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

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ ВИЗНАЧЕННЯ ОПЕРАЦІЙНОЇ СИСТЕМИ МЕРЕЖЕВИХ ВУЗЛІВ ІЗ ЗАСТОСУВАННЯМ МАШИННОГО НАВЧАННЯ»

Виконав: студент 2 курсу, групи БС-21м
спеціальності 125 Кібербезпека
_____ Борусевич А. В.

Керівник: к. т. н., доцент каф. ЗІ
_____ Куперштейн Л. М.
«19» грудня 2022 р.

Опонент: к. т. н. доцент каф. ПЗ
_____ Хошаба О. М.
«16» грудня 2022 р.

Допущено до захисту
Завідувач кафедри ЗІ
д. т. н., проф.
_____ Лужецький В. А.
«19» грудня 2022 р.

Вінниця ВНТУ – 2022 року

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра захисту інформації
Рівень вищої освіти II (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 125 «Кібербезпека»
Освітньо-професійна програма – «Безпека інформаційних і комунікаційних систем»

ЗАТВЕРДЖУЮ
Зав. кафедри ЗІ, д. т. н., проф.
_____ В. А. Лужецький
15 вересня 2022 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Борусевичу Артуру Вячеславовичу

1. Тема роботи: «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» керівник роботи: Куперштейн Леонід Михайлович, к. т. н., доцент, затверджені наказом ВНТУ №203 від 14.09.2022.
2. Строк подання студентом роботи – 19 грудня 2022 року
3. Вихідні дані до роботи:
   - засіб повинен запускатись в операційних системах Windows 10 та Linux;
   - засіб повинен надавати достовірність визначення типу операційної системи;
   - засіб повинен надавати перелік пов'язаних із визначеною операційною системою загроз.
4. Зміст текстової частини: Вступ. 1. Аналіз предметної області. 2. Розробка інформаційної технології. 3. Програмна реалізація інформаційної технології. 4. Економічне обґрунтування. Висновки. Список використаних джерел. Додатки.

5. Перелік графічного матеріалу.
Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання. Архітектура програмного засобу (плакат, А4). Схема алгоритму роботи програмного засобу (плакат, А4). Інтерфейс програмного засобу (плакат, А4). Структура класифікатора сімейств операційних систем (плакат, А4). Структура ансамблевої моделі (плакат, А4). Схема процесу інтелектуального аналізу (плакат, А4).

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1	Куперштейн Л. М., доц. кафедри ЗІ		
2	Куперштейн Л. М., доц. кафедри ЗІ		
3	Куперштейн Л. М., доц. кафедри ЗІ		
4	Лесько О.Й., проф. кафедри ЕВПМ		

7. Дата видачі завдання – 1 вересня 2022 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз завдання. Вступ	01.09.2022 – 04.09.2022	
2	Аналіз інформаційних джерел за напрямком магістерської кваліфікаційної роботи	05.09.2022 – 15.09.2022	
3	Науково-технічне обгрунтування	16.09.2022 – 22.09.2022	
4	Розробка технічного завдання	23.09.2022 – 04.10.2022	
5	Розробка рішень	05.10.2022 – 09.11.2022	
6	Практична реалізація, моделювання, експериментування, результати	10.11.2022 – 24.11.2022	
7	Аналіз виконання ТЗ, висновки	25.11.2022 – 29.11.2022	
8	Оформлення пояснювальної записки	30.11.2022 – 06.12.2022	
9	Попередній захист МКР та доопрацювання МКР	07.12.2022 – 19.12.2022	
10	Представлення МКР до захисту, рецензування	20.12.2022 – 21.12.2022	
11	Захист МКР	22.12.2022 – 26.12.2022	

Студент _____ Борусевич А. В.
(підпис)

Керівник роботи _____ Куперштейн Л. М.
підпис

АНОТАЦІЯ

УДК 004.855.5

Борусевич А. В. Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання. Магістерська кваліфікаційна робота зі спеціальності 125 – Кібербезпека, освітня програма – Безпека інформаційних і комунікаційних систем. Вінниця: ВНТУ, 2022. 93 с.

Укр. мовою. Бібліогр.: 31 назв; рис.: 30; табл.: 23.

Магістерська кваліфікаційна робота присвячена розробці інформаційної технології визначення операційної системи мережевих вузлів із застосуванням машинного навчання та програмного засобу, що реалізує вказану технологію. В рамках роботи проведено аналіз існуючих методів та засобів визначення операційної системи. На основі існуючих методів було запропоновано підхід до визначення операційної системи з використанням ансамблевої моделі машинного навчання. Розроблено програмний засіб, який дозволив протестувати спроектовану технологію.

