAB є мова MATLAB, мова на основі матриці, що дозволяє найбільш природне вираження обчислювальної математики. MATLAB простий для початківців, які тільки починають вивчати мову програмування, оскільки придбаний пакет включає все, що вам потрібно. До недоліків можна віднести, що користувач повинен купувати кожен модуль і оплачувати його. Також, під час перехресного компілювання або перетворення MATLAB в інший код мови дуже важко. Це дуже складно або вимагає глибоких знань MATLAB для вирішення всіх помилок.

Python пропонує безліч безкоштовних бібліотек, включаючи Tensorflow (машинне навчання), SciPy (наукові обчислення) та NumPy (підтримка багатовимірних масивів та операції над ними) для проектування проекту на основі ШІ [47].

Програмісти можуть прискорити розробку спеціальних програмних засобів, скориставшись кількома інтегрованими середовищами розробки (IDE) для Python. PyCharm – одна з найбільш широко використовуваних IDE для мови програмування Python [48]. В даний час Python IDE використовується на великих підприємствах, таких як Twitter, Pinterest, HP, Symantec та Groupon. У той же час, інструменти та функції, надані PyCharm, допомагають програмістам швидко та ефективно писати різноманітні модулі на Python. Розробники можуть навіть налаштувати інтерфейс PyCharm відповідно до своїх конкретних потреб та переваг.

Крім того, вони можуть розширити IDE, вибравши з понад 50 плагінів для задоволення складних вимог проекту.

3.2 Розробка структури програмного засобу

При розробці великих програм доцільно частину підпрограм і інших ресурсів, збирати разом і компілювати окремо від основної програми у вигляді бібліотек, ресурсів або модулів. Програмний модуль є незалежною частиною. Це означає, що кожен програмний модуль розроблюється, компілюється і налагоджується окремо від інших частин програми.

Програмний засіб складається з декількох модулів:

– Модуль перехоплення трафіку;

– Модуль аналізу пакетів;

– Модуль інтелектуального аналізу;

– Модуль механізмів реакції.

Схема роботи програмного засобу зображено на рис. 3.1.

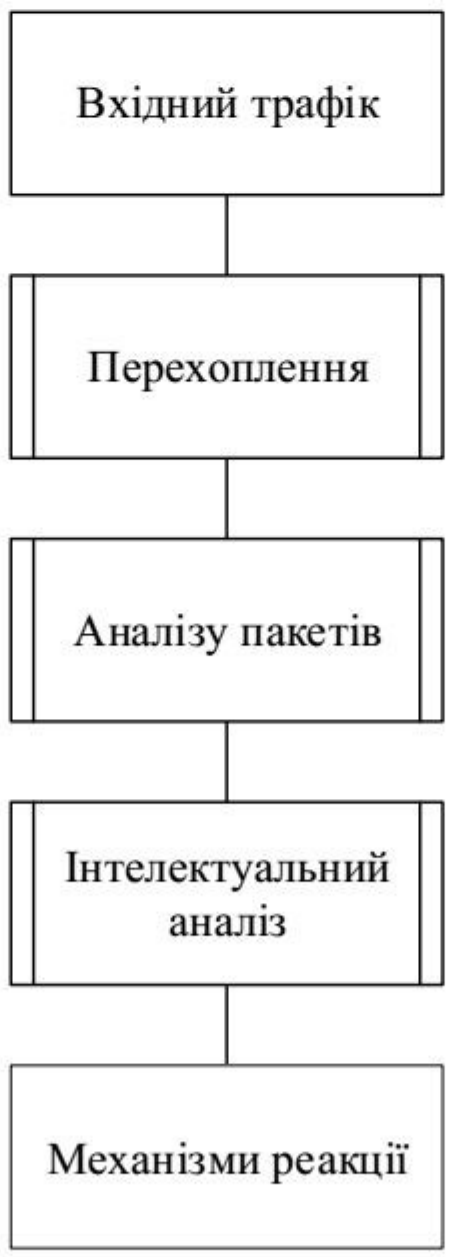


Рисунок 3.1 – Схема роботи програмного засобу

Модуль перехоплення передбачає сканування вхідного трафіку використовуючи мережевий інтерфейс та збереження результатів у файл.

Для його реалізації імпортується модуль pyshark, який базується на основі Wireshark. Pyshark містить функцію LiveCapture, яка передбачає безперервне перехоплення пакетів та може приймати декілька параметрів, серед яких: interface – вибір з якого інтерфейсу ведеться перехоплення та output_file – шлях до файлу в який буде все зберігатись. Результат зберігається як об'єкт. Щоб почати перехоплювати необхідно звернутись до об'єкту та викликати функцію sniff з параметром timeout, значення в секундах, після якого всі дані збережуться у файл. Частина коду представлено нижче:

```
live_capture = pyshark.LiveCapture(output_file=PACKETS_PATH)
live_capture.sniff(timeout=TIME)
```

Збережені перехопленні пакети можна переглядати, вони місять детальну інформації про пакет, приклад пакету зображено на рис. 3.2.

```
Layer IP:
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
    Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    0000 00.. = Differentiated Services Codepoint: Default (0)
    .... ..00 = Explicit Congestion Notification: Not ECN-Capable Transport (0)
    Total Length: 1378
    Identification: 0xeba6 (60326)
    Flags: 0x0000
    0... .... .... .... = Reserved bit: Not set
    .0.. .... .... .... = Don't fragment: Not set
    ..0. .... .... .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment offset: 0
    Time to live: 123
    Protocol: UDP (17)
    Header checksum: 0xef02 [validation disabled]
    Header checksum status: Unverified
    Source: 178.74.235.236
    Destination: 192.168.1.2
Layer UDP:
    Source Port: 443
    Destination Port: 63000
    Length: 1358
    Checksum: 0x645d [unverified]
    Checksum Status: Unverified
    Stream index: 0
    Timestamps
    Time since first frame: 0.003043000 seconds
```

Рисунок 3.2 – Вигляд перехопленого пакету

Модуль перехоплення та аналізу пакетів працюють з використання спільної бібліотеки і мають мету сформувати дані для подальшого аналізу. Схему алгоритму роботи двох модулів зображено на рис. 3.3.

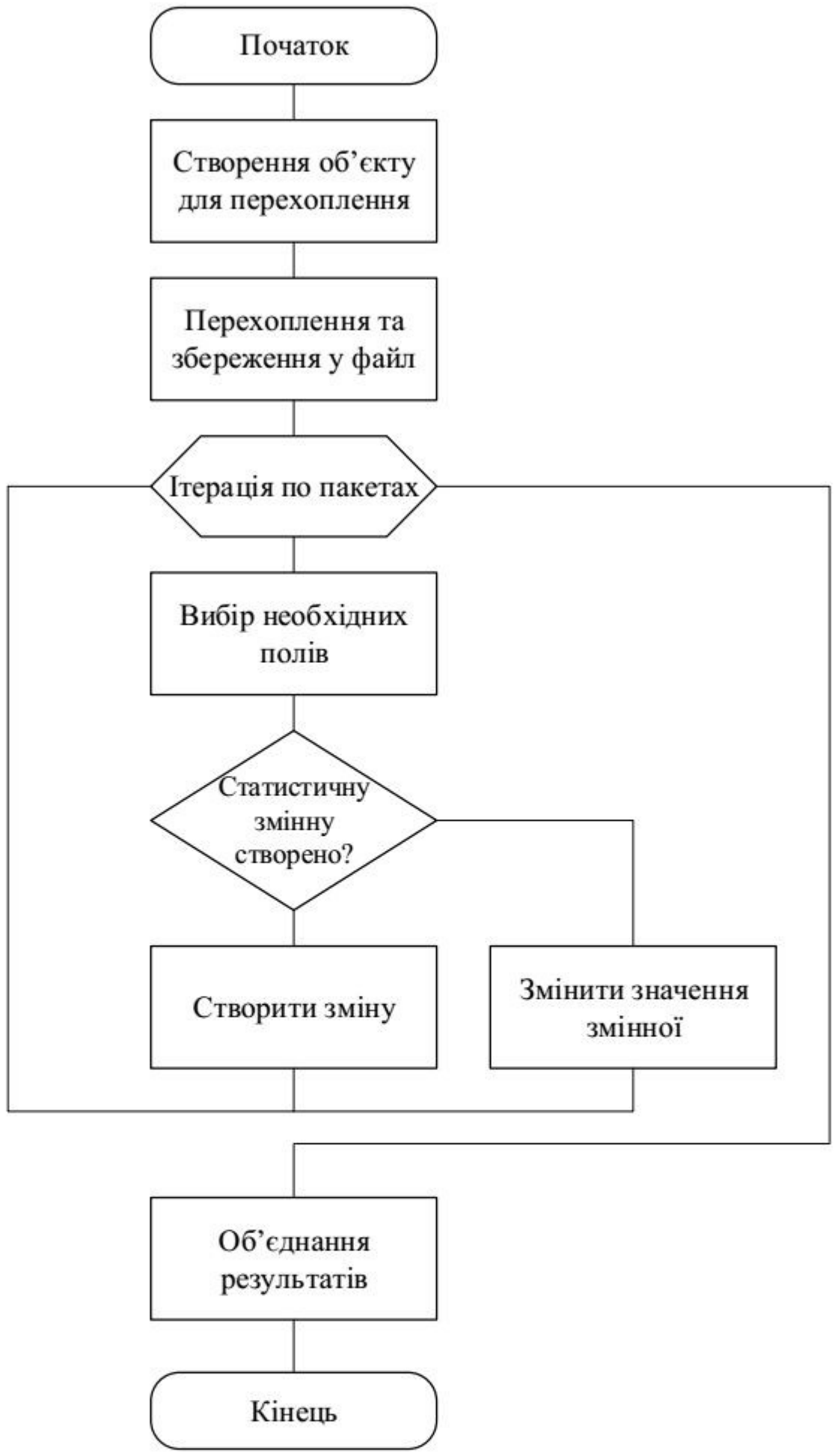


Рисунок 3.3 – Схема алгоритму перехоплення та аналізу пакетів

Модуль аналізу пакетів передбачає зчитування пакетів із збереженого файлу, вибір необхідних параметрів та формування статистичних параметрів на основі всього файлу. Для цього використовується попередній модуль з іншою функцією FileCapture. Файл зчитується по пакету, перевіряючи чи є всі необхідні поля, якщо все пройшло вдало вони дістаються та зберігаються як список. Частина коду представлено нижче:

```
file_packets = pyshark.FileCapture(input_file=INPUT_FILE_PATH)
for packet in file_packets:
    all_params, protocol = check_param(packet)
    if all_params:
       packet_fields.append([packet.ip.addr, packet[protocol].srcport,
       packet[protocol].dstport, packet[protocol].layer_name,
       packet.length] , packet[protocol].time_delta,
       packet[protocol].time_relative])
```

Параметри які дістаються: IP-адрес, порт джерела, порт отримувача, протокол, довжина пакету, час з першого та час з попереднього фрейму. Фрагмент зібраних полів пакетів зображено на рис. 3.4.

```
['140.82.114.26', '443', '53700', 'tcp', '78', '0.000000000', '0.000000000']
['192.168.1.2', '53700', '443', 'tcp', '82', '0.000538000', '0.000538000']
['140.82.114.26', '443', '53700', 'tcp', '60', '0.115435000', '0.115973000']
['192.168.133.1', '5353', '5353', 'udp', '70', '0.000000000', '0.000000000']
['192.168.133.1', '5353', '5353', 'udp', '70', '0.004410000', '0.004410000']
['192.168.133.1', '137', '137', 'udp', '92', '0.000000000', '0.000000000']
['192.168.133.1', '5353', '5353', 'udp', '70', '0.005274000', '0.009684000']
```

Рисунок 3.4 – Фрагмент зібраних полів пакетів

Для статистичних параметрів створено функцію packet_statistics. В неї передаються поля з якими працювали в попередній функції для накопичення статистики. Для цього функція перевіряє чи створено екземпляр, існуючі змінюються, в іншому випаду буде створено новий словник. Частина коду представлено нижче:

```
if not exist:
  statistics.append({'ip': ip, 'srcport': srcport, 'dstport':
  dstport, 'packet_count': 1,'length': length, 'time_frame':
  time_frame})
else:
  dic_params = statistics[index] dic_params['packet_count'] =
  dic_params.get('packet_count') + 1
  dic_params['length'] = dic_params.get('length') + length
  dic_params['time_frame'] = dic_params.get('time_frame') +
  time_frame
  dic_params['bps'] = dic_params.get('bps') + bps
```

Статистичні параметри які формуються: кількість пакетів, загальна довжина, тривалість потоку та бітів за секунду. Фрагмент сформованих статистичних параметрів зображено на рис. 3.5.

```
, 'packet_count': 2, 'length': '7860', 'time_frame': 0.115973, 'bps': 0}
'packet_count': 21, 'length': '828260605454606092707064647070646454549282', 'time_frame': 3.20694, 'bps': 0}
, 'packet_count': 20, 'length': '707092707064646060826060545464645454928 2', 'time_frame': 3.204424, 'bps': 0}
3', 'packet_count': 20, 'length': '707092707064646060826060545464645454928 2', 'time_frame': 3.209969, 'bps': 0}
 'packet_count': 19, 'length': '606082606070709270705464645464645454 92', 'time_frame': 2.4609550000000002, 'bps'
3', 'packet_count': 12, 'length': '60608260605454818154548 2', 'time_frame': 1.743098, 'bps': 0}
 'packet_count': 16, 'length': '606060605454707070706464645454', 'time_frame': 1.702445, 'bps': 0}
```

Рисунок 3.5 – Фрагмент сформованих статистичних параметрів

Як підсумок, результати двох модулів об’єднуються шукаючи однакові IP та порти. Частина коду представлено нижче:

```
for fields in main_fields:
    for stats in statistics:
        if compare_fields(fields, stats):
        temp = fields.copy() + list(map(lambda x: stats.get(x),
            stats_names))
            full_params.append(temp)
```

Фрагмент об’єднаних полів зображено на рис. 3.6.

```
['169.254.161.245', '57394', '5355', 'udp', '64', '0.000000000', '0.000000000', 20, '707092707064646060826060545464645454928 2', 3.209969, 0]
['169.254.161.245', '53435', '5355', 'udp', '64', '0.000000000', '0.000000000', 20, '707092707064646060826060545464645454928 2', 3.209969, 0]
['192.168.1.2', '137', '137', 'udp', '82', '0.000000000', '0.000000000', 21, '828260605454606092707064647070646454549282', 3.20694, 0]
['192.168.1.2', '5353', '5353', 'udp', '60', '0.000000000', '0.000000000', 21, '828260605454606092707064647070646454549282', 3.20694, 0]
['192.168.56.1', '5353', '5353', 'udp', '60', '0.000000000', '0.000000000', 19, '606082606070709270705464645464645454 92', 2.4609550000000002,
['169.254.161.245', '5353', '5353', 'udp', '60', '-0.014612000', '-0.000134000', 20, '707092707064646060826060545464645454928 2', 3.209969, 0]
['169.254.200.128', '5353', '5353', 'udp', '60', '0.000000000', '0.000000000', 12, '60608260605454818154548 2', 1.743098, 0]
['192.168.133.1', '5353', '5353', 'udp', '60', '-0.013894000', '-0.000164000', 20, '707092707064646060826060545464645454928 2', 3.204424, 0]
['192.168.42.1', '5353', '5353', 'udp', '60', '0.000000000', '0.000000000', 16, '606060605454707070706464645454', 1.702445, 0]
['192.168.1.2', '5353', '5353', 'udp', '60', '0.005689000', '0.005689000', 21, '828260605454606092707064647070646454549282', 3.20694, 0]
['192.168.56.1', '5353', '5353', 'udp', '60', '0.005355000', '0.005355000', 19, '606082606070709270705464645464645454 92', 2.4609550000000002,
['169.254.161.245', '5353', '5353', 'udp', '60', '0.005191000', '0.005057000', 20, '707092707064646060826060545464645454928 2', 3.209969, 0]
['169.254.200.128', '5353', '5353', 'udp', '60', '0.005102000', '0.005102000', 12, '60608260605454818154548 2', 1.743098, 0]
```

Рисунок 3.6 – Фрагмент сформованих статистичних параметрів

Модуль інтелектуального аналізу містить у собі попередню обробку даних. Перетворення даних, тобто заміна текстових даних на чисельні та нормалізація, пошук найбільшого значення для кожної колонки та їх ділення. Після таких дій всі дані будуть в проміжку від 0 до 1. Частина коду представлено нижче:

```
for row in data:
  max = [m if m > row[c] else row[c] for c, m in enumerate(max)]
for row in data:
  row = [row[c] / m for c, m in enumerate(max)]
```

Після виконання всіх перетворень, вхідний вектор даних для нейронної мережі готовий. Фрагмент вхідних даних НМ зображено на рис. 3.7.

```
[0.00832009737386149, 0.008249534450651769, 1.0, 1.0, 0.9512195121951219, 0.0, 0.0, 0.09523809523809523, 9.48976680556465e-39, 0.03612900934557312, 1.0]
[0.00011353816912943438, 1.0, 0.008249534450651769, 1.0, 1.0, 0.004660631524234417, 0.0046390108042389175, 1.0, 1.0, 0.9990563771799665, 1.0]
[0.00832009737386149, 0.008249534450651769, 1.0, 1.0, 0.7317073170731707, 1.0, 1.0, 0.09523809523809523, 9.48976680556465e-39, 0.03612900934557312, 1.0]
[0.011353817686927186, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.0, 0.0, 0.9523809523809523, 0.008537080025393037, 0.998272568
[0.011353817686927186, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.03820331788452376, 0.03802609228009968, 0.9523809523809523, 0
[0.011353817686927186, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.04568804955169576, 0.08350219447630051, 0.9523809523809523, 0
[0.011353817686927186, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.035050028154372595, 0.11838962517137608, 0.9523809523809523,
[0.9999997702686257, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.0, 0.0, 0.9523809523809523, 0.008537080025393037, 1.0, 1.0]
[0.9999997702686257, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.04489972711915797, 0.04469143679994481, 0.9523809523809523, 0.0
[0.9999997702686257, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.04321046476371985, 0.08770144775076956, 0.9523809523809523, 0.0
[0.9999997702686257, 0.09968342644320298, 0.09968342644320298, 0.5, 0.8536585365853658, 0.03731104084549747, 0.12483940227466737, 0.9523809523809523, 0.0
[0.001135384296842683, 0.09968342644320298, 0.09968342644320298, 0.5, 0.7317073170731707, 0.0, 0.0, 0.9047619047619048, 7.317535110076251e-05, 0.76666005
```

Рисунок 3.7 – Фрагмент вхідних даних НМ

Модуль інтелектуального аналізу передбачає аналіз результату попереднього модулю нейронною мережею, що включає обчислення вхідних даних на: вхідному шарі, прихованих та вихідному (рис. 3.8).