Ілюстративна частина складається з 6 плакатів з демонстрацією результатів моделювання і проведених досліджень.

В економічному розділі оцінено витрати на розробку технології та програмного засобу.

Ключові слова: тестування на проникнення, операційна система, машинне навчання, збір даних.

ABSTRACT

Borusevych A. V. Information technology for determining the operating system of network nodes with the use of machine learning. Master's thesis on specialty 125 – Cybersecurity, educational program - Security of information and communication systems. Vinnytsia: VNTU, 2022. – 93 p.

In Ukrainian. Bibliographer: 31 titles; fig.: 30; tabl.: 23.

Master's thesis is devoted to the development of information technology for determining the operating system of network nodes using machine learning and a software tool that implements the specified technology. As part of the work, an analysis of existing methods and means of determining the operating system was carried out. On the basis of the existing methods, an approach to the definition of the operating system using an ensemble model of machine learning was proposed. A software tool was developed that allowed testing the designed technology.

The illustrative part consists of 6 posters with a demonstration of the results of modeling and conducted research.

The economic section estimates the costs of technology and software development.

Keywords: penetration testing, operating system, machine learning, data collection.

ЗМІСТ

ВСТУП

Розвиток нових технологій збільшує не лише їх кількість на ринку, а й кількість інцидентів інформаційної безпеки. Щодня з'являються відомості про нові вразливості бібліотек та інструментів, що використовуються як у продуктах великих корпорацій, так і у вузькоспеціалізованих програмах, які створені ентузіастами для вирішення конкретних задач. Варто зазначити, що до переліку програмного забезпечення також відносяться операційні системи, які є базою для використання створених програмних засобів.

Зазвичай розробники акумулюють зміни, щоб за допомогою встановлення одного виправлення виправити одразу декілька вразливостей. Проте чимала кількість користувачів відмовляється встановлювати останні оновлення. Особливо актуальна ця проблема серед розробників програмного забезпечення, оскільки усе більш популярною стає тактика BYOD (Bring Your Own Device), в рамках якої співробітників заохочуються користуватись власним персональним комп'ютером або ноутбуком для виконання завдань.

Описані раніше факти створюються потребу для адміністраторів мереж мати інструмент, що дозволить виявляли: неавторизовані пристрої; пристрої, на яких встановлена стара та\або вразлива версія операційної системи. Фахівцям з тестування на проникнення також важливо мати можливість аналізу складу мережі задля визначення можливих точок входу та небезпеки.

Актуальність. Тестування на проникнення – один із найбільш ефективних способів покращення рівня безпеки мережі. Авторизовані атаки на систему дозволяють виявити слабкі місця захисту мережі, а детальний звіт з результатами тестування є основою плану підвищення рівня безпеки. Один із перших і найважливіших етапів – збір даних про ціль авторизованої атаки. Необхідні дані включають у себе: архітектуру мережі, кількість вузлів, а також – операційна система мережевих вузлів. Хибне визначення операційної системи вузла може мати негативні наслідки як на ресурси, які витрачаються на проведення тестування (час та кошти), так і безпосередньо на результати тестування.

Фахівцю, який проводить тестування на проникнення, необхідно мати інструмент для визначення операційної системи вузлів у мережі, разом із переліком вразливостей, що існують у вказаній операційній системі. Адміністратору мережі підприємства потрібно мати інформацію про появу неавторизованих пристроїв у мережі. Сучасні засоби визначення операційної системи базуються на сигнатурах, а це в свою чергу створює потребу у постійному оновленні баз сигнатур, аби завжди мати актуальні дані. Розробка інформаційної технології визначення операційної системи мережевих вузлів із застосуванням методів машинного навчання дозволить збільшити достовірність визначення операційної системи і відповідно покращити процес збору даних під час проведення тестування на проникнення.

Об'єктом дослідження є процес визначення типу операційної системи мережевого вузла під час здійснення тестування на проникнення.

Предметом дослідження є методи та засоби виявлення операційної системи мережевого вузла під час здійснення тестування на проникнення.

Метою магістерської кваліфікаційної роботи є покращення процесу тестування на проникнення на етапах збору даних та сканування цільового вузла за рахунок інтелектуального визначення типу операційної системи мережевого пристрою.

Для досягнення мети необхідно виконати наступні завдання:

- розробити інформаційну технологію;
- згенерувати необхідний набір даних для навчання моделі машинного навчання;
- виконати навчання моделі машинного навчання;
- розробити програмний засіб, що реалізує інформаційну технологію;
- виконати тестування програмного засобу.

Методи дослідження. Для реалізації поставлених задач були використані методи статистичні методи для підготовки вхідної та вихідної інформації для моделювання моделі машинного навчання, методи проектування програмного забезпечення для розробки та верифікації інтелектуальної технології.

Наукова новизна. Запропонована інформаційна технологія визначення операційної системи мережевих вузлів, яка полягає у аналізі мережевого трафіку моделлю машинного навчання для підвищення точності виявлення операційних систем, що дозволяє покращити процес тестування на проникнення на етапах збору даних та сканування цільового вузла.

Практична цінність полягає у тому, розроблено програмний засіб, який дозволяє в активному і пасивному режимах здійснювати збір даних щодо сімейства та версії операційної системи мережевих вузлів під час здійснення тестування на проникнення.

За результатами науково-дослідної роботи отримано авторське свідоцтво на твір «Комп'ютерна програма «Засіб для визначення типу операційної системи віддаленого вузла» [1]. Результати магістерської роботи доповідались на таких конференціях [2-6]:

- XIV Всеукраїнська науково-практична конференція «Актуальні проблеми комп'ютерних наук АПКН-2022» відбулася 18-19 листопада 2022р;
- Всеукраїнська науково-практична інтернет-конференція «Молодь в науці: дослідження, проблеми, перспективи (МН-2022)» відбулась 16-17 червня 2022 р;
- LI Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії відбулася 31 травня 2022р;
- Міжнародний науковий симпозіум «Intelligent Solutions» відбувся 28-30 вересня 2021р;
- L Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії відбулася 11-12 березня 2021р.

Матеріали доповіді «Remote Host Operation System Type Detection Based on Machine Learning Approach» з міжнародного наукового симпозіуму «Intelligent Solutions» входять до міжнародної бази наукових робіт Scopus.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз мережевих пристроїв та їх операційних систем

Сучасні комп'ютерні мережі є доволі комплексними об'єктами, які можуть містити велику кількість об'єктів. До мережевих пристроїв відносять:

1) кінцеві пристрої користувача (персональні комп'ютери, ноутбуки);
2) периферійні пристрої (принтери, сканери);
3) мобільні пристрої (смартфони, планшети);
4) маршрутизатори;
5) інші мережеві пристрої (повторювачі, концентратори, мости, комутатори).

Усі мережі будуються на маршрутизаторах. Приклад мережі наведено на рис. 1.1.

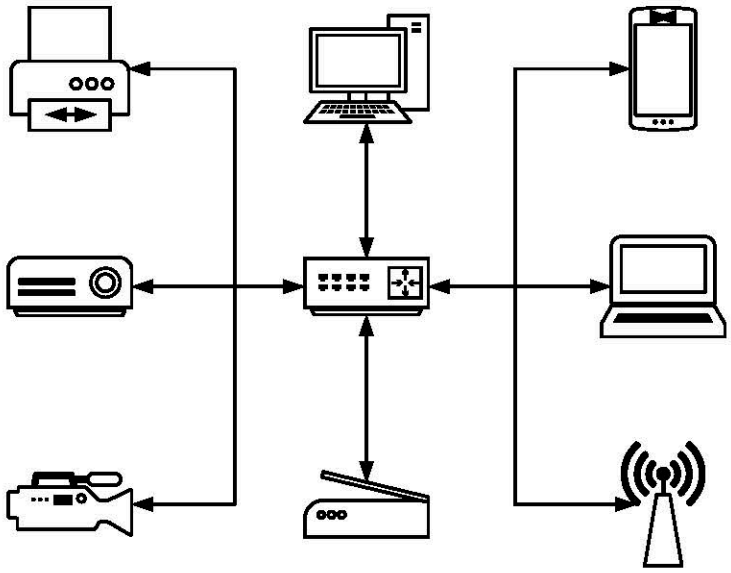


Рисунок 1.1 – Приклад архітектури локальної мережі

Маршрутизатор – це пристрій, що поєднує дві або більше мереж або підмереж з комутацією пакетів. Він виконує дві основні функції: керує трафіком між цими мережами, пересилаючи пакети даних на призначені IP-адреси, і дозволяє кільком пристроям використовувати одне Інтернет-з'єднання.

Для ефективного спрямування пакетів маршрутизатор використовує внутрішню таблицю маршрутизації – список шляхів до різних мережевих

пунктів призначення. Маршрутизатор читає заголовок пакета, щоб визначити, куди він прямує, а потім звертається до таблиці маршрутизації, щоб визначити найбільш ефективний шлях до цього пункту призначення. Потім він пересилає пакет до наступної мережі на шляху [7].