Рисунок 3.8 – Модуль інтелектуального аналізу

Результатами модулю інтелектуального аналізу має бути класифіковані дані, які передбачають різні типи DDoS-атак.

Модуль механізмів реакції містить у собі функції повідомлення та блокування відповідних результатів. Повідомлення передбачає візуальне попередження про ймовірнісну атаку, яке відображається у вікні. Блокування – реакція на задану кількість класифікованих типів атак. Блокування виконується за допомогою iptables – утиліта за допомогою якої адміністратори створюють і змінюються правила, що керують фільтрацією і перенаправленням пакетів. Частину коду представлено нижче:

```
subprocess.call(["iptables", "-A", "INPUT", "-s", arg, "-j", "DROP"])
```

Внесений IP-адрес за допомогою цього правила буде відкидати усі з'єднання.

3.3 Розробка та навчання нейронної мережі

Створення нейронної мережі складається з декількох етапів:

– підготовка навчальної вибірки;

– налаштування параметрів мережі;

– навчання НМ.

Формування навчальної вибірки є одною з найважливіших частин при проектуванні нейронної мережі. Тому її формування було наближено до реального. Для цього було здійснено моделювання атаки, яке описано в підрозділі 2.2.

Використовуючи модуль для аналізу пакетів, дані збережено у csv файлі для подальшої обробки. Для нормалізації даних, необхідно видалити рядки які повторюються, пусті значення змінити на 0 та текстові значення перетворити у числові. Розмірність вибірки становить 156 273 рядків, які розділені на 11 вхідних параметрів та 10 типів для класифікації: Normal 30%, SYN-flood 8%, UDP-flood 6%, MSSQL 5%, SSDP 9%, DNS 10%, LDAP 7%, SNMP 8%, TFTP 8%, NTP 9%. На рис. 3.9 зображено фрагмент даних з csv файлу.

	A	B	C	D	E	F	G	H	I	J	K
106979	172.16.0.5	43443	6652	udp	482	0.016558	0.000103	2450	1180900	4.022313	0
106980	172.16.0.5	54741	9712	udp	482	0.001372	0	1495	720590	2.549848	0
106981	140.82.114.26	443	53700	tcp	78	0	0	2	7860	0.115973	0
106982	192.168.133.1	5353	5353	udp	70	0.005274	0.009684	20	7.07E+39	3.204424	0
106983	169.254.161.245	57394	5355	udp	64	0	0	20	7.07E+39	3.209969	0
106984	192.168.1.2	5353	5353	udp	60	0.005689	0.005689	21	8.28E+41	3.20694	0

Рисунок 3.9 – Фрагмент необроблених даних файлу з csv файлу

Необроблені дані піддаються перетворенню та нормалізації як це було показано раніше.

Для моделювання архітектури нейронних мереж буде використовуватись python бібліотека TensorFlow. Це система машинного навчання другого покоління, створена для заміни DistBelief [47]. Мета цієї системи допомогти дослідникам і користувачам створити моделі для вирішення проблем обробки природної мови та розпізнавання зображень за допомогою бібліотек машинного навчання та графіків потоку даних. Вона також може бути використана у глибокому навчанні для створення нейромережевої архітектури та розробки алгоритмів машинного навчання.

Щоб розпочати моделювання, достатньо імпортувати саму бібліотеку TensoFlow та модуль keras у якому безпосередньо розташовані архітектури нейронних мереж. Частина коду представлено нижче:

```
from tensorflow import keras
```

Основним будівельним блоком нейронної мережі є шар. Шари витягують уявлення з даних, що надходять у них. Оголошення моделі Sequential означає, що мережа будує один шар за один раз. Перше, що потрібно зробити це забезпечити, щоб вхідний шар мав потрібну кількість вхідних параметрів. Це можна вказати під час створення вхідного шару за допомогою аргументу input_shape та де він буде становити 11, адже всього в нас 11 вхідних змінних.

Модель нейронної мережі прямого поширення представлено нижче:

```
model = keras.models.Sequential([
    keras.layers.Flatten(input_shape=train_data_shape),
    keras.layers.Dense(16, activation='sigmoid'),
    keras.layers.Dense(16, activation='sigmoid'),
    keras.layers.Dense(10, activation='softmax'),])
```

Повністю з'єднані шари у нейронній мережі визначаються за допомогою класу Dense. Ми можемо вказати кількість нейронів чи вузлів у шарі, як перший аргумент та вказати функцію активації як другий. Розроблена модель складається: з вхідного шару, двох прихованих з 16 нейроннами і функціями активації sigmoid та вихідного шару з функцією активації softmax.

Перш ніж модель буде готова до навчання, їй потрібно ще кілька налаштувань. Вони додаються під час кроку компіляції моделі:

– функція втрат – вимірює наскільки точна модель під час тренувань;

– оптимізатор – це те, як модель оновлюється на основі даних, які вона бачить і функції втрат.

– показники – використовується для контролю кроків навчання та тестування.

Частина коду представлено нижче:

```
model.compile(
    loss='categorical_crossentropy',
    optimizer='adam',
    metrics=['accuracy'])
```

Після модулювання та налаштування НМ можна приступати до навчання. Навчання моделі нейронної мережі включає наступних кроків:

1) Подати навчальні дані моделі.

2) Модель вчиться асоціювати вхідні вектор та мітки.

3) Модель робить передбачення щодо тестового набору.

Для початку навчання необхідно ввести команду model.fit з параметрами: вхідні дані, очікувані результати та кількість епох навчання. Частина коду представлено нижче:

```
model.fit(train_data, train_target, epochs=40)
```

Після виконання команди розпочнеться навчання з усіма параметрами які було вказано вище. Процес навчання нейронної мережі зображено на рис. 3.10.

```
153780/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
153970/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
154180/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
154380/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
154550/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
154730/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
154920/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
155080/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
155250/156273 [============================>.] - ETA: 0s - loss: 0.3455 - accuracy: 0.8383
155430/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
155620/156273 [============================>.] - ETA: 0s - loss: 0.3455 - accuracy: 0.8382
155820/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
156030/156273 [============================>.] - ETA: 0s - loss: 0.3453 - accuracy: 0.8383
156220/156273 [============================>.] - ETA: 0s - loss: 0.3454 - accuracy: 0.8383
```

Рисунок 3.10 – Процес навчання моделі НМ

При моделюванні інших мереж змінюється параметри, а структура зберігається: проектування моделі, налаштування та навчання.

Параметри для моделюванні рекурентної мережі LSTM представлено нижче:

```
model = keras.models.Sequential([
  keras.layers.Embedding(input_dim=train_data_shape),
  keras.layers.LSTM(30,activation='sigmoid',
       return_sequences=True, dropout=0.1,),
  keras.layers.LSTM(30,activation='sigmoid',
       return_sequences=True, dropout=0.1,),
  keras.layers.Dense(10, activation='softmax'),])
```

Архітектура рекурентної нейронної мережі LSTM зображено на рис. 3.11.

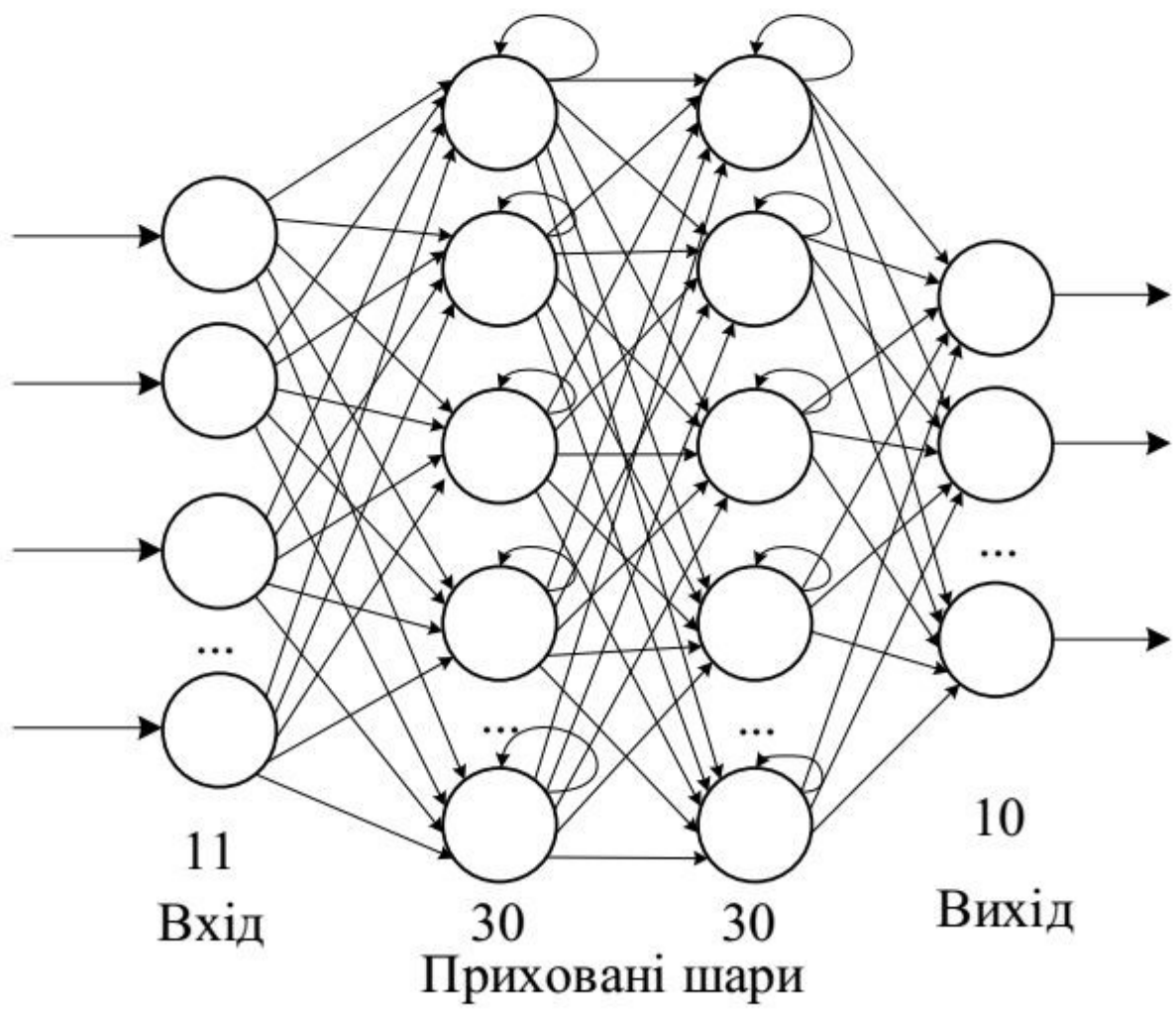


Рисунок 3.11 – Рекурентна нейронна мережа LSTM

Найкращі результати показала рекурентна мережа LSTM. Для перевірки похибки першого та другого роду було побудовано матрицю похибок з використанням бібліотеки matplotlib. Матрицю похибок для мережі LSTM зображено на рис. 3.12.

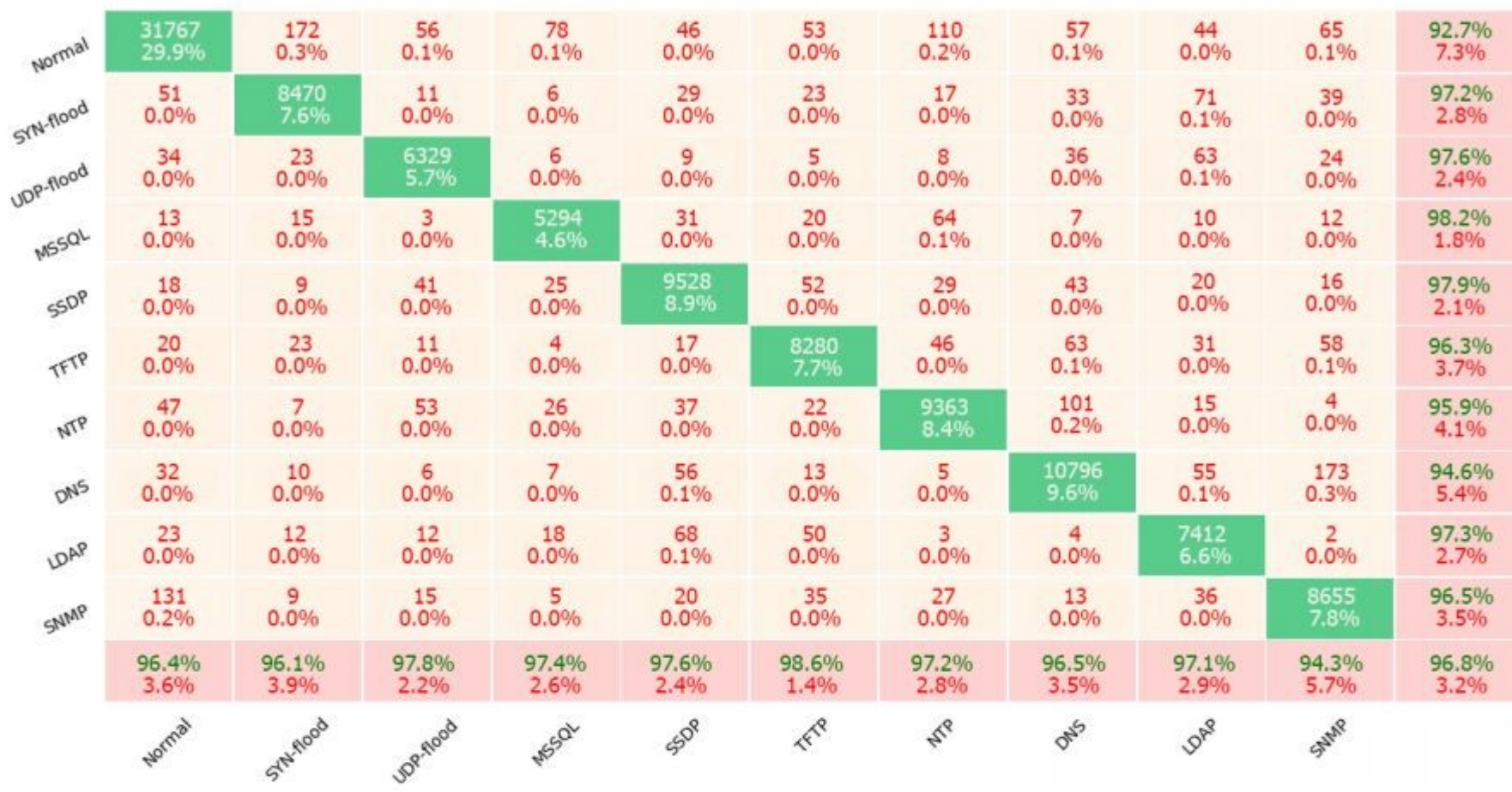


Рисунок 3.12 – Матриця похибок рекурентної мережі LSTM

Точність обраховувалась як відношення суми класифікованих даних порівнюючи з округленими даними нейронної мережі до загальної кількості правильних відповідей.

Результати спроектованих моделей зображено в табл. 3.1.

Таблиця 3.1 – Порівняльні характеристики нейронних мереж

Тип мережі	Кількість шарів/нейронів	Функції активації/метод навчання	Час навчання	Похибка 1-го роду	Похибка 2-го роду	Точність
FFNN	1/12	ReLu-Softmax/Cross_entr	3:08	24,3%	10,1%	65,6%
	2/16	Sigmoid-Sigmoid-Softmax/Cross_entr	2:54	6,5%	8,2%	85,3%
CNN	2/10	ReLu-ReLu-Softmax/Sigmoid-Cross_entropy	3:43	5,5%	6,1%	88,4%
	2/20	Sigmoid-Sigmoid-Softmax/ Cross_entr	5:50	4,0%	4,1%	91,9%
RNN (LSTM)	2/12	ReLu-ReLu-Softmax/ Cross_entr	4:11	4,6%	3,0%	92,4%
	2/30	Sigmoid-Sigmoid-Softmax/ Cross_entr	8:33	1,7%	1,5%	96,8%

Найкращі результати показала рекурентна мережа типу LSTM на вхід якої подаються 11 параметрів, архітектура мережі містить два приховані шари по 30 нейронів та функції активації sigmoid і softmax. Точність визначення такої мережі на тестовій виборці становить 96,8%.

3.4 Реалізація інтерфейсу та тестування

Для розробки інтерфейсу програмного засобу обрано бібліотеку pyqt5, яка дає змогу спроектувати будь-який інтерфейс до відповідної задачі. Одна з особливостей бібліотеки, що надихає розробників користуватися PyQt5 – це PyQt5 Designer, завдяки якому можна створювати складні GUI додатки досить швидко [49]. Потрібно тільки перетягувати свої віджети для створення власної форми.

Після того як інтерфейс було спроектовано, збережений файл ui необхідно конвертувати з розширенням py для модифікації кнопок та методів, які вони обробляють. Головне вікно програмного засобу зображено на рис. 3.12. Воно містить у собі таблицю куди відображаються запити, кнопки керування та вибору. Як перший параметр за замовчування обрано збереження перехопленого трафіку у файл.

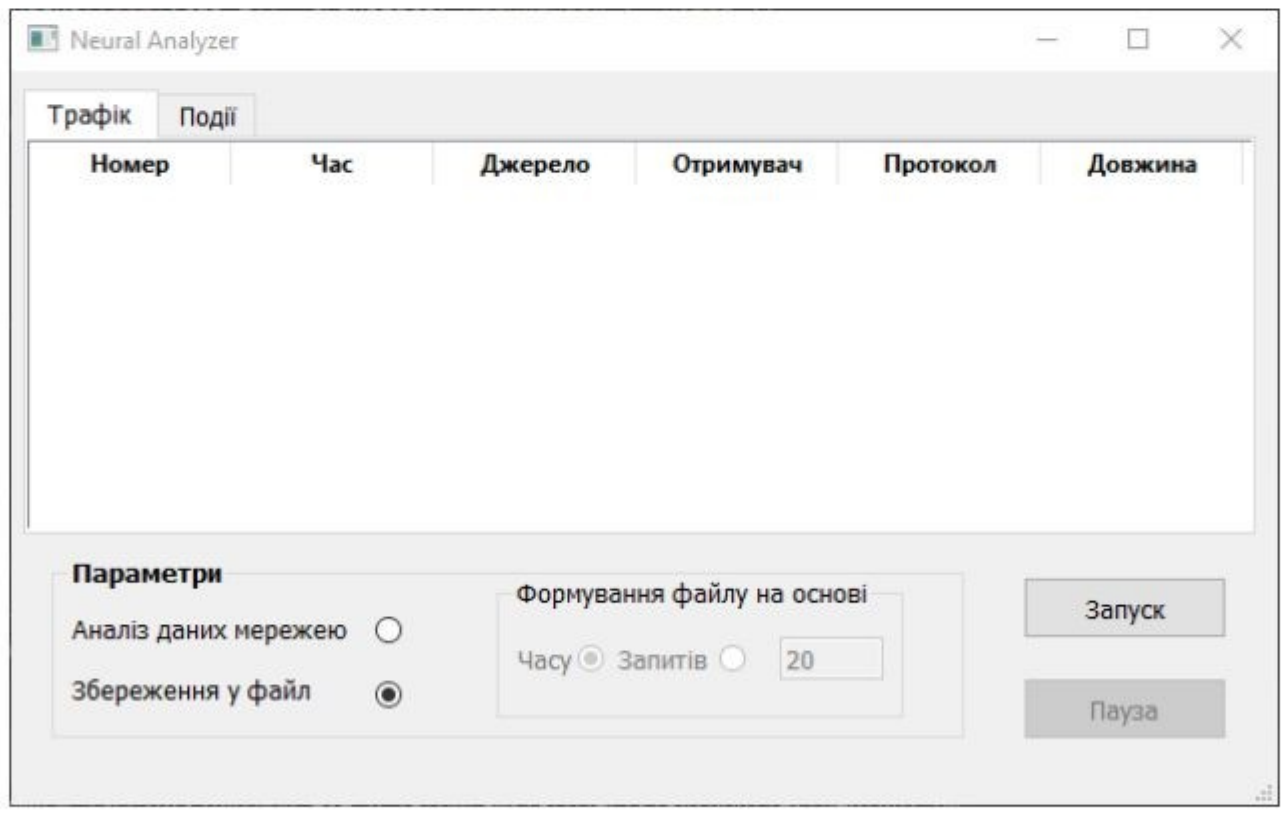


Рисунок 3.12 – Головне вікно програмного засобу

Другим параметром для вибору є аналіз даних нейронною мережею. Додатково можна обрати на протязі якого проміжку буде формуватися вхід дані

для мережі, які беруться з файлу. На вибір час – значення у секундах на протязі якого формуються файли або запити – максимальна кількість запитів (рис. 3.13).

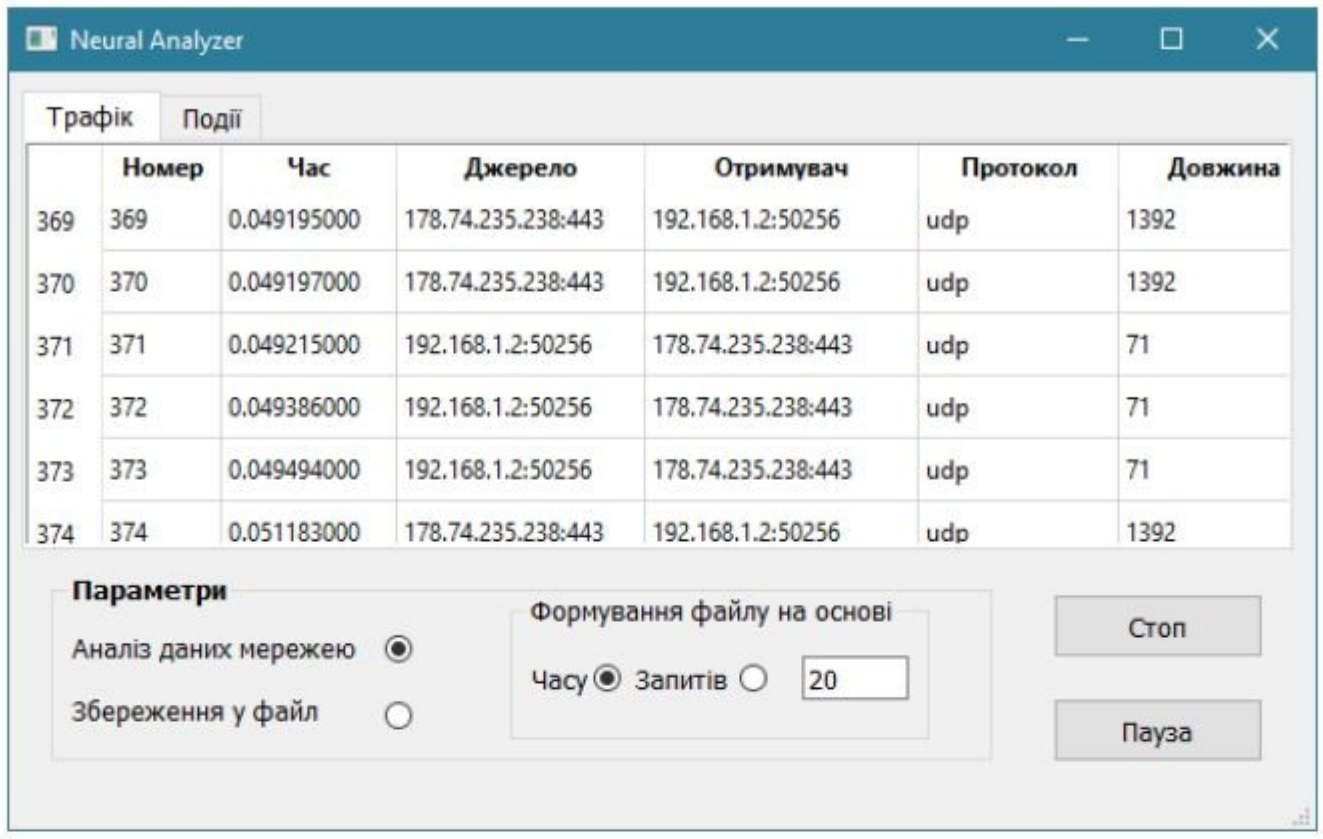


Рисунок 3.13 – Головне вікно процес сканування трафіку

Після запуску сканування та вибору параметрів у вкладці «Події» буде відображатись аналіз даних зроблений нейронною мережею. Вихідні дані показують: розмір файлу, дату, кількість типів атак та IP-адреси з яких вони здійснювались. Також доступно налаштування дій, які можна виконувати при ймовірній атаці: блокування після – кількості пакетів яких було зараховано до одного типу атаки та повідомити – спливаюче вікно яке вказує кількість атак після кожного проміжного сканування. Головне вікно з вкладкою «Події» зображено на рис. 3.14.

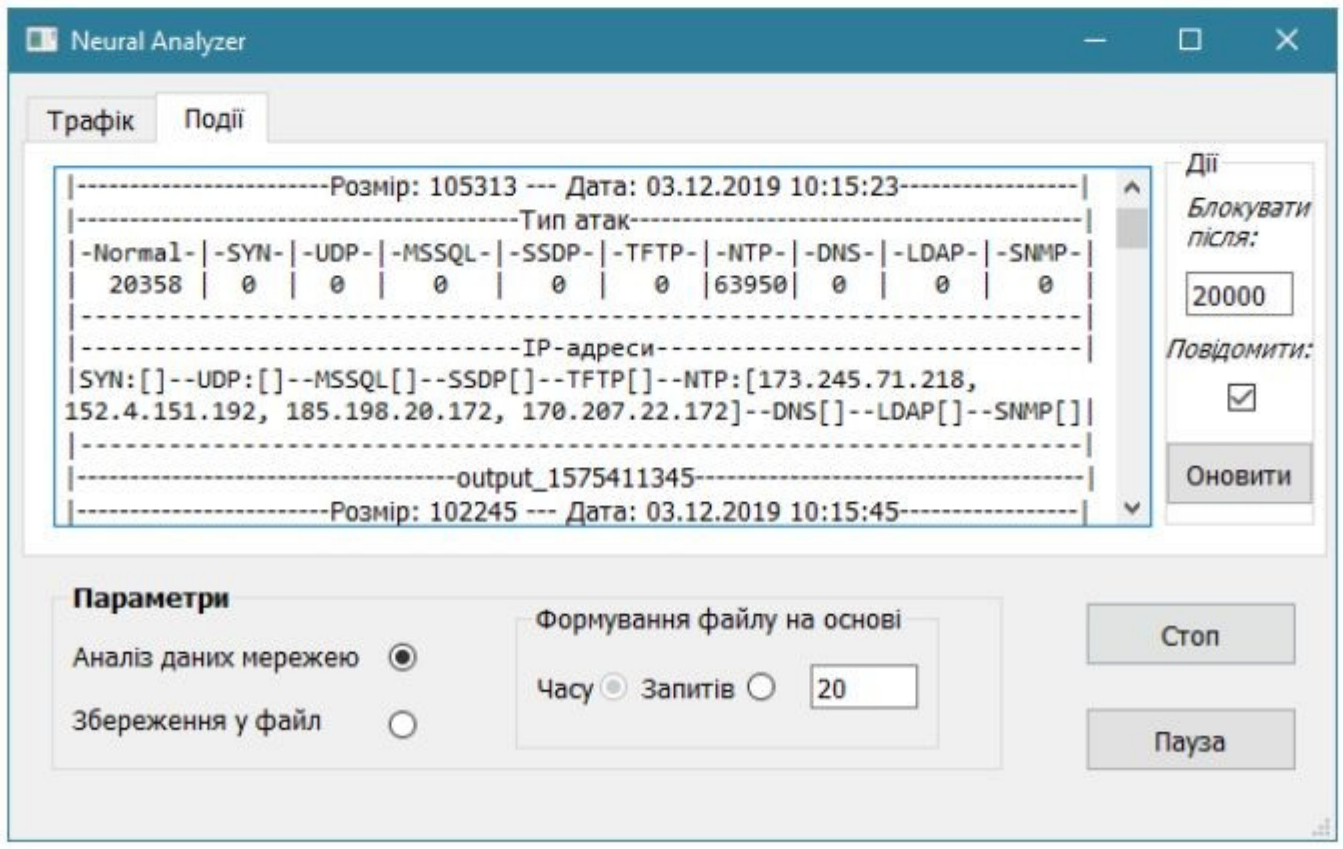


Рисунок 3.14 – Головне вікно з вкладкою «Події»

Після кожного проаналізованого файлу може виводитись повідомлення з інформацією про тип атаки та кількість запитів. Спливаюче вікно з інформацією зображено на рис. 3.15.

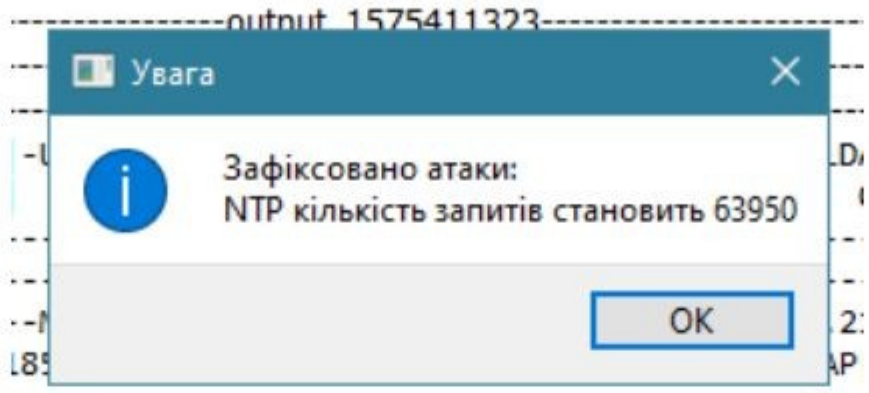


Рисунок 3.15 – Спливаюче вікно з інформацією

Під час запуску сканування, якщо модуль не виявить мережевий інтерфейс, або трафік на протязі певного часу буде відображено попередження про такого роду помилку (рис. 3.16).

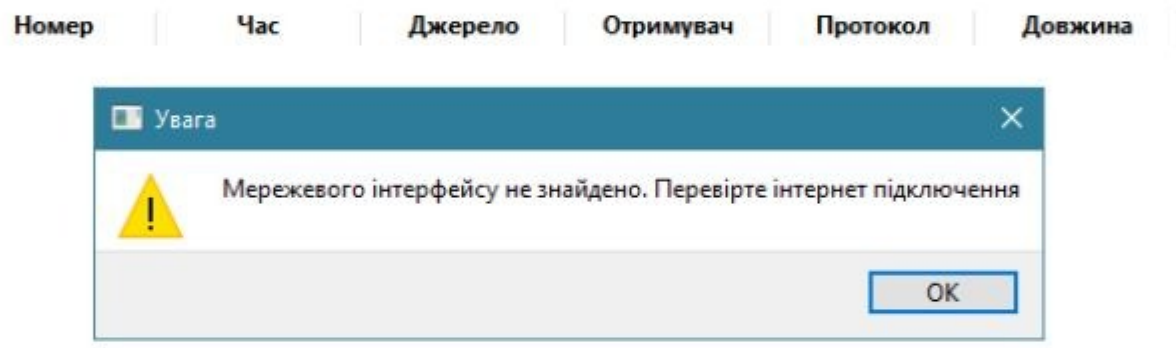


Рисунок 3.16 – Спливаюче вікно з попередженням

Під час сканування трафік збирається та зберігається у файл, який буде використовувати нейронна мережа. Якщо файл не сформувався, то вмикається перший запобіжник який очікує додатковий час на формування фалу. У раз його відсутності і після очікування повідомляється про помилку (рис. 3.17).

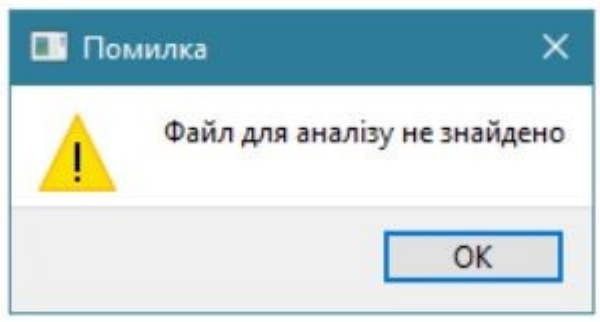


Рисунок 3.17 – Спливаюче вікно з попередженням

Проведені тексти показали, що програма працює відповідно до вимог, виконуючи завдання для яких була розроблена, але великі дані можуть викликати довшу обробку.

4 ЕКОНОМІЧНА ЧАСТИНА

Метою економічної частини магістерської кваліфікаційної роботи є довести економічну доцільність та ефективність впровадження інтелектуальної інформаційної технології для виявлення DDoS-атак, для цього необхідно виконати такі етапи:

- оцінити комерційний потенціал розробки;
- спрогнозувати витрати на виконання наукової роботи та впровадження її результатів;
- спрогнозувати комерційній ефект від реалізації результатів розробки;
- розрахувати ефективність вкладених інвестицій та період їх окупності.

4.1 Оцінювання економічного потенціалу розробки

Метою проведення технологічного аудиту є оцінювання комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Об'єктом дослідження магістерської кваліфікаційної роботи є розробка та програмна реалізація процесу виявлення DDoS-атак.

Для проведення технологічного аудиту залучено трьох незалежних експерти: к.т.н., доц. кафедри ЗІ Куперштейн Л. М., к.т.н., доц. кафедри ЗІ Войтович О. П., к.т.н. доц. кафедри ЗІ Баришев Ю. В. Кожен з експертів повинен ознайомитися з запропонованою розробкою, та заповнити таблицю, яка визначає рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можливу оцінку в балах. Після виконання цього, підраховується середньоарифметична сума балів та визначається який рівень комерційного потенціалу має нова розробка.

Оцінювання комерційного потенціалу розробки здійснюється за критеріями, наведеними в таблиці 4.1.

Таблиця 4.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки пові-домлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання комерційного потенціалу наведено в табл. 4.2.

Таблиця 4.2 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали експерта		
	Куперштейн Л.М.	Войтович О.П.	Баришев Ю.В.
	Бали, виставлені експертами:		
1	4	3	3
2	3	2	2
3	4	3	3
4	4	2	2
5	3	2	3
6	2	2	1
7	2	1	1
8	3	2	2
9	1	1	1
10	4	4	4
11	4	3	3
12	4	4	4
Сума балів	$СБ_1$=38	$СБ_2$=29	$СБ_3$=29
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^i СБ_i}{i} = \frac{38+29+29}{3} = 32$		

Отже, з отриманих даних таблиці 4.2 видно, що середньоарифметична сума балів дорівнює 32, тобто нова розробка має рівень комерційного потенціалу вище середнього.

Потенційними споживачами даної розробки є постачальники інтернет послуг та підприємств, які надають доступ до своїх сайтів, яким є критично важливим їх доступність у будь який час. До цього списку також можна віднести і фірми середнього розміру, які можуть не мати великого капіталу для забезпечення безпеки їх систем.

Реалізація розробки планується через спільне підприємство. Підприємство більше орієнтоване на прямий канал збуту, де розробка буде пропонуватися безпосередньо зацікавленим фірмам, через інтернет ресурс, де користувачі зможуть придбати товар та одразу його завантажити.

4.2 Прогнозування витрат на виконання науково-дослідної

Прогнозування витрат на виконання науково-дослідної чи дослідно-конструкторської роботи може складатися з таких етапів:

1-й етап: розрахунок витрат, які безпосередньо стосуються виконавців даного розділу роботи.