Існують наступні види маршрутизаторів:

- внутрішні маршрутизатори – пристрої для мереж невеликого розміру, мають кілька локальних портів для підключення пристроїв внутрішньої мережі та один-два порти для підключення до глобальної мережі;
- пограничні маршрутизатори – пристрої, що з'єднують одну або кілька локальних мереж та виконує шлюзову функцію для міжмережевого трафіку, мають не менше одного інтерфейсу, що належать магістральній зоні;
- магістральні маршрутизатори – пристрої, що об'єднують кілька зон пограничних маршрутизаторів, мають не менше одного інтерфейсу, що належать магістральній зоні, та велику кількість портів.

Наразі майже усі маршрутизатори мають операційну систему (з відкритим кодом або власним пропрієтарним програмним забезпеченням). Наявність операційної системи дозволяє виконувати гнучке налаштування пристрою (правил маршрутизації, фільтрації трафіку тощо) через різного роду інтерфейси. Більшість виробників використовують для налаштування локальні веб-сторінки (рис. 1.2, 1.3).

Рисунок 1.2 – Графічний інтерфейс маршрутизатора Cisco

Рисунок 1.3 – Графічний інтерфейс домашнього маршрутизатора Huawei

Наявність виконуваного коду створює потенційні місця для вразливостей (виконання шкідливого коду, отримання доступу неавторизованим користувачем тощо).

Наявність актуальних даних про операційні системи, що встановлені на мережевих пристроях (маршрутизатори, комутатори тощо) є особливо важливою не лише під час розгляду оновлення мережі в цілому, а й також в разі, коли відомі постачальники (Cisco, Juniper) оголошують про виявлені вразливості у свої продуктах. Дана інформація знадобиться при формуванні графіку технічних робіт для оновлення операційних систем (виправлення вразливостей у операційних системах шляхом встановлення оновлень, планові оновлення операційних систем в разі завершення терміну офіційної підтримки від виробника), а також при виконанні аудиту безпеки, оскільки певні вразливості можуть бути використані зловмисниками, коли на використовуваних мережевих пристроях встановлені різні версії операційних систем одного виробника [8].

Із маршрутизаторами пов’язані різного роду ризики:

– Denial of Service (DoS) – ризик відмови в обслуговуванні шляхом обмеження можливостей маршрутизатора виконувати свої функції;

- розкриття інформації – ризик отримання доступу до інформації неавторизованим користувачем;
- порушення цілісності повідомлень – ризик втручання у роботу системи;
- доступ для неавторизованих користувачів;
- підміна авторизаційних даних – отримання доступу неавторизованими користувачами;
- поширення шкідливого програмного забезпечення [9].

За даними CVE, кількість вразливостей маршрутизаторів, які були виявлені, зросли із 130 у 2019 році до 206 у 2020 році, що становить собою зріст у розмірі 58% (рис. 1.4). Причиною є глобальна епідемія, яка створила потребу у дистанційній роботі. 48% користувачів не змінювала жодних налаштувань у власному маршрутизаторі [10]. Кількість вразливостей знову зросла у 2021 році, коли було виявлено 321 вразливість маршрутизаторів, що означає зріст у 54% (зріст у 2,46 рази у порівнянні з 2019 роком).

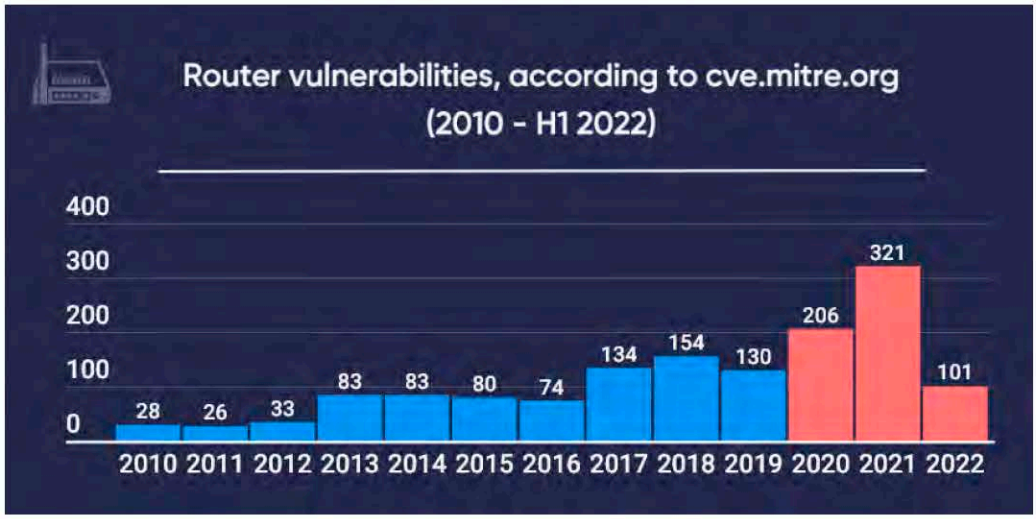


Рисунок 1.4 – Статистика кількості виявлених вразливостей маршрутизаторів (2010 – перша половина 2022)