2-й етап: розрахунок загальних витрат на виконання даної роботи.

3-й етап: прогнозування загальних витрат на виконання та впровадження результатів даної роботи.

1-й етап. Розрахунок витрат можна здійснити за такими статтями та формулами:

1) Основна заробітна плата із розробників (дослідників) $З_О$, якщо вони працюють в наукових установах бюджетної сфери:

$$З_о = \frac{М}{Т_р} \cdot t \text{ грн.}, \tag{4.1}$$

де М – місячний посадовий оклад конкретного розробника, наукового керівника;

$Т_р$ – кількість робочих днів у місяці, $Т_р = 23$ дні.

t – число робочих днів роботи розробника (дослідника).

Заробітна плата розробника:

$$З_Р = \frac{11000}{23} \cdot 65 = 31086{,}95 \text{ (грн)}$$

Заробітна плата наукового керівника проекту:

$$З_{НК} = \frac{9000}{23} \cdot 22 = 8608{,}69 \text{ (грн)}$$

Витрати на оплату праці, основна заробітна плата:

$$З_о = З_Р + З_{НК} = 31086{,}95 + 8608{,}69 = 39695{,}64 \text{ (грн)}$$

Таблиця 4.3 – Розрахунки основної заробітної плати спеціаліста

Найменування посади виконавця	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи, t	Витрати на оплату праці, грн.
Розробник	11000	478,26	65	31086,95
Науковий керівник проекту	9000	391,3	22	8608,69
Всього:				39696,64

2) Додаткова заробітна плата $З_Д$ всіх розробників та робітників, які брали участь у виконанні даної роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_Д = (0,1 \ldots 0,12) \cdot З_О \tag{4.2}$$

Додаткова заробітна плата усіх робітників та розробників:

$$З_Д = 0,12 \cdot 39696,64 = 4763,59 \text{ (грн)}$$

3) Нарахування на заробітну плату $Н_{ЗП}$ розробників та робітників, які брали участь у виконанні даної роботи, розраховуються за формулою:

$$Н_{ЗП} = (З_О + З_Р + З_Д) \cdot \frac{\beta}{100}, \tag{4.3}$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_д$ – додаткова заробітна плата розробників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, %.

З 01.01.2016 року ставка єдиного внеску на загальнообов'язкове державне соціальне страхування встановлено 22%.

Нарахування на заробітну плату:

$$Н_{ЗП} = (39696,64 + 4763,59) \cdot \frac{22}{100} = 9781,25 \text{ (грн)}$$

4) Амортизаційні відрахування обладнання можна розрахувати за формулою:

$$А = \frac{Ц \cdot Н_а}{100} \cdot \frac{Т}{12} \text{ грн.}, \tag{4.4}$$

де Ц – загальна балансова вартість обладнання, Ц = 9800 грн.;

$Н_а$ – річна норма амортизаційних відрахувань. Для нашого випадку можна прийняти, що $Н_а$ = 20%;

Т – термін використання обладнання, Т = 3 міс.

Амортизаційні відрахування для персонального комп'ютера становить:

$$А = \frac{9800 \cdot 20}{100} \cdot \frac{3}{12} = 490 \text{ (грн)}$$

5) Витрати на матеріали М, що були використанні під час виконання роботи, розраховуються по кожному виду матеріалів за формулою:

$$М = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \text{ грн.}, \quad (4.5)$$

де $Н_i$ – витрати матеріалу *i*-го найменування;

$Ц_i$ – вартість матеріалу *i*-го найменування, грн./кг.;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

$В_i$ – маса відходів матеріалу *i*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *i*-го найменування, грн/кг;

n – кількість видів матеріалів.

Таблиця 4.4 – Вартість матеріалів, що були використані для розробки ПЗ

Найменування матеріалу	Одиниці виміру	Ціна, грн.	Витрачено	Вартість витрачених матеріалів, грн.
Флешка	Шт.	240	1	240
Пачка паперу	Шт.	75	65	48,75
Ручка	Шт.	22	1	22
Всього				310,75

Загальна вартість витрат становить:

$$М = 310,75 \cdot 1,1 = 341,82 \text{ (грн)}$$

6) Витрати на силову електроенергію $В_е$, розраховується за формулою:

$$В_е = В \cdot П \cdot Ф \cdot К_П \text{ грн.}, \quad (4.6)$$

де В – вартість 1кВт-год. електроенергії, В = 2,44 грн/кВт;

П – установлена потужність ноутбука, П = 0,06 кВт;

Ф – фактична кількість годин роботи ноутбука, $Ф = 7\ год. \cdot 65\ днів = 455$;

Кп – коефіцієнт використання потужності, $К_п = 0,75$.

Витрати на силову енергію становлять:

$$В_е = 2,44 \cdot 0,06 \cdot 455 \cdot 0,75 = 49,95\ (грн)$$

7) Інші витрати $І_в$ можна прийняти як (100…300%) від суми основної заробітної плати розробників, тобто:

$$В_{ін} = (1...3) \cdot З_о \tag{4.7}$$

Інші витрати становлять:

$$В_{ін} = 2 \cdot 39696,64 = 79393,28\ (грн)$$

8) Сума усіх витрат становить:

$$В_{ін} = 39696,64 + 4763,59 + 9781,25 + 490 + 341,82 + 49,95 + 79393,28 = 134516,53\ (грн)$$

2-й етап. Розрахунок загальних витрат на виконання роботи. Загальна вартість роботи визначається за $В_{заг}$ формулою:

$$В_{заг} = \frac{В}{\alpha} \tag{4.8}$$

де α – частка витрат, які безпосередньо здійснює виконавець даного етапу роботи, у відн. одиницях = 1.

$$В_{заг} = \frac{134516,53}{1} = 134516,53\ (грн)$$

3-й етап. Прогнозування загальних витрат на виконання та впровадження результатів. Прогнозування загальних витрат ЗВ на виконання та впровадження результатів здійснюється за формулою:

$$ЗВ = \frac{В_{заг}}{\beta}, \tag{4.9}$$

де β – коефіцієнт, який характеризує етап (стадію) виконання даної роботи, на стадії розробки промислового зразка, $\beta \approx 0,7$.

$$ЗВ = \frac{134516,53}{0,7} = 192166,47\ (грн)$$

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

Виконання наукової роботи та впровадження її результатів буде здійснюватися протягом одного року. Позитивні результати від впровадження розробки очікуються протягом трьох років від впровадження розробки. Одним із основних позитивних результатів є зростання величини прибутку. Зростання чистого прибутку забезпечить підприємству (організації) надходження додаткових коштів, які дозволять покращити фінансові результати діяльності.

Збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від впровадження розробки, розраховується за формулою:

$$\Delta П_i = \sum_{1}^{n} (\Delta П_я \cdot N + П_я \cdot \Delta N)_i, \qquad (4.10)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження результатів розробки у даному році;

N – основний кількісний показник, який визначає діяльність підприємства у даному році до впровадження результатів наукової розробки;

ΔN – покращення основного кількісного показника діяльності підприємства від впровадження результатів розробки;

$П_я$ – основний якісний показник, який визначає діяльність підприємства у даному році після впровадження результатів наукової розробки;

n – кількість років, протягом яких очікується отримання позитивних результатів від впровадження розробки.

В результаті впровадження результатів наукової розробки витрати матеріалів на розробку алгоритму зменшаться на 250 грн (що автоматично спричинить збільшення чистого прибутку підприємства на 250 грн), а кількість користувачів збільшиться: протягом першого року – на 200 користувачів, протягом другого року – на 150 користувачів, протягом третього року – на 100 користувачів.

Реалізація продукції до впровадження результатів наукової розробки складала 600 шт., а прибуток, що його отримувало підприємство (організація) на одиницю продукції до впровадження результатів наукової розробки – 400 грн.

Спрогнозуємо збільшення чистого прибутку від впровадження результатів наукової розробки у кожному році відносно базового.

Збільшення чистого продукту $\Delta П_i$ протягом першого року складатиме:

$$\Delta П_1 = 250 \cdot 600 + (400 + 250) \cdot 200 = 280000 \text{ (грн)}$$

Протягом другого року:

$$\Delta П_2 = 250 \cdot 600 + (400 + 250) \cdot (200 + 150) = 377500 \text{ (грн)}$$

Протягом третього року:

$$\Delta П_3 = 250 \cdot 600 + (400 + 250) \cdot (200 + 150 + 100) = 442500 \text{ (грн)}$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

В даному підрозділі необхідно кількісно спрогнозувати, яку вигоду, зиск можна отримати у майбутньому від впровадження результатів виконаної робти.

Розрахунок ефективності вкладених інвестицій передбачає проведення таких робіт:

1-й крок. Розрахуємо теперішню вартість інвестицій PV, що вкладаються в наукову розробку. Такою вартістю ми можемо вважати прогнозовану величину загальних витрат ЗВ на виконання та впровадження результатів НДДКР, розраховану раніше, тобто будемо вважати, що ЗВ = PV = $192166{,}47$ (грн).

2-й крок. Розрахуємо очікуване збільшення прибутку $\Delta П_i$, що його отримає підприємство (організація) від впровадження результатів наукової розробки, для кожного із років, починаючи з першого року впровадження. Таке збільшення прибутку також було розраховане нами раніше та становить:

$$\Delta П_1 = 280000 \text{ (грн)}, \quad \Delta П_2 = 377500 \text{ (грн)}, \quad \Delta П_3 = 442500 \text{ (грн)}.$$

3-й крок. Для спрощення подальших розрахунків необхідно побудувати вісь часу, на яку наносять всі платежі (інвестиції та прибутки), що мають місце під час виконання науково-дослідної роботи та впровадження її результатів.

Якщо загальні витрати ЗВ на виконання та впровадження результатів НДДКР (або теперішня вартість інвестицій PV) дорівнюють 192166,47 грн., а результати вкладених у наукову розробку інвестицій почнуть виявлятися вже вкінці другого року впровадження. То ці результати виявляться у тому, що у першому році підприємство отримає збільшення чистого прибутку на 280000 грн. відносно базового року, у другому році – збільшення чистого прибутку на 377500 грн. (відносно базового року), у третьому році – збільшення чистого прибутку на 442500 грн. (відносно базового року).

Тоді рисунок, що характеризує рух платежів (інвестицій та додаткових прибутків) буде мати вигляд, наведений на рис. 4.1.

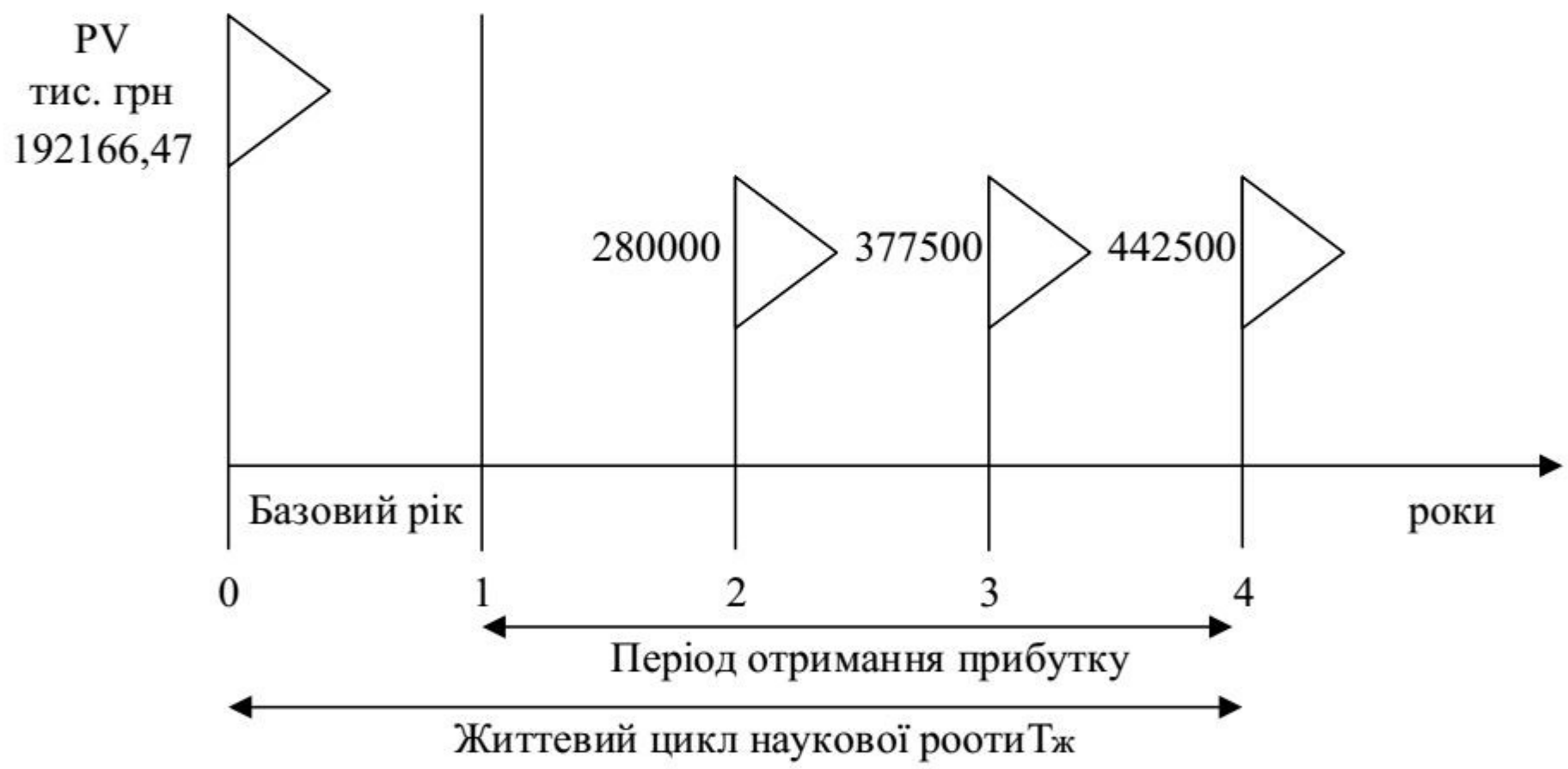


Рисунок 4.1 – Вісь часу з фіксацією платежів, що мають місце під час розробки та впровадження результатів НДДКР

4-й крок. Розрахуємо абсолютну ефективність вкладених інвестицій $E_{абс}$. Якщо $E_{абс} \leq 0$ то результати від проведення наукових досліджень та їх впровадження буде збитковим і вкладати гроші в проведення цих досліджень ніхто не буде.

Для цього показника скористаємося формулою:

$$E_{абс} = (ПП - PV), \quad (4.11)$$

де ПП – приведена вартість всіх чистих прибутків, що їх отримає підприємство (організація) від реалізації результатів наукової розробки, грн.;

PV – теперішня вартість інвестицій PV = ЗВ = 192166,47 грн.

У свою чергу, приведена вартість всіх чистих прибутків ПП розраховується за формулою:

$$ПП = \sum_{1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.12)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному із років, протягом яких виявляються результати виконаної та впровадженої НДДКР, грн;

т – період часу, протягом якого виявляються результати впровадженої НДДКР, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні; для України цей показник знаходиться на рівні 0,1;

t – період часу (в роках) від моменту отримання чистого прибутку до точки „0”.

$$ПП = \frac{280000}{(1+0,1)^2} + \frac{377500}{(1+0,1)^3} + \frac{442500}{(1+0,1)^4} = 231404,95 + 283621,33 + 302233,45 = 817259,73 \text{ (грн)}$$

$$E_{абс} = (817259,73 - 192166,47) = 625093,26 \text{ (грн)}$$

Оскільки $E_{абс} > 0$, то вкладання коштів на виконання та впровадження результатів НДДКР може бути доцільним.

5-й крок. Розрахуємо відносну (щорічну) ефективність вкладених в наукову розробку інвестицій $E_в$. Для цього використаємо формулу:

$$E_в = \sqrt[T_ж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.13)$$

де $E_{абс}$ – абсолютна ефективність вкладених інвестицій, грн;

PV –теперішня вартість інвестицій PV = ЗВ, грн;

$T_ж$ – життєвий цикл наукової розробки, роки.

Далі, розрахована величина $E_в$ порівнюється з мінімальною (бар'єрною) ставкою дисконтування $\tau_{мін}$, яка визначає ту мінімальну дохідність, нижче за яку інвестиції вкладатися не будуть. У загальному вигляді мінімальна (бар'єрна) ставка дисконтування $\tau_{мін}$ визначається за формулою:

$$\tau = d + f \tag{4.14}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2019 році в Україні d = (0,14...0,2);

f – показник, що характеризує ризикованість вкладень; зазвичай, величина f = (0,05...0,1), але може бути і значно більше.

Якщо величина $E_в > \tau_{мін}$, то інвестор може бути зацікавлений у фінансуванні даної наукової розробки. В іншому випадку фінансування наукової розробки здійснюватися не буде.

Спочатку спрогнозуємо величину $\tau_{мін}$. Припустимо, що за даних умов $\tau_{мін} = 0,18 + 0,07 = 0,25$.

Тоді відносна (щорічна) ефективність вкладних інвестицій в проведення наукових досліджень та впровадження їх результатів складе:

$$E_в = \sqrt[4]{1 + \frac{625093,26}{192166,47}} - 1 = \sqrt[4]{4,25} - 1 = 1,43 - 1 = 0,43 \text{ або } 43\%$$

Оскільки $E_в = 43\% > \tau_{мін} = 25\%$, то інвестор буде зацікавлений вкладати гроші в дану наукову розробку.

6-й крок. Розраховують термін окупності вкладених у реалізацію наукового проекту інвестицій. Термін окупності вкладених у реалізацію наукового проекту інвестицій $T_{ОК}$ можна розрахувати за формулою:

$$T_{ОК} = \frac{1}{E_в} \tag{4.15}$$

Якщо $T_{ОК}$ < 3…5-ти років, то фінансування даної наукової розробки в принципі є доцільним. В інших випадках потрібні додаткові розрахунки та обґрунтування.

Термін окупності становить:

$$T_{ОК} = \frac{1}{0,43} = 2,32 \quad (4.16)$$

$T_{ОК}$ < 5 років, що свідчить про доцільність фінансування даної наукової розробки.

Рівень комерційного потенціал інтелектуальної інформаційної технології для виявлення DDoS-атак є вище середнього. Показники ефективності показують, що даний метод є доцільним і буде цікавий для інвестора. Термін окупності розробленого проекту менше 5-ти років, що підтверджує доцільність вкладання коштів в дану розробку.

ВИСНОВКИ

Результатом виконання магістерської кваліфікаційної роботи є інформаційна технологія виявлення DDoS-атак на основі нейронної мережі LSTM та програмний засіб для виявлення різних типів атак. Було проведено аналіз літературних джерел який показав, що DDoS-атаки є актуальними до сьогодні. З кожним роком збитки від таких атак зростають, що призводить до попиту на розробку інформаційних технологій для протидії від них. Розглянуто структури атак та їх типи, що дає змогу зрозуміти, який шаблон вони мають. Проаналізовані інтелектуальні підходи захисту, які мітять у собі цілу низку методів до створення технології виявлення та протидії, яка б базувалась на машинному навчанні. Інформаційне забезпечення яке дає змогу реалізувати майже будь-яку DDoS-атаку стає все простішим у використанні. Використання скриптів зводить все до виконання автоматизованих команд, для виконання яких не потрібно знати алгоритм чи технічну реалізацію.

Для формування набору даних було змодельовано атаки наближені до реальних. Було використано скрипт MASSCAN, який збирав IP-адреси з відповідними відкрити портам. Скрипт Saddam тестував зібрані адреси, фільтрував їх та на їх основі здійснював атаку. Створено модуль перехоплення та аналізу пакетів, який передбачає сканування вхідного трафіку, формування статистичних параметрів та звичайних. Досліджено існуючі підходи для виявлення DDoS-атак. Майже половина дослідників використовує застарілий набір даних, який містить не актуальні атаки. Інші вказують майже стовідсоткову точність виявлення не демонструючи параметри для навчання чи дані.

Для реалізації програмного забезпечення було обрано мову програмування Python, який містить у собі велику кількість відкритих бібліотек, що дає використовувати його майже у будь-яких задачах. Найпопулярнішою бібліотекою для побудови нейронної мережі є TensorFlow, який дає змогу моделювати та тестувати розроблено архітектуру за декілька хвилин. Для кращого відлагодження програму було розбито на модулі. Модуль перехоплення передбачає сканування мережевого трафіку та зберігання у файл. Модуль аналізу пакетів отримує дані з

файлу, обирає поля та формує статистичні дані. Модуль інтелектуального аналізу передбачає попередню нормалізацію даних для нейронної мережі та безпосередній їх аналіз мережою. Модуль механізмів реакції містить у собі повідомлення про ймовірнісну атаку та блокування за допомогою iptables утиліти.

Різні типи архітектур для нейронної мережі було змодельовано: FNN, CNN, LSTM RNN. Найкращі результати показала мережа LSTM точність якої становить 96,8%. При аналізі результатів матриці похибок, помилка першого роду, коли мережа розпізнає нормальний трафік становить 1,7%, другого роду 1,5%, що є малими значеннями для такої вибірки.

При розрахунку витрат на розробку програмного продукту, експлуатаційних витрат, обсягу робіт при використанні програмного продукту визначено, що розробка і впровадження програмного продукту є економічно доцільним і виправданим.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. DDoS-атаки: реальна небезпека віртуального світу [Електронний ресурс]. – Режим доступу: http://zillya.ua/ddos-ataki-realna-nebezpeka-virtualnogo-svitu – Назва з екрану.
2. До проблеми формування набору даних для дослідження DDoS-атак. Конференція ВНТУ електронні наукові видання, XLVIII Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії / Б. В. Кульчицький, Л. М. Куперштейн – ВНТУ, 2019.
3. DDoS attacks in 2019 [Електронний ресурс]. – Режим доступу: https://securelist.com/ddos-report-in-q3-2019/89700/ – Назва з екрану.
4. Github biggest attack [Електронний ресурс]. – Режим доступу: https://blogs.akamai.com/2019/03/github-fueled-13-tbps-attacks.html
5. NETSCOUT Arbor's 14th Annual Worldwide Infrastructure Security Reporthttps [Електронний ресурс]. – Режим доступу : https://www.netscout.com/report/ – Назва з екрану
6. Принципи організації DDoS-атак [Електроний ресурс] – Режим доступу: http://hi-news.pp.ua/tehnka-tehnologyi/2883-ddos-ataka-yak-zrobiti-programi-dlya-ddos-atak.html – Назва з екрану
7. A. Asosheh and N. Ramezani, «A comprehensive taxonomy of DDoS attacks and defense mechanism applying in a smart classification» WSEAS Transactions on Computers, vol. 7, no. 4, pp. 281–290, 2008.
8. Voytovych O.P. Investigation of denial-of-service attacks / O.P. Voytovych, E.I. Kolibabchuk, L.M. Kupershtein // Вісник ХНУ. Технічні науки − №3. – 2016. – С. 129−133.
9. K. M. Prasad, A. R. Reddy, and K. V. Rao, "DoS and DDoS attacks: Defense, detection and traceback mechanisms survey," Global J. Comput. Sci. Technol., vol. 14, no. 7, pp. 1019, 2017.
10. E. Doron, and A. Wool, WDA: A Web farm Distributed Denial of Service attack attenuator, Computer Networks, vol. 55, no. 5, pp. 1037-1051, April 2011.
11. M. Aamir and M. A. Zaidi, ``A survey on DDoS attack and defense strategies: From traditional schemes to current techniques," Interdiscipl. Inf. Sci., vol. 19, no. 2, pp. 173-200, 2013.
12. G. Somani, M. S. Gaur, D. Sanghi, M. Conti, and R. Buyya, ``DDoS attacks in cloud computing: Issues, taxonomy, and future directions," Comput. Commun., vol. 107, pp. 30-48, Jul. 2017.
13. N. S. Rao, K. C. Sekharaiah, and A. A. Rao, ``A survey of distributed denial-of-service DDoS-attacks," Innov. Comput. Sci. Eng., vol. 32, pp. 221-230, May 2019.

14. S. M. Mousavi, ``Early detection of DDoS attacks in software deepened networks controller," Ph.D dissertation, Dept. Elect. Comput. Eng., Carleton University Ottawa, Ottawa, Canada, 2014.
15. K. M. Prasad, A. R. Reddy, and K. V. Rao, "DoS and DDoS attacks: Defense, detection and traceback mechanisms" Global J. Comput. Sci. Technol., vol. 14, no. 7, pp. 1-19, 2014.
16. M. Uysal and S. R. Ranjan Swaminathan, ``DDoS-resilient scheduling to counter attacks under imperfect detection," in Proc. 25TH IEEE Int. Conf. Comput. Commun., Apr. 2006, pp. 1-13.
17. D. Kshirsagar, S. Sawant, A. Rathod, and S. Wathore, 2016, CPU Load Analysis and Minimization for TCP SYN Flood Detection, Procedia Computer Science, 85, International Conference on Computational Modelling and Security (CMS 2016), vol. 85, pp. 626-633, June 2016.
18. IMPERVA INCAPSULA, DDoS AttackAvailable [Електроний ресурс] – Режим доступу: https://www.incapsula.com/ddos/ddos-attacks
19. A. A. Acharya and K. M. Arpitha, An Intrusion Detection System Against UDP Flood Attack and Ping of Death Attack (DDOS) in MANET, International Journal of Engineering and Technology (IJET), Vol 8 No 2, 2016.
20. A. Saied, R. Overill, and T. Radzik, Neurocomputing, Detection of known and unknown DDoS-attacks using Artificial Neural Networks, vol. 172, no.1, pp. 385-393, Jan 2016.
21. J. E. Varghese and B. Muniyal, "An investigation of classification algorithms for intrusion detection system – a quantitative approach," in 2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Sept 2017, pp. 2045–2051
22. M. A. M. Hasan, M. Nasser, S. Ahmad, and K. I. Molla, "Feature selection for intrusion detection using fuzzy logic," Journal of information security, vol. 7, no. 03, p. 129, 2016.
23. A. Ferriyan, A. H. Thamrin, K. Takeda, and J. Murai, "Feature selection using genetic algorithm to improve classification in network intrusion detection system," in 2017 International Electronics Symposium on Knowledge Creation and Intelligent Computing (IES-KCIC), Sept 2017, pp. 46–49.
24. S. H. Li, Y. C. Kao, Z. C. Zhang, Y. P. Chuang, and D. C. Yen, ``A network behavior-based botnet detection mechanism using PSO and K-means," ACM Trans. Manage. Inf. Syst., vol. 6, no. 1, p. 3, Apr. 2015.
25. B. J. Radford, L. M. Apolonio, A. J. Trias, and J. A. Simpson, "Network traffic anomaly detection using neural networks," CoRR, vol. abs/1803.10769, 2018. [Електроний ресурс] – Режим доступу: http://arxiv.org/abs/1803.10769

26. Y. Yu, M. Abadi, P. Barham, E. Brevdo, M. Burrows, A. Davis, J. Dean, S. Ghemawat, T. Harley, P. Hawkins et al., "Dynamic control flow in large-scale soft agents," in Proceedings of the Thirteenth EuroSys Conference. ACM, 2018, p. 18.
27. Y. Li, R. Ma, and R. Jiao, "A hybrid malicious code detection method based on deep learning," International Journal of Security and its Applications, vol. 9, no. 5, pp. 205–216, 2015.
28. M. Tavallaee, E. Bagheri, W. Lu, and A. Ghorbani, A Detailed Analysis of the Data Sets, Submitted to Second IEEE Symposium on Computational Intelligence for Security and Defense Applications (CISDA), 2017.
29. R. R. R. Robinson and C. Thomas, "Ranking of machine learning algorithms based on the performance in classifying DDoS attacks," 2015 IEEE Recent Advances in Intelligent Computational Systems (RAICS), Trivandrum, 2015, pp. 185-190.
30. Low Orbit Ion Cannon (LOIC) [назва з екрану]. – Режим доступу до джерела: https://github.com/NewEraCracker/LOIC
31. Sourceforge, Xoic Tool to make (D)DoS attacks [назва з екрану]. – Режим доступу до джерела: https://sourceforge.net/directory/os:windows/?q=xoic
32. Sourceforge, DDOSIM - Layer 7 DDoS Simulator [назва з екрану]. – Режим доступу до джерела: https://sourceforge.net/projects/ddosim/
33. Sourceforge, R-U-Dead-Yet? (RUDY) [назва з екрану]. – Режим доступу до джерела: https://sourceforge.net/projects/ r-u-dead-yet/
34. Saddam [назва з екрану]. – Режим доступу до джерела: https://sourceforge.net/projects/saddam/
35. MASSCAN[назва з екрану]. – Режим доступу до джерела: https://sourceforge.net/projects/masscan/
36. Memcrashe DDoS exploit [назва з екрану]. – Режим доступу до джерела:https://github.com/649/Memcrashed-DDoS-Exploit
37. aSYNcron[назва з екрану]. – Режим доступу до джерела:https://github.com/649/a-syn-cron
38. M. Kim, H. Na, K. Chae, H. Bang, and J. Na: A Combined Data Mining Approach for DDoS Attack Detection, Lecture Notes in Computer Science, Vol. 3090, pp. 943-950, 2004.
39. G. Loukas, and O. Gulay. "Likelihood ratios and recurrent random neural networks in detection of denial of service attacks." 2007.
40. X. Yuan, C. Li and X. Li, "DeepDefense: Identifying DDoS Attack via Deep Learning," 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, 2017, pp. 1-8.
41. A. B. M. A. A. Islam and T. Sabrina, "Detection of various denial of service and Distributed Denial of Service attacks using RNN ensemble," 2009 12th International Conference on Computers and Information Technology, Dhaka, 2009, pp. 603-608.

42. The Most Popular Programming Languages of 2019 [назва з екрану]. – Режим доступу до джерела: https://blog.newrelic.com/technology/most-popular-programming-languages-of-2019/
43. About Python [назва з екрану]. – Режим доступу до джерела: https://www.python.org/about/apps/
44. What is java? [назва з екрану]. – Режим доступу до джерела: https://java.com/en/download/faq/whatis_java.xml
45. About C# [назва з екрану]. – Режим доступу до джерела: https://www.webopedia.com/TERM/C/C_sharp.html
46. What is MATLAB [назва з екрану]. – Режим доступу до джерела: https://www.mathworks.com/discovery/what-is-matlab.html
47. Why TensorFlow? [назва з екрану]. – Режим доступу до джерела: https://www.tensorflow.org/about
48. PyCharm features [назва з екрану]. – Режим доступу до джерела: https://www.jetbrains.com/pycharm/features/
49. PyQt5-tools Desighner [назва з екрану]. – Режим доступу до джерела: https://build-system.fman.io/pyqt5-tutorial
50. Методичні вказівки до виконання студентами-магістрантами економічної частини магістерських кваліфікаційних робіт / Уклад. В. О. Козловський – Вінниця: ВНТУ, 2012. – 22 с.
51. Козловський В. О. Економіка, організація виробництва та менеджмент в дипломних роботах. Навчальний посібник / В. О. Козловський – Вінниця: ВНТУ, 2004. – 94 с.
52. Козловський В. О. Техніко-економічні обґрунтування та економічні розрахунки в дипломних проектах та роботах. Навчальний посібник / В. О. Козловський – Вінниця: ВНТУ, 2003. – 75 с

ДОДАТКИ

Додаток А

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації

ЗАТВЕРДЖУЮ
Зав. кафедри ЗІ, д. т. н., проф
_____ В. А. Лужецький
«___» _____ 2019 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу на тему:
«Інтелектуальна інформаційна технологія виявлення DDoS-атак»

08-20.МКР.009.00.000 ТЗ

Керівник магістерської кваліфікаційної роботи
к.т.н., доц. кафедри ЗІ
_____ Л. М. Куперштейн
«___» _____ 2019 р.

1 Підстави для проведення робіт

Робота проводиться на підставі наказу № 254 ректора ВНТУ від 02.10.2019 р.
Дата початку роботи 01.09.19 р.
Дата закінчення роботи 12.12.19 р.

2 Мета та призначення НДР

Метою роботи є підвищення точності виявлення DDoS-атак на основі нейромережевого підходу.
Об'єктом дослідження є процес виявлення DDoS-атак.
Предметом дослідження є методи та засоби виявлення DDoS-атак.
Актуальність. Сучасні кібератаки, які завдають великого збитку компаніям, вважаються віддаленими, серед яких найбільш небезпечною є атака на відмову в обслуговуванні, а особливо – їх розподілена реалізація – DDoS-атака (Distributed Denial of Service). Ця атака популярна своєю простотою, численними відкритими відомостями про її реалізації і нечисленними обчислювальними ресурсами. DDoS-атаки є активним інструментом конкурентної боротьби , а також інформаційних воєн.

3 Мета та призначення розробки

Вхідними даними НДР є:
3.1 DDoS-атаки: реальна небезпека віртуального світу [Електронний ресурс]. – Режим доступу: http://zillya.ua/ddos-ataki-realna-nebezpeka-virtualnogo-svitu – Назва з екрану.
3.2 Кульчицький Б. В. До проблеми формування набору даних для дослідження DDoS-атак. Конференція ВНТУ електронні наукові видання, XLVIII Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії / Б. В. Кульчицький, Л. М. Куперштейн – ВНТУ, 2019.
3.3 DDoS attacks in 2019 [Електронний ресурс]. – Режим доступу: https://securelist.com/ddos-report-in-q3-2019/89700/ – Назва з екрану.
3.4 NETSCOUT Arbor's 14th Annual Worldwide Infrastructure Security Reporthttps [Електронний ресурс]. – Режим доступу : https://www.netscout.com /report/ – Назва з екрану

4 Виконавці НДР

Студент групи 1БС-18м Кульчицький Богдан Володимирович

5 Вимоги до виконання НДР

Для розширення можливостей розробленого засобу необхідно розв'язати такі задачі:

– проаналізувати сучасні типи DDoS-атак та інтелектуальні методи захисту;

– розробити інформаційну технологію;

– згенерувати необхідний набір даних;

– виконати моделювання різних архітектур нейронних мереж;

– розробити програмний засіб та провести тестування.

6 Вимоги до програмної документації

Графічна і текстова документація повинна відповідати діючим стандартам України.

7 Етапи НДР

Робота з теми виконується у 8 етапів.

Зміст етапу	Початок	Закінчення	Очікувані результати	Звітна документація
Аналіз завдання. Вступ	01.09.2019	04.09.2019	Вступ	Чернетка вступу
Розробка технічного завдання	05.09.2019	15.09.2019	Технічне завдання	Проект технічного завдання
Аналіз літературних джерел за напрямком магістерської кваліфікаційної роботи	15.09.2019	10.10.2019	Аналіз існуючих аналогів. Аналіз відомих методів. Постановка завдання	Чернетка першого розділу
Розробка інтелектуальної інформаційної технології	11.10.2019	25.10.2019	Розроблено інформаційну технологію на основі рекурентної нейронної мережі	Чернетка другого розділу
Розробка засобу, що реалізує аналіз даних на основі запропонованої інформаційної технології	26.10.2019	01.11.2019	Розроблений засіб, що реалізує аналіз даних на основі розробленого методу	Чернетка третього розділу
Експериментальні дослідження	02.11.2019	05.11.2019	Експериментально перевірено розроблений засіб	Чернетка четвертого розділу
Розробка економічного розділу	06.11.2019	09.11.2019	Економічні показники дослідження	Чернетка економічного розділу
Оформлення пояснювальної записки	10.11.2019	20.12.2019	Пояснювальна записка	Пояснювальна записка

8 Очікувані результати та порядок реалізації НДР

Передбачається розробка нового (удосконалення існуючого) методу, який спрямований на підвищення достовірності виявлення DDoS-атак в інтернет-сервісах. Заплановане створення програмного засобу, який може бути використаний установами, органами державної влади, які є суб'єктами забезпечення кібербезпеки, суб'єктами сектору безпеки і охорони.

9 Матеріали які подаються після закінчення НДР

По завершенню роботи подається пояснювальна записка та ілюстративна частина.