Окрім безкоштовних операційних систем, існують також спеціалізовані операційні системи, що встановлюються на пристроях певних виробників. Найбільш популярними на ринку наразі такі виробники: Cisco, Juniper, HPE, Dell, NOKIA, AVAYA та HUAWEI [11].

1.2 Задача виначення операційної системи у кібербезпеці

Тестування на проникнення – це симульована атака інформаційної системи. Мета такої атаки – виявити слабкі місця безпеки мережі, наявність вразливостей, які можуть бути використані зловмисниками. Прикладом вразливості є відсутність обробки введеної користувачем інформації, яка може призвести до виконання атак з ін'єкцією шкідливого коду (наприклад, SQL-ін'єкція, що дозволяє отримати перелік усіх користувачів системи, або видалити усі записи у базі даних, що пов'язані із певним користувачем).

Результати тестування на проникнення можуть бути використані для тонкого налаштування фаєрволів, міжмережевих екранів, усунення вразливостей у програмному продукті тощо [12].

В загальному тестування на проникнення відбувається у 5 етапів (рис. 1.5):

1. планування та розвідка;
2. сканування;
3. отримання доступу;
4. підтримка доступу;
5. аналіз результатів.

Під час етапу планування та розвідки вирішуються такі задачі:

- визначення обсягу та цілей тестування, включаючи які системи будуть розглянуті та які методи тестування будуть використані;
- збір даних (назви та розміри мереж, доменні імена, встановлена операційна система тощо) для кращого розуміння про те, як працює ціль тестування та які потенційні вразливості можуть бути використані.

На наступному етапі виконується аналіз реакції цілі на спроби втручання. Зазвичай використовуються два методи:

- статичний аналіз – дослідження коду продукту для оцінки поведінки системи від час роботи;

– динамічний аналіз – дослідження коду продукту під час роботи, який є більш практичним, оскільки забезпечує доступ до параметру продуктивності продукту в режимі реального часу.

Рисунок 1.5 – Схема процесу тестування на проникнення

На етапі отримання доступу виконується безпосередня використання вразливостей системи або продукту для оцінки потенційної шкоди, яку може нанести використання даної вразливості. Під час цього етапу фахівці з тестування на проникнення використовують зібрані дані для формування і проведення атак [13].

Після отримання доступу необхідно оцінити чи може використання вразливостей призвести до отримання постійного доступу у скомпроментованій системі. Суть цього полягає у імітації складних тривалих загроз, коли необхідно очікувати виконання наступної атаки або зібрати необхідну кількість інформації. В кінці виконується аналіз результатів тестування на проникнення. Результатом аналізу є звіт, де зазначаються:

– вразливості, які були використані;
– дані, які було отримано;
– кількість часу, протягом якого фахівець з тестування на проникнення міг знаходитись у системі непоміченим.

Вказана інформація аналізується фахівцями з інформаційної безпеки для формування переліку дій, спрямованих на усунення вразливостей, оновлення конфігурацій, виправлення помилок в коді тощо.

Процес збору інформації є один із найважливіших під процесу виконання тестування на проникнення. Від успішності збору інформації залежить: кількість вразливостей, які можуть бути перевірені; кількість сценаріїв проведення атак, які можуть бути застосовані; кількість шляхів втручання у систему (як фізично, так і технічно).

Одним із важливих факторів, що впливає на перелік вразливостей, які можуть бути використані, є операційна система мережевого вузла. Від точності визначення залежить кількість ресурсів, які будуть витрачені на виконання атаки (час, кошти), та власне успішність даної атаки. Фахівцю необхідно мати достовірну інформацію про встановлену операційну систему, оскільки це може спричинити передчасне спрацювання системи виявлення вторгнень та власне заблокувати можливість виконання атаки, оскільки система може перейти у стан, що не є придатним для подальшого виконання тестування.

Операційні системи використовують велику кількість протоколів. Для впровадження існують стандарти, проте значення певних параметрів різниться у різних операційних системах. Надалі розглянемо особливості деяких параметрів протоколів IP, ICMP та TCP у різних операційних системах.