10 Порядок приймання НДР та її етапів

Апробація на науково-технічних конференціях. Результати роботи будуть розглядатися на засіданні ДЕК із захисту магістерських кваліфікаційних робіт.
Попередній захист та доопрацювання МКР грудень 2019 р.
Представлення МКР до захисту 17 грудня 2019 р.

11 Вимоги до розроблення документації

Документація буде виконуватись за допомогою комп'ютерного набору у відповідності вимог ДСТУ 3008-95. «Документація. Звіти у сфері науки і техніки. Структура і правила оформлення»

12 Вимоги щодо технічного захисту інформації з обмеженим доступом

У зв'язку з тим що дана робота не містить інформації, що потребує захисту у відповідності до законів України, заходи з її технічного захисту не передбачаються.

Розробив студент групи 1БС-18м _____ Кульчицький Б.В.

Додаток Б
Лістинг програми

Інтерфейс програмного засобу

```
import capture_module as cm
from PyQt5 import QtCore, QtGui, QtWidgets
from PyQt5.QtWidgets import QMessageBox
class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(629, 363)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.tabs = QtWidgets.QTabWidget(self.centralwidget)
        self.tabs.setGeometry(QtCore.QRect(6, 9, 621, 221))
        font = QtGui.QFont()
        font.setPointSize(10)
        self.tabs.setFont(font)
        self.tabs.setObjectName("tabs")
        self.tab = QtWidgets.QWidget()
        self.tab.setObjectName("tab")
        self.tableWidget = QtWidgets.QTableWidget(self.tab)
        self.tableWidget.setGeometry(QtCore.QRect(0, 0, 611, 201))
        self.tableWidget.setAutoFillBackground(True)
        self.tableWidget.setVerticalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOff)
        self.tableWidget.setHorizontalScrollBarPolicy(QtCore.Qt.ScrollBarAlwaysOff)
        self.tableWidget.setSizeAdjustPolicy(QtWidgets.QAbstractScrollArea.AdjustIgnored)
        self.tableWidget.setEditTriggers(QtWidgets.QAbstractItemView.NoEditTriggers)
        self.tableWidget.setCornerButtonEnabled(False)
        self.tableWidget.setObjectName("tableWidget")
        self.tableWidget.setColumnCount(6)
        self.tableWidget.setRowCount(0)
        item = QtWidgets.QTableWidgetItem()
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        item.setFont(font)
        self.tableWidget.setHorizontalHeaderItem(0, item)
```

```
item = QtWidgets.QTableWidgetItem()
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
item.setFont(font)
self.tableWidget.setHorizontalHeaderItem(1, item)
item = QtWidgets.QTableWidgetItem()
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
item.setFont(font)
self.tableWidget.setHorizontalHeaderItem(2, item)
item = QtWidgets.QTableWidgetItem()
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
item.setFont(font)
self.tableWidget.setHorizontalHeaderItem(3, item)
item = QtWidgets.QTableWidgetItem()
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
item.setFont(font)
self.tableWidget.setHorizontalHeaderItem(4, item)
item = QtWidgets.QTableWidgetItem()
font = QtGui.QFont()
font.setBold(True)
font.setWeight(75)
item.setFont(font)
self.tableWidget.setHorizontalHeaderItem(5, item)
self.tabs.addTab(self.tab, "")
self.tab_2 = QtWidgets.QWidget()
self.tab_2.setObjectName("tab_2")
self.textEdit = QtWidgets.QTextEdit(self.tab_2)
self.textEdit.setGeometry(QtCore.QRect(13, 10, 521, 171))
self.textEdit.setObjectName("textEdit")
self.groupBox_3 = QtWidgets.QGroupBox(self.tab_2)
self.groupBox_3.setGeometry(QtCore.QRect(540, 0, 71, 181))
```

```
self.groupBox_3.setObjectName("groupBox_3")
self.label_5 = QtWidgets.QLabel(self.groupBox_3)
self.label_5.setGeometry(QtCore.QRect(10, 20, 61, 31))
font = QtGui.QFont()
font.setPointSize(9)
font.setBold(False)
font.setItalic(True)
font.setWeight(50)
self.label_5.setFont(font)
self.label_5.setObjectName("label_5")
self.lineEdit_2 = QtWidgets.QLineEdit(self.groupBox_3)
self.lineEdit_2.setGeometry(QtCore.QRect(10, 60, 51, 21))
self.lineEdit_2.setObjectName("lineEdit_2")
self.label_6 = QtWidgets.QLabel(self.groupBox_3)
self.label_6.setGeometry(QtCore.QRect(0, 80, 71, 31))
font = QtGui.QFont()
font.setPointSize(9)
font.setBold(False)
font.setItalic(True)
font.setWeight(50)
self.label_6.setFont(font)
self.label_6.setObjectName("label_6")
self.checkBox = QtWidgets.QCheckBox(self.groupBox_3)
self.checkBox.setGeometry(QtCore.QRect(30, 110, 81, 20))
self.checkBox.setText("")
self.checkBox.setObjectName("checkBox")
self.monitorButton_3 = QtWidgets.QPushButton(self.groupBox_3)
self.monitorButton_3.setGeometry(QtCore.QRect(0, 140, 71, 31))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(False)
font.setWeight(50)
self.monitorButton_3.setFont(font)
self.monitorButton_3.setObjectName("monitorButton_3")
self.tabs.addTab(self.tab_2, "")
self.monitorButton = QtWidgets.QPushButton(self.centralwidget)
self.monitorButton.setGeometry(QtCore.QRect(510, 250, 101, 31))
font = QtGui.QFont()
```

```
font.setPointSize(10)
font.setBold(False)
font.setWeight(50)
self.monitorButton.setFont(font)
self.monitorButton.setObjectName("monitorButton")
self.groupBox = QtWidgets.QGroupBox(self.centralwidget)
self.groupBox.setGeometry(QtCore.QRect(20, 240, 451, 91))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(True)
font.setWeight(75)
self.groupBox.setFont(font)
self.groupBox.setObjectName("groupBox")
self.label = QtWidgets.QLabel(self.groupBox)
self.label.setGeometry(QtCore.QRect(10, 20, 141, 31))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(False)
font.setWeight(50)
self.label.setFont(font)
self.label.setObjectName("label")
self.label_2 = QtWidgets.QLabel(self.groupBox)
self.label_2.setGeometry(QtCore.QRect(10, 50, 141, 31))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(False)
font.setWeight(50)
self.label_2.setFont(font)
self.label_2.setObjectName("label_2")
self.radioButton = QtWidgets.QRadioButton(self.groupBox)
self.radioButton.setGeometry(QtCore.QRect(160, 26, 20, 21))
self.radioButton.setText("")
self.radioButton.setObjectName("radioButton")
self.radioButton_2 = QtWidgets.QRadioButton(self.groupBox)
self.radioButton_2.setGeometry(QtCore.QRect(160, 60, 16, 17))
self.radioButton_2.setText("")
self.radioButton_2.setChecked(True)
self.radioButton_2.setObjectName("radioButton_2")
```

```
self.groupBox_2 = QtWidgets.QGroupBox(self.groupBox)
self.groupBox_2.setEnabled(False)
self.groupBox_2.setGeometry(QtCore.QRect(220, 10, 201, 71))
font = QtGui.QFont()
font.setBold(False)
font.setUnderline(False)
font.setWeight(50)
font.setStrikeOut(False)
font.setKerning(True)
self.groupBox_2.setFont(font)
self.groupBox_2.setObjectName("groupBox_2")
self.label_3 = QtWidgets.QLabel(self.groupBox_2)
self.label_3.setGeometry(QtCore.QRect(10, 20, 31, 41))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(False)
font.setUnderline(False)
font.setWeight(50)
font.setStrikeOut(False)
font.setKerning(True)
self.label_3.setFont(font)
self.label_3.setObjectName("label_3")
self.radioButton_3 = QtWidgets.QRadioButton(self.groupBox_2)
self.radioButton_3.setEnabled(False)
self.radioButton_3.setGeometry(QtCore.QRect(40, 20, 20, 41))
self.radioButton_3.setText("")
self.radioButton_3.setChecked(True)
self.radioButton_3.setObjectName("radioButton_3")
self.label_4 = QtWidgets.QLabel(self.groupBox_2)
self.label_4.setGeometry(QtCore.QRect(60, 20, 51, 41))
font = QtGui.QFont()
font.setPointSize(10)
font.setBold(False)
font.setUnderline(False)
font.setWeight(50)
font.setStrikeOut(False)
font.setKerning(True)
self.label_4.setFont(font)
```

```
        self.label_4.setObjectName("label_4")
        self.radioButton_4 = QtWidgets.QRadioButton(self.groupBox_2)
        self.radioButton_4.setGeometry(QtCore.QRect(110, 20, 20, 41))
        self.radioButton_4.setText("")
        self.radioButton_4.setObjectName("radioButton_4")
        self.lineEdit = QtWidgets.QLineEdit(self.groupBox_2)
        self.lineEdit.setGeometry(QtCore.QRect(140, 30, 51, 21))
        self.lineEdit.setObjectName("lineEdit")
        self.monitorButton_2 = QtWidgets.QPushButton(self.centralwidget)
        self.monitorButton_2.setEnabled(False)
        self.monitorButton_2.setGeometry(QtCore.QRect(510, 300, 101, 31))
        font = QtGui.QFont()
        font.setPointSize(10)
        font.setBold(False)
        font.setWeight(50)
        self.monitorButton_2.setFont(font)
        self.monitorButton_2.setCheckable(False)
        self.monitorButton_2.setFlat(False)
        self.monitorButton_2.setObjectName("monitorButton_2")
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtWidgets.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 629, 21))
        self.menubar.setObjectName("menubar")
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtWidgets.QStatusBar(MainWindow)
        self.statusbar.setObjectName("statusbar")
        MainWindow.setStatusBar(self.statusbar)
        self.retranslateUi(MainWindow)
        self.tabs.setCurrentIndex(1)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)
        self.monitorButton.clicked.connect(self.monitor)
        self.radioButton.clicked.connect(self.radio_checked_neural)
        self.radioButton_2.clicked.connect(self.radio_checked_file)
        # self.radioButton_3.clicked.connect(self.pop_up_error)
    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "Neural Analyzer"))
        self.tableWidget.setSortingEnabled(True)
```

```
        item = self.tableWidget.horizontalHeaderItem(0)
        item.setText(_translate("MainWindow", "Номер"))
        item = self.tableWidget.horizontalHeaderItem(1)
        item.setText(_translate("MainWindow", "Час"))
        item = self.tableWidget.horizontalHeaderItem(2)
        item.setText(_translate("MainWindow", "Джерело"))
        item = self.tableWidget.horizontalHeaderItem(3)
        item.setText(_translate("MainWindow", "Отримувач"))
        item = self.tableWidget.horizontalHeaderItem(4)
        item.setText(_translate("MainWindow", "Протокол"))
        item = self.tableWidget.horizontalHeaderItem(5)
        item.setText(_translate("MainWindow", "Довжина"))
        self.tabs.setTabText(self.tabs.indexOf(self.tab), _translate("MainWindow",
"Трафік"))
        self.groupBox_3.setTitle(_translate("MainWindow", "Дії"))
        self.label_5.setText(_translate("MainWindow", "Блокувати \n"
                                                      "після:"))
        self.label_6.setText(_translate("MainWindow", "Повідомити:"))
        self.monitorButton_3.setText(_translate("MainWindow", "Оновити"))
        self.tabs.setTabText(self.tabs.indexOf(self.tab_2), _translate("MainWindow",
"Події"))
        self.monitorButton.setText(_translate("MainWindow", "Запуск"))
        self.groupBox.setTitle(_translate("MainWindow", "Параметри"))
        self.label.setText(_translate("MainWindow", "Аналіз даних мережею"))
        self.label_2.setText(_translate("MainWindow", "Збереження у файл"))
        self.groupBox_2.setTitle(_translate("MainWindow", "Формування файлу на основі"))
        self.label_3.setText(_translate("MainWindow", "Часу"))
        self.label_4.setText(_translate("MainWindow", "Запитів"))
        self.monitorButton_2.setText(_translate("MainWindow", "Пауза"))
    def monitor(self):
        data = cm.traffic_file()
        for row_num, row in enumerate(data):
            for cell_num, cell in enumerate(row):
                num_rows = self.tableWidget.rowCount()
                self.tableWidget.insertRow(num_rows)
                self.tableWidget.setItem(row_num, cell_num,
                                         QtWidgets.QTableWidgetItem(cell))
        self.monitorButton.setText("Стоп")
        self.monitorButton_2.setEnabled(True)
```

```
    def radio_checked_neural(self):
        if self.radioButton.isChecked():
            self.groupBox_2.setEnabled(True)
    def radio_checked_file(self):
        if self.radioButton_2.isChecked():
            self.groupBox_2.setEnabled(False)
if __name__ == "__main__":
    import sys
    app = QtWidgets.QApplication(sys.argv)
    MainWindow = QtWidgets.QMainWindow()
    ui = Ui_MainWindow()
    ui.setupUi(MainWindow)
    MainWindow.show()
    sys.exit(app.exec_())
```

Модуль перехоплення

```
import pyshark
import datetime
import time

TIMESTAMP = int(datetime.datetime.now().timestamp())
PACKETS_PATH = "resources/output_" + str(TIMESTAMP)
INPUT_FILE_PATH = "resources/output"
PACKETS_TIME = 20
statistics = []
def capturing_module():
    print("Start")
    live_capture = pyshark.LiveCapture(output_file=PACKETS_PATH)
    print(live_capture.sniff(packet_count=PACKETS_TIME))
    print("Over")
def compare_fields(main, stat):
    return main[0] == stat.get("ip") and main[1] == stat.get("srcport") and main[2] ==
stat.get("dstport")
def combine_params(main_fields):
    full_params = []
    stats_names = ["packet_count", "length", "time_frame", "bps"]
    for fields in main_fields:
        for stats in statistics:
```

```
            if compare_fields(fields, stats):
                temp = fields.copy() + list(map(lambda x: stats.get(x), stats_names))
                full_params.append(temp)
    return full_params
def formatting_data(data):
    for row in data:
        row[0] = int(row[0].replace(".", ""))
        row[1] = int(row[1])
        row[2] = int(row[2])
        row[3] = 0.5 if row[3] == "udp" else 1.0
        row[4] = int(row[4])
        row[5] = float(row[5])
        row[6] = float(row[6])
        row[8] = float(row[8])
        row[9] = float(row[9])
    return data
def normalization(data):
    max_value = [0 * i for i in range(11)]
    for row in data:
        max_value = [m if m > row[c] else row[c] for c, m in enumerate(max_value)]
    for row in data:
        row = [row[c] / m for c, m in enumerate(max_value)]
        print(row)
def check_param(packet):
    all_params = True
    protocol = ''
    if 'ip' not in packet:
        all_params = False
    else:
        if 'tcp' in packet:
            protocol = 'tcp'
        elif 'udp' in packet:
            protocol = 'udp'
        else:
            all_params = False
    return all_params, protocol
def packet_statistics(packet, ip, srcport, dstport, length, time_frame):
    global statistics
```

```
    time_frame = float(time_frame)
    index = 0
    exist = False
    for i, dic in enumerate(statistics):
        if dic['ip'] == ip and dic['srcport'] and dic['dstport']:
            exist = True
            index = i
            break
    if not exist:
        statistics.append({'ip': ip, 'srcport': srcport, 'dstport': dstport,
'packet_count': 1,
                           'length': length, 'time_frame': time_frame, 'bps': 1})
    else:
        dic_params = statistics[index]
        dic_params['packet_count'] = dic_params.get('packet_count') + 1
        dic_params['length'] = dic_params.get('length') + length
        dic_params['time_frame'] = dic_params.get('time_frame') + time_frame
def analyze_file():
    packet_fields = []
    packets_count = 0
    file_packets = pyshark.FileCapture(input_file=INPUT_FILE_PATH)
    for packet in file_packets:
        packets_count += 1
        all_params, protocol = check_param(packet)
        if all_params:
            packet_fields.append([packet.ip.addr, packet[protocol].srcport,
packet[protocol].dstport,
                                  packet[protocol].layer_name, packet.length,
packet[protocol].time_delta,
                                  packet[protocol].time_relative])
            packet_statistics(packet, packet.ip.addr, packet[protocol].srcport,
packet[protocol].dstport, packet.length,
                              packet[protocol].time_relative)
    formatted = formatting_data(combine_params(packet_fields))
    normalization(formatted)
def traffic_file():
    count = 0
    data = list()
    file_packets = pyshark.FileCapture(input_file=INPUT_FILE_PATH)
    for packet in file_packets:
```

```
        trans = packet.transport_layer.lower()
        data.append([packet.number, packet.frame_info.time_relative, packet["ip"].addr +
":" + packet[trans].srcport,
                     packet["ip"].dst + ":" + packet[trans].dstport, trans,
                     packet.length])
        if count == 400:
            break
        else:
            count += 1
    return data
```

Модуль нейронної мережі

```
from __future__ import absolute_import
from __future__ import division
from __future__ import print_function
from datetime import datetime
from packaging import version
from keras.models import Sequential
from keras.layers import Dense, LSTM
from input_module as im
model = keras.models.Sequential([
       keras.layers.Embedding(input_dim=im.train_data_shape),
       keras.layers.LSTM(30,activation='sigmoid',
            return_sequences=True, dropout=0.1,),
       keras.layers.LSTM(30,activation='sigmoid',
            return_sequences=True, dropout=0.1,),
       keras.layers.Dense(10, activation='softmax'),])
       model.compile(
           loss='categorical_crossentropy',
           optimizer='adam',
           metrics=['accuracy'])
       model.fit(train_data, train_target, epochs=110)
```

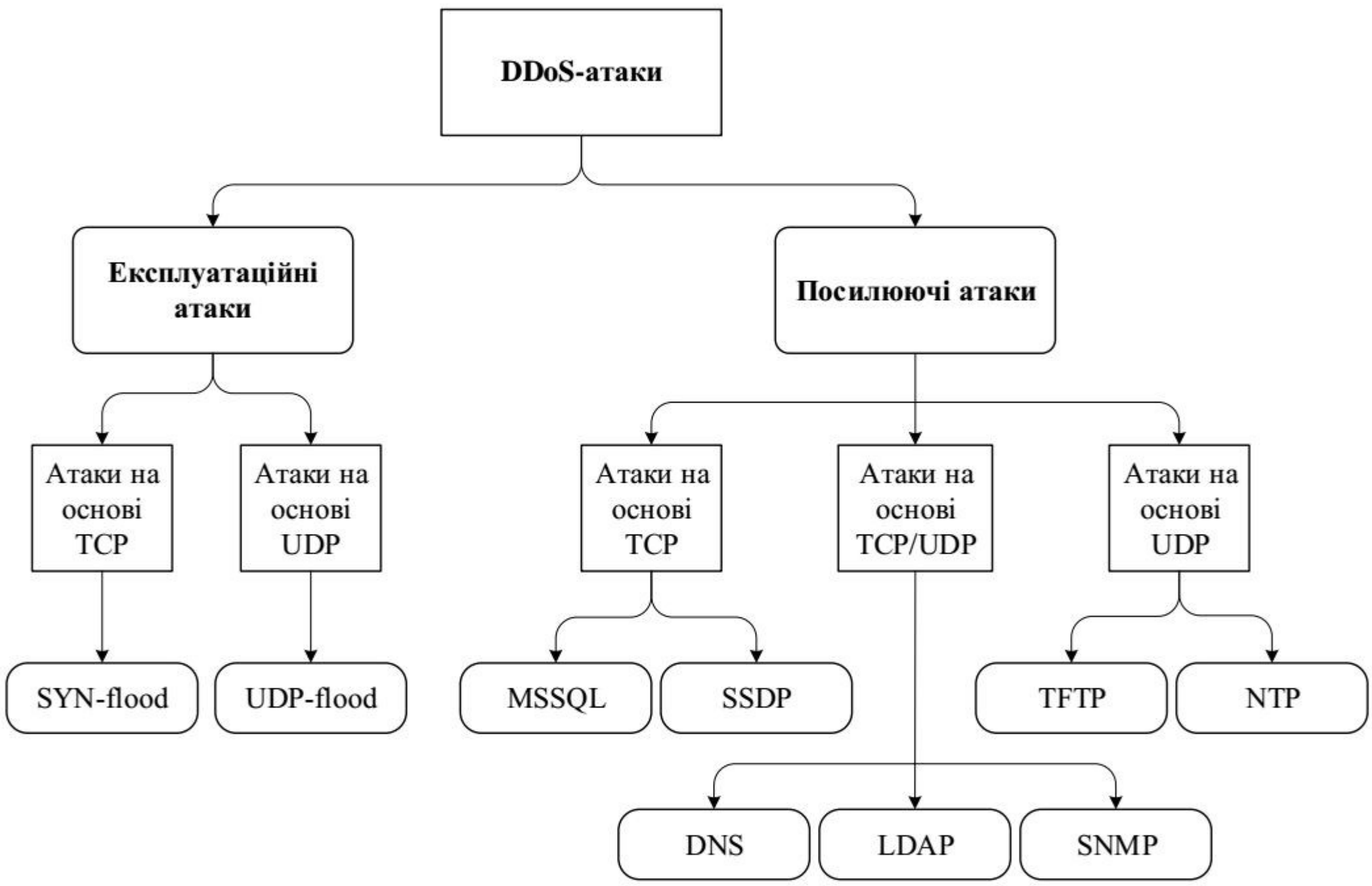
Модуль блокування

```
import sys
import getopt
import subprocess

def main(argv):
       try:
              opts, args = getopt.getopt(argv, "ha:d:lv", ["help", "add=", "delete=", "list",
"version"])
```

```
        if len(opts) == 0:
                if len(args) > 0:
                        opts = []
                        for arg in args:
                                opts.append(("-a", arg))
                else:
                        opts = [("-h","")]

        for opt, arg in opts:
                if opt in ("-h", "--help"):
                        help()
                elif opt in ("-v", "--version"):
                        print version()
                elif opt in ("-a", "--add"):
                        subprocess.call(["iptables",  "-A",  "INPUT",  "-s",  arg,  "-j",
"DROP"])
                elif opt in ("-d", "--delete"):
                        subprocess.call(["iptables",  "-D",  "INPUT",  "-s",  arg,  "-j",
"DROP"])
                elif opt in ("-l", "--list"):
                        subprocess.call(["iptables", "-L", "-n"])
    except getopt.GetoptError, err:
            print "blockip error: " + str(err)
            sys.exit(2)
    except:
            print "blockip error (unknown)."
            sys.exit(2)
```

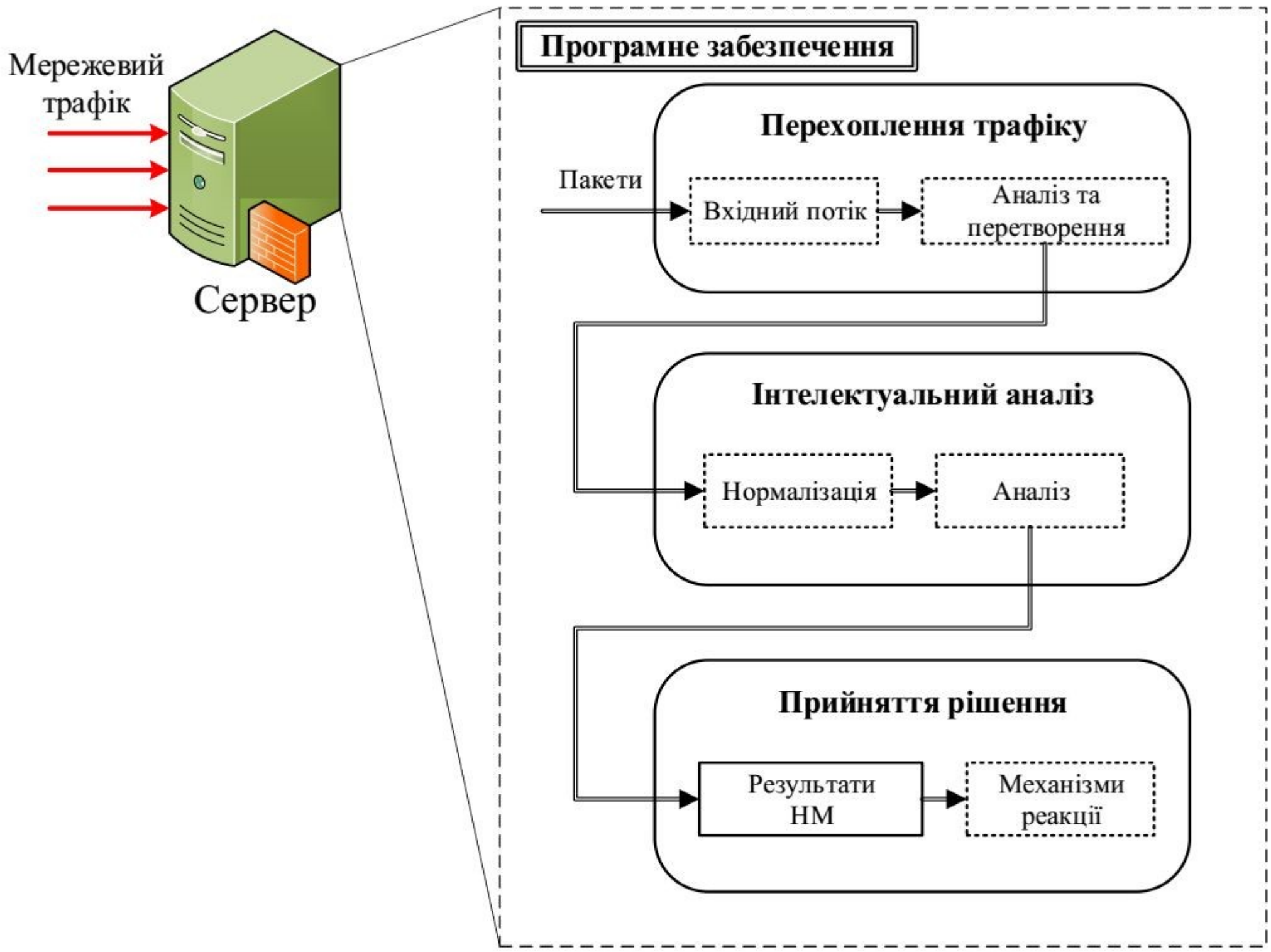
ІЛЮСТРАТИВНА ЧАСТИНА

ТИПИ DDOS-АТАК



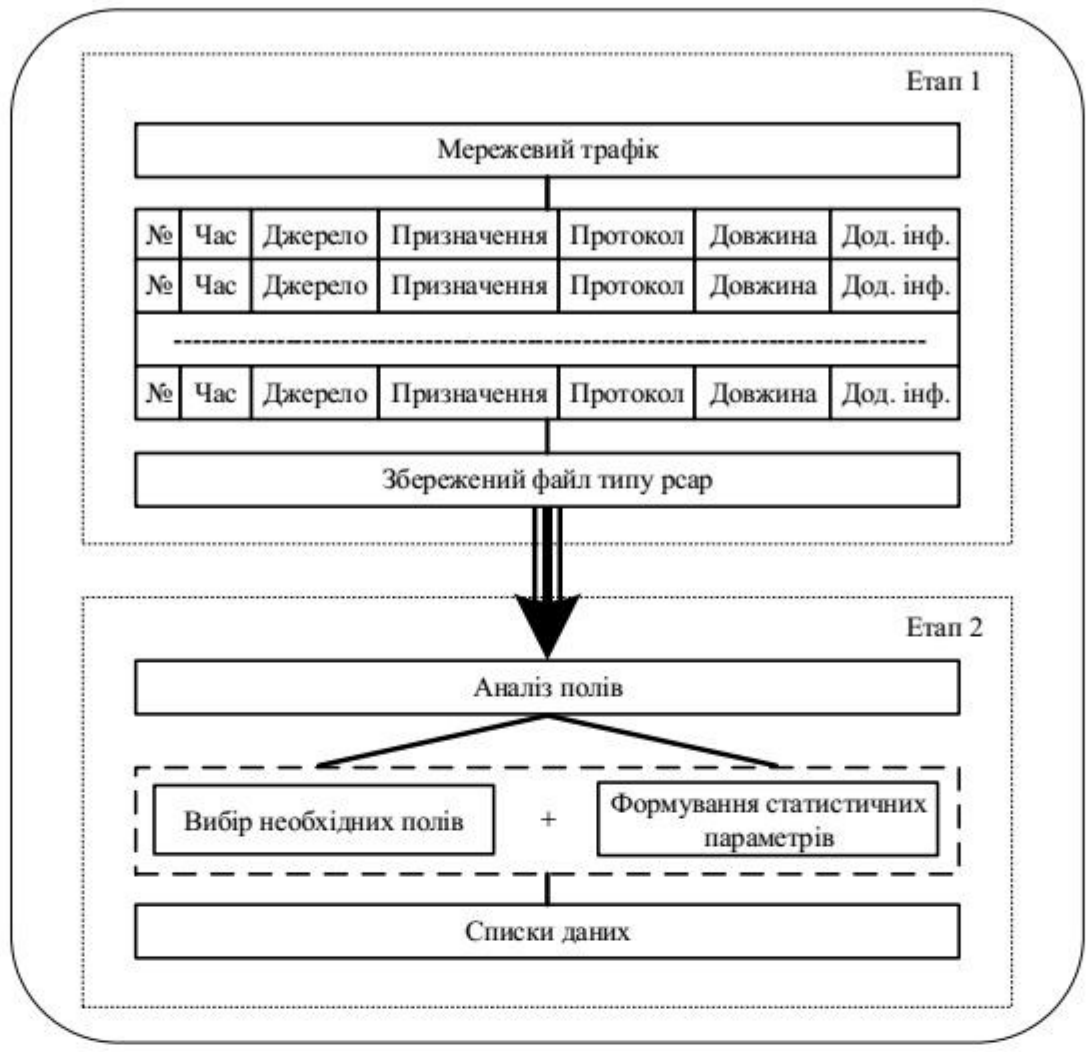
					08.20.МКР.009.00 ІЧ1			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Типи DDoS-атак	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Реценз.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

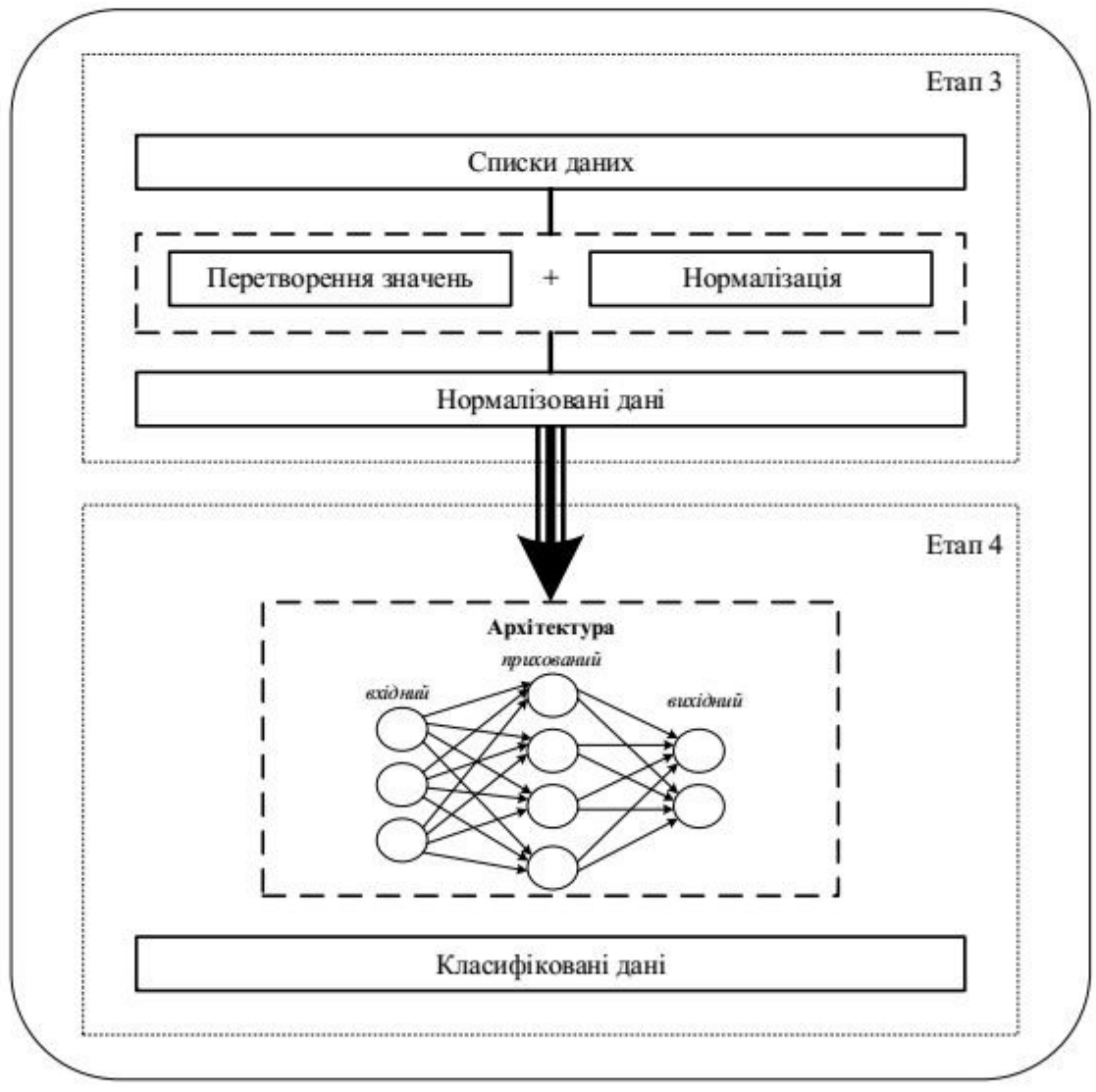
СКЛАДОВІ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ ВИЯВЛЕННЯ DDOS-АТАК

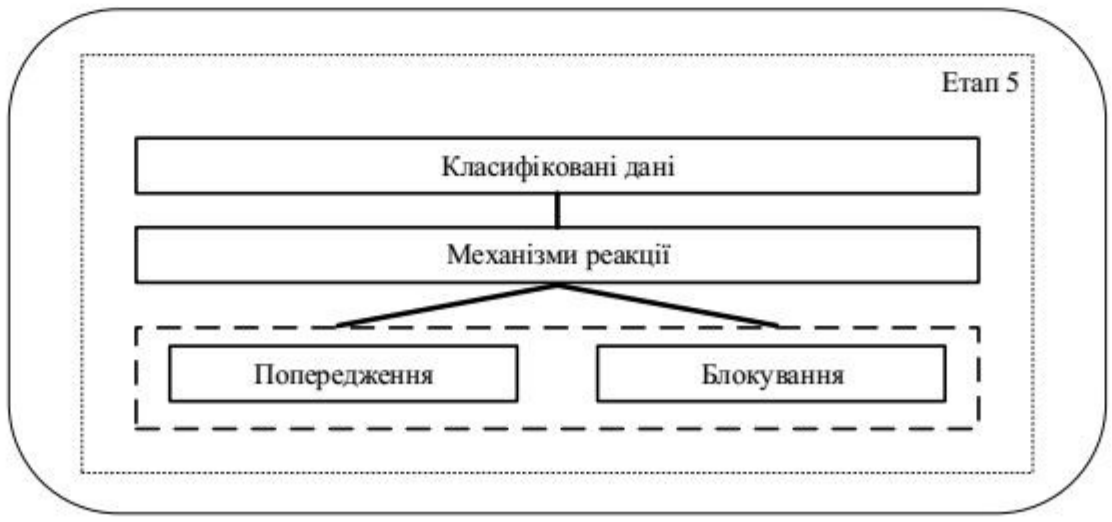


					08.20.МКР.009.00.000 ІЧ2			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Складові інформаційної технології виявлення DDoS-атак	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Реценз.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