Протокол IP є основним протоколом зв'язку в наборі протоколів Інтернету. Його задача – передача дейтаграм між межами мереж. Протокол має завдання доставляти пакети з вихідного вузла до вузла призначення, використовуючи при цьому виключно IP-адресу, яка зазначається у заголовках пакетів.

За результатами попередніх досліджень, параметр протоколу IP, який буде використовуватись для визначення операційної системи вузла це TTL. TTL - час існування пакету (максимальна кількість переходів між роутерами). Значення параметра задається по різному у кожній операційній системі. У табл. 1.1 наведено значення TTL для різних операційних систем.

Таблиця 1.1 - Значення TTL для операційних систем

Сімейство операційної системи	Версія	TTL
Cisco	CSR1000v 16.12.3	254
Cisco	Cisco 3725 124-25.T14	254
Windows	XP	128
Windows	7	128
Windows	8.1	128
Windows	10	128
Linux	5.9.0	64
Linux	5.16.0	255
Ubuntu	20.04.1	255
MacOS	10.12.4	64
IOS	15.5	64

Протокол ICMP – протокол, який використовується мережевими пристроями для надсилання повідомлень про помилки та службових повідомлень. За результатами попередніх досліджень, параметр протоколу протоколу ICMP, які можуть використовуватись для визначення операційної системи вузла - Identifier та Sequence Number. Перший параметр використовується для ідентифікації запитів до різних джерел, другий – для ідентифікації запитів в межах 1 цільового вузла. У табл. 1.2 наведено значення поля Identifier для різних операційних систем.

Таблиця 1.2 – Значення Identifier для операційних систем

Сімейство операційної системи	Версія	Identifier
Linux	5.9.0	0x0000
Linux	5.16.0	0x0010
Cisco	CSR1000v 16.12.3	0x0200
Windows	XP	0x0200
Windows	7	0x0300
Windows	10	0x0300
MacOS	10.2.1	0x0300

Протокол TCP – основний протокол обміну даними. Забезпечує надійну та впорядковану передачу даних між застосунками у мережі. За результатами попередніх досліджень, один із параметрів протоколу TCP, які може бути

використаний для визначення операційної системи вузла – це Window size. Параметр використовується для визначення того, скільки ще байт може отримати вузол. У табл. 1.3 наведено значення даного параметра для певних операційних систем.

Таблиця 1.3 – Значення Window size для операційних систем

Сімейство операційної системи	Версія	Значення
Linux	5.9.0	5840
Linux	5.16.0	65535
Windows	XP	65535
Windows	7	8192
Windows	10	8192
MacOS	10.12.4	65535
MacOS	11.4	65535

Є ще інші параметри протоколів IP, TCP, ICMP, проте значення цих параметрів є менш репрезентативним, і такі значення варто використовувати лише в комплексі з іншими.

Наразі фахівці мають доступ до невеликої кількості програмних засобів, які дозволяють вирішити задачу визначення операційної системи мережевого вузла. Часто функціонал пропонується як одна з функцій програмного засобу, проте не є її основним призначенням. Для визначення операційної системи існуючі продукти використовують один із двох методів визначення:

– активний метод – на досліджувану систему надсилаються пакети трафіку різних протоколів, в залежності від реалізації, для того, щоб отримати відповіді, на основі яких буде сформовано висновок щодо встановленої операційної системи;

– пасивний метод – пристрій, що виконує визначення операційної системи, прослуховує мережевий трафік для того, щоб отримати необхідні значення параметрів протоколів для формування висновку щодо встановленої операційної системи.

Найбільш популярними програмними засобами, що вирішують задачу визначення операційної системи, є Nmap, NetScanTools Pro, p0f та NetworkMiner.

Nmap - безкоштовний програмний засіб, основне призначення якого – сканування мережі та портів вузлів. Додатково при скануванні можна виконати визначення операційної системи, що встановлена на вузлі. Для цього програма формує «відбиток», надсилаючи до 16 TCP, UDP та ICMP пакетів на відомі відкриті та закриті порти. Потім програма очікує та аналізує відповіді на ці пакети. В результаті формується висновок, де зазначається тип операційної системи пристрою і достовірність цього висновку. В разі якщо відсутні повні співпадіння із сигнатурою, виконується оцінка за балами (кожен параметр має відповідну вагу в балах). У програмі відсутня можливість визначення операційної системи пасивним методом, оскільки вважається, що засіб створений для швидкого активного аналізу [14]. Однак недоліком такого підходу є те, що системи захисту мають змогу розпізнати спробу сканування вузла і блокувати запити, що надходять на вузол. Користувачі також можуть надсилати нові сигнатури розробникам, якщо їм відома точна назва операційної системи, яка не була виявлена програмним засобом [15].

p0f - це програмне забезпечення, яке використовує багато складних, суто пасивних механізмів визначення операційної системи пристрою за будь-якими випадковими зв'язками TCP (часто такими, як звичайний SYN). Програма аналізує TCP SYN, TCP SYN+ACK, HTTP запити та відповіді [16]. Завдяки використанню пасивного методу, p0f надає змогу виявляти операційні системи вузлів у тих мережах, де пакети Nmap блокуються, а також робить процес визначення непомітним для системи захисту вузла, що сканується.