СХЕМА ПРОЦЕСІВ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ





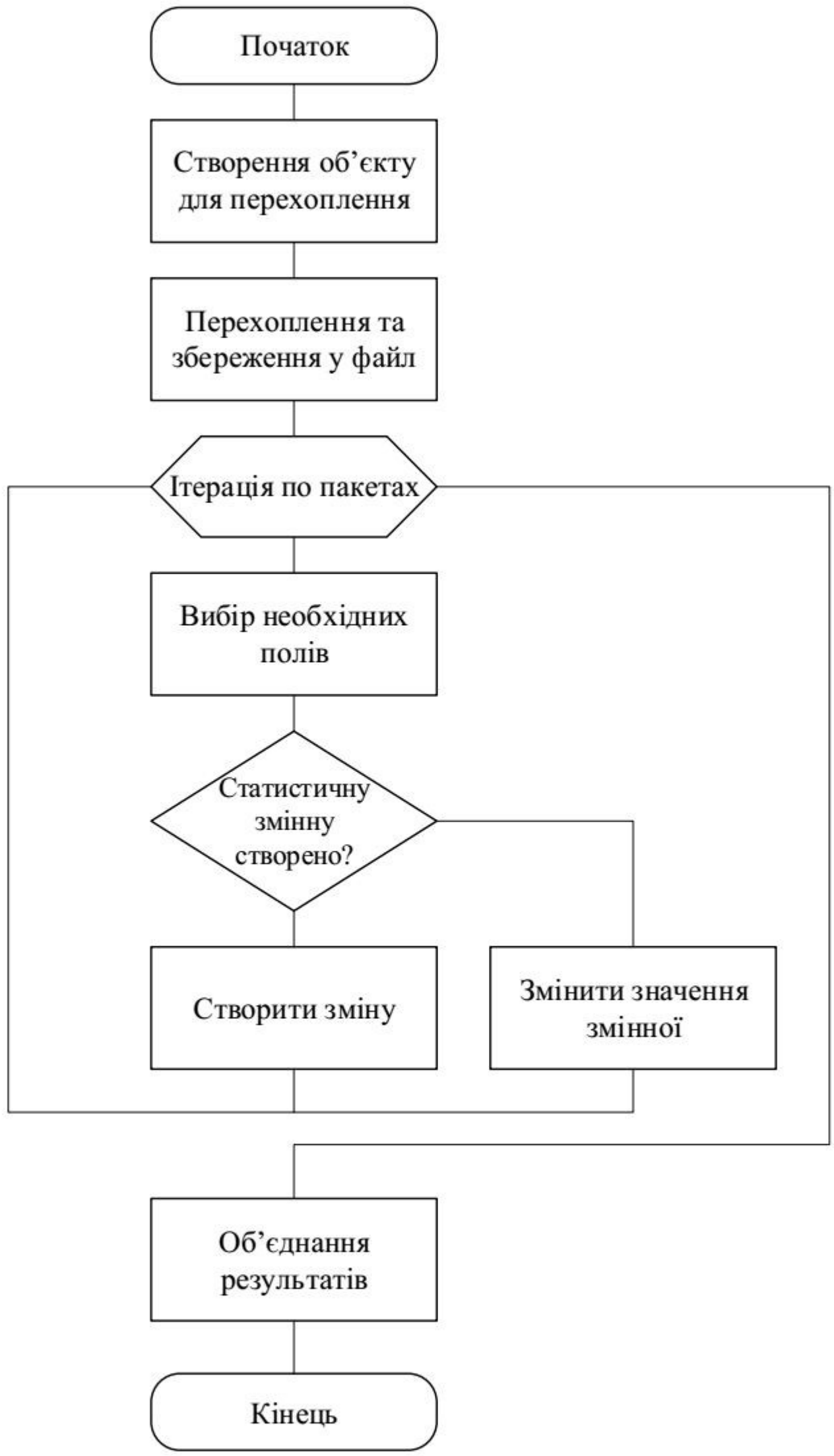


					08.20.МКР.009.00.000 ІЧЗ			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Схема процесів інформаційної технології	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Рецензн.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

СХЕМА ФОРМУВАННЯ НАБОРУ ДАНИХ

					08.20.МКР.009.00.000 І4			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Схема формування набору даних	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Рецензі.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

СХЕМА АЛГОРИТМУ ПЕРЕХОПЛЕННЯ ТА АНАЛІЗУ ПАКЕТІВ



					08.20.МКР.009.00.000 І45			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Схема алгоритму перехоплення та аналізу пакетів	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Реценз.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

МОДУЛЬ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ

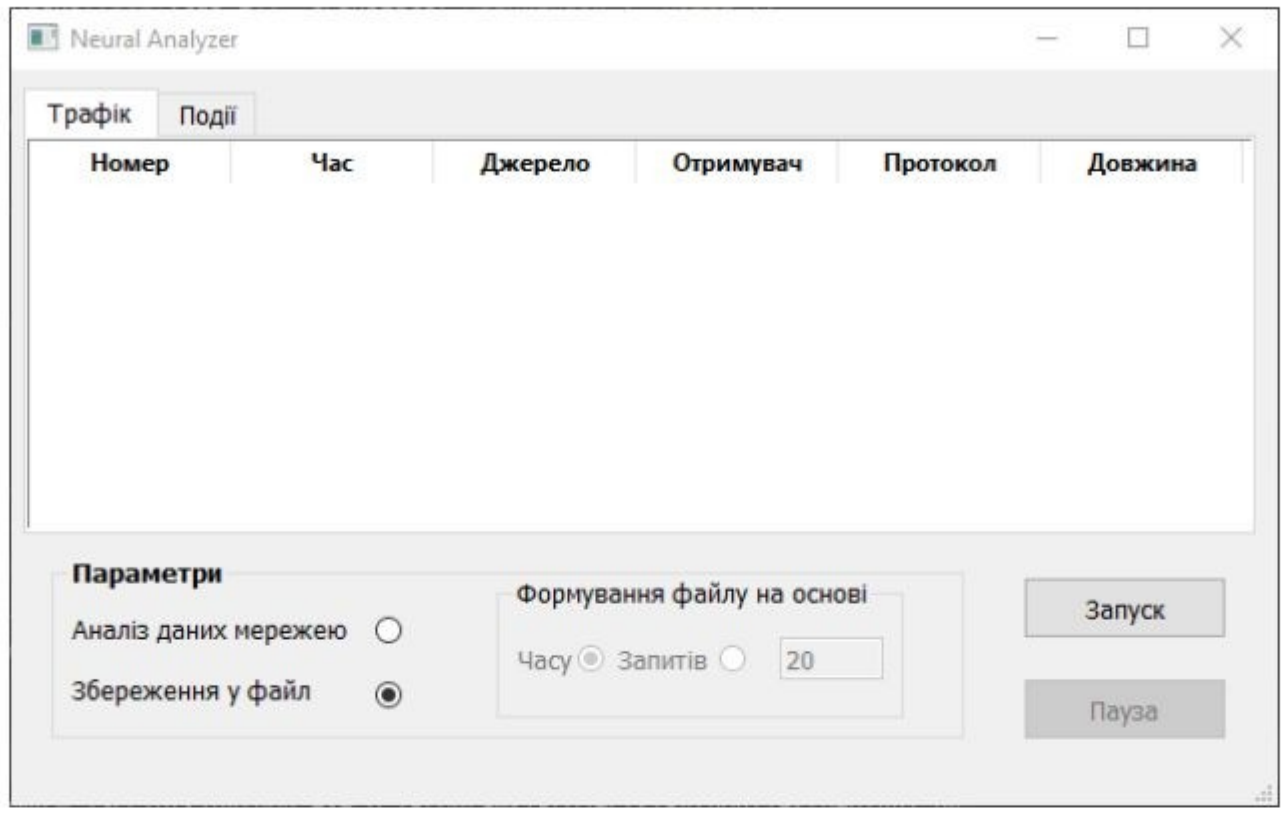
					08.20.МКР.009.00.000 ІЧ6			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Модуль інтелектуального аналізу	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Реценз.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

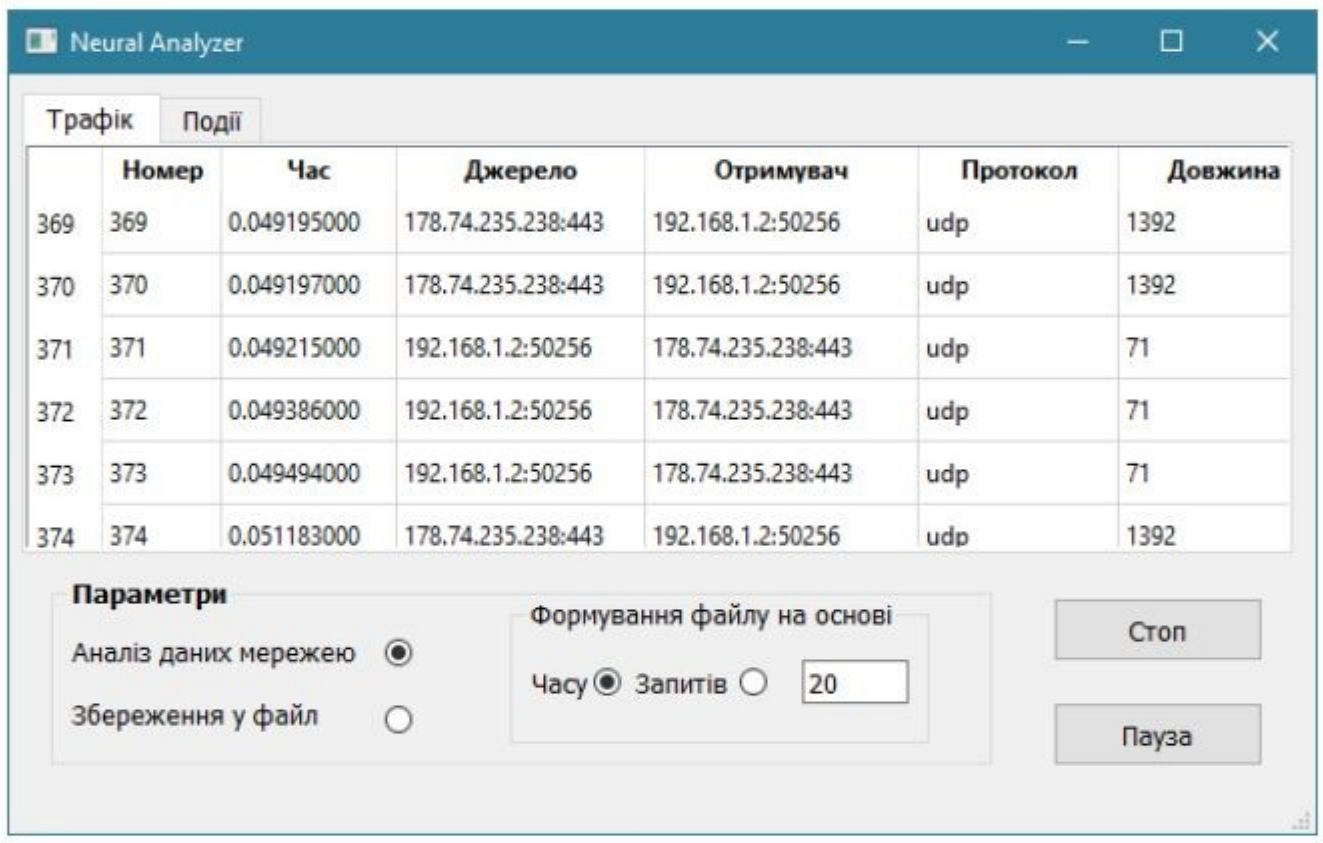
МАТРИЦЯ ПОХИБОК РЕКУРЕНТНОЇ МЕРЕЖІ LSTM

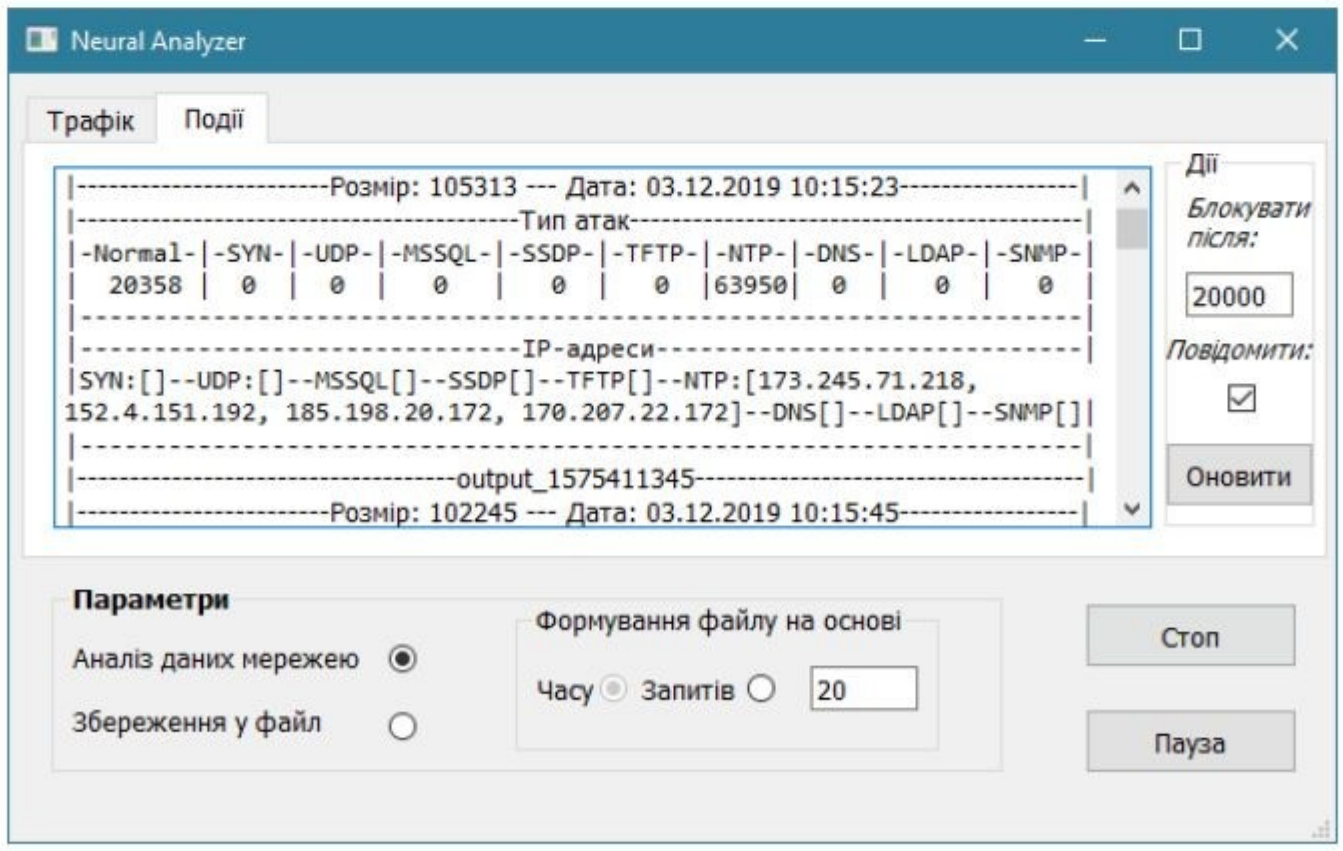
	Normal	SYN-flood	UDP-flood	MSSQL	SSDP	TFTP	NTP	DNS	LDAP	SNMP	
Normal	31767 29.9%	172 0.3%	56 0.1%	78 0.1%	46 0.0%	53 0.0%	110 0.2%	57 0.1%	44 0.0%	65 0.1%	92.7% 7.3%
SYN-flood	51 0.0%	8470 7.6%	11 0.0%	6 0.0%	29 0.0%	23 0.0%	17 0.0%	33 0.0%	71 0.1%	39 0.0%	97.2% 2.8%
UDP-flood	34 0.0%	23 0.0%	6329 5.7%	6 0.0%	9 0.0%	5 0.0%	8 0.0%	36 0.0%	63 0.1%	24 0.0%	97.6% 2.4%
MSSQL	13 0.0%	15 0.0%	3 0.0%	5294 4.6%	31 0.0%	20 0.0%	64 0.1%	7 0.0%	10 0.0%	12 0.0%	98.2% 1.8%
SSDP	18 0.0%	9 0.0%	41 0.0%	25 0.0%	9528 8.9%	52 0.0%	29 0.0%	43 0.0%	20 0.0%	16 0.0%	97.9% 2.1%
TFTP	20 0.0%	23 0.0%	11 0.0%	4 0.0%	17 0.0%	8280 7.7%	46 0.0%	63 0.1%	31 0.0%	58 0.1%	96.3% 3.7%
NTP	47 0.0%	7 0.0%	53 0.0%	26 0.0%	37 0.0%	22 0.0%	9363 8.4%	101 0.2%	15 0.0%	4 0.0%	95.9% 4.1%
DNS	32 0.0%	10 0.0%	6 0.0%	7 0.0%	56 0.1%	13 0.0%	5 0.0%	10796 9.6%	55 0.1%	173 0.3%	94.6% 5.4%
LDAP	23 0.0%	12 0.0%	12 0.0%	18 0.0%	68 0.1%	50 0.0%	3 0.0%	4 0.0%	7412 6.6%	2 0.0%	97.3% 2.7%
SNMP	131 0.2%	9 0.0%	15 0.0%	5 0.0%	20 0.0%	35 0.0%	27 0.0%	13 0.0%	36 0.0%	8655 7.8%	96.5% 3.5%
	96.4% 3.6%	96.1% 3.9%	97.8% 2.2%	97.4% 2.6%	97.6% 2.4%	98.6% 1.4%	97.2% 2.8%	96.5% 3.5%	97.1% 2.9%	94.3% 5.7%	96.8% 3.2%

					08.20.МКР.009.00.000 І4Ї			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Матриця похибок рекурентної мережі LSTM	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Рецензн.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						

ФРАГМЕНТ ІНТЕРФЕЙСУ ДОДАТКУ







					08.20.МКР.009.00.000 ІЧ8			
Змн.	Арк.	№ докум.	Підпис	Дата				
Розроб.		Кульчицький			Інтелектуальна інформаційна технологія виявлення DDoS-атак. Фрагмент інтерфейсу додатку	Літ.	Арк.	Аркушів
Перевір.		Куперштейн					1	1
Реценз.		Крупельницький				ВНТУ, 1БС-18м		
Н. Контр.		Куперштейн						
Затверд.		Лужецький В.А.						