Прикладом іншої програми, яка використовує пасивний метод виявлення типу операційної системи, є NetworkMiner. Усе, що необхідно зробити пентестеру – запустити сканування на обраному інтерфейсі. Програма надає відповідь із певною достовірністю, використовуючи при цьому кілька баз сигнатур [17].

Описані програмні засоби використовують лише один з двох методів визначення операційної системи. А оскільки програми мають різні бази сигнатур, результати можуть суттєво відрізнятися. Також розглянуті програмні

засоби не надають користувачеві інформацію щодо потенційних вразливостей виявленої операційної системи. Ця інформація може бути використана не лише для оновлення версії операційної системи, а й слугувати основою для дій щодо плану покращення безпеки мережі.

Джерела таких вразливостей можуть бути різні. Деякі можуть бути виявлені статичними аналізаторами коду для запобігання різного роду вразливостей (передача паролів у відкритому вигляді, використання застарілих методів шифрування\гешування). Інший тип аналізаторів вказує на використання вразливих або застарілих бібліотек. Проте на відміну від програмного забезпечення, де оновлення безпеки можуть також містити корисний для користувача функціонал, операційна система рідко оновлюється користувачами. Аби виконати повноцінний аналіз результатів визначення операційних систем, що встановлені на мережевих вузлах, потрібно мати перелік вразливостей, пов'язаних із виявленими операційними системами.

Однією із найбільших та найпопулярніших баз вразливостей є Common Vulnerabilities and Exposures (CVE). Метою CVE є виявлення, визначення та каталогізація оприлюднених вразливостей у кібербезпеці [18]. База CVE забезпечує швидку кореляцію даних з різних сумісних джерел, завдяки чому можна отримати інформацію щодо виправлення вразливості у окремій базі. Дані з CVE також дозволяють здійснювати аудит сканерів вразливостей, системи виявлення вторгнень, оскільки маючи перелік вразливостей у вигляді списку CVE записів, можна додати відповідні сигнатури для виявлення у майбутньому. Кожен запис містить унікальний номер вразливості, опис вразливості, список посилань, дата створення запису.

Проте головним недоліком використання зазначеної бази є те, що вразливості описані в загальному для усіх систем, а в описі не обов'язково міститиметься назва операційної системи, де було виявлено вказану вразливість.

Для того щоб можна було отримувати актуальну інформацію про вразливості операційних систем, зокрема маючи можливість здійснювати пошук

за назвою, версією операційної системи, можна скористатись відкритою базою NVD.

NVD – це урядова база США, де зберігаються стандартизовані дані про вразливості, які представлені за допомогою протоколу автоматизації вмісту безпеки (SCAP). У записах вразливостей містяться: посилання на списки перевірки безпеки системи, перелік пов'язаних вразливостей програмного забезпечення, помилки конфігурації, назви продуктів та показники впливу. Після появи запису у базі CVE, запис у NVD з'являється протягом години. Після цього аналітики NVD розпочинають роботу для опису загрози. В разі відсутності інформації, аналітики будуть розглядати найгірший варіант задля уникнення ситуації, коли ймовірна шкода від реалізації вразливості буде недооцінена [19].

Вказані переваги дозволять вчасно отримати інформації про вразливості виявлених операційних систем. Тому при розробці технології варто використовувати бібліотеку для отримання актуальної інформації щодо вразливостей.

1.3 Формалізація вимог та постановка задачі

Сучасні програмні засоби, що вирішають задачу визначення операційної системи мережевих вузлів використовують сигнатурний метод. Недоліком цього підходу є низька достовірність визначення в разі відсутності запису в базі сигнатур. Для підвищення достовірності виявлення можна використовувати методи машинного навчання. Існуючі значення параметрів протоколів не дають можливости створити однозначні правила визначення операційних систем, і також такі правила будуть мати великий обсяг, а в разі додавання нової операційної системи до бази сигнатур – потрібно повторно витрачати час на створення правила. Тому варто застосувати методи машинного навчання. Ансамблевий підхід дозволить масштабувати модель під час нових досліджень, оскільки кожна модель машинного навчання буде навчена для вирішення

конкретної задачі (визначення операційної системи в межах певного сімейства). Такий підхід дозволяє зменшити витрати і збільшити достовірність визначення.

У даній роботі об'єктом виступає процес визначення типу операційної системи мережевого вузла під час здійснення тестування на проникнення.

Головною метою є покращення процесу тестування на проникнення на етапах збору даних та сканування цільового вузла за рахунок інтелектуального визначення типу операційної системи мережевого пристрою.

Вимоги до проектування технології:

- технологія повинна вирішувати задачу виявлення операційної системи визначеного віддаленого вузла;
- архітектура програмного забезпечення, що реалізує технологію, повинна включати в себе: модуль керування, модуль взаємодії з мережею, модуль попередньої обробки та модуль інтелектуального аналізу даних;
- модуль керування повинен надавати користувачеві елементи керування, інформаційні повідомлення та результати роботи програмного засобу;
- модуль взаємодії з мережею повинен надсилати пакети трафіку на мережеві вузли, операційну систему яких потрібно визначити, а також отримувати пакети, що передаються мережею, для подальшої обробки;
- модуль обробки повинен отримувати пакети трафіку для отримання параметрів протоколів та формування сигнатури мережевого вузла для подальшої роботи;
- модуль інтелектуального аналізу повинен виконувати визначення операційної системи вказаного користувачем вузла та надавати значення операційної системи, достовірність визначення, перелік пов'язаних вразливостей;
- для навчання моделей машинного навчання, що будуть використовуватись модулем інтелектуального аналізу, потрібно надати набір даних.

На основі проаналізованих даних, визначено наступні задачі:

- спроектувати архітектуру інформаційної технології;
- розробити алгоритми роботи технології;

- розробити модуль обробки даних;
- розробити модуль інтелектуального аналізу даних;
- розробити архітектуру ансамблевої моделі машинного навчання;
- обрати мову програмування та інструменти для розробки;
- виконати навчання моделей машинного навчання;
- реалізувати графічний інтерфейс програмного засобу;
- розробити програмний засіб, що реалізує інформаційну технологію;
- виконати тестування програмного засобу.

2 РОЗРОБКА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

2.1 Структура інформаційної технології

Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання складається з декількох процесів, кожен з яких має власне функціональне призначення. Схема процесів інформаційної технології наведена на рис. 2.1.

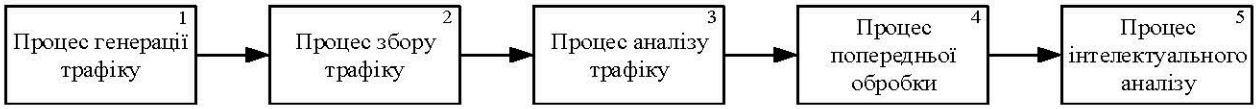


Рисунок 2.1 – Схема процесів

Розглянемо окремо кожний процес.

1) Процес генерації трафіку. Процес включає в себе такі етапи (рис. 2.2):

- формування ICMP пакетів (з полем Type 8 Echo Request) та TCP пакетів;
- надсилання сформованих пакетів на вказані користувачем мережеві адреси вузлів.

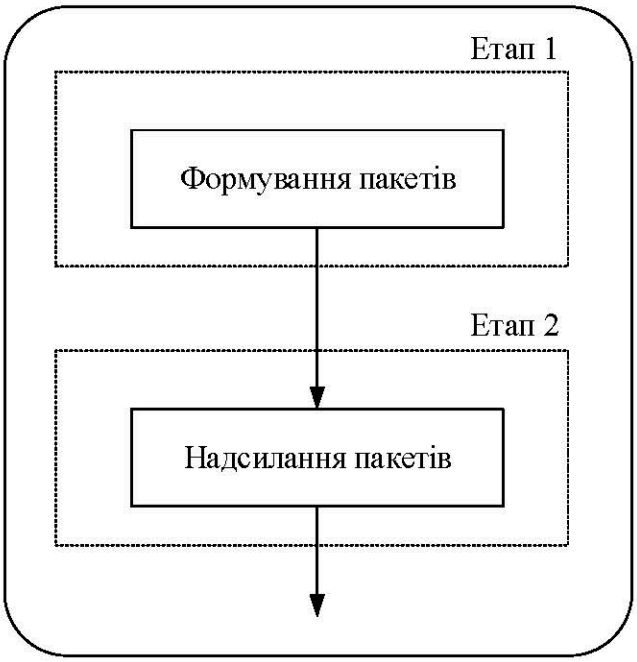


Рисунок 2.2 – Схема процесу генерації трафіку

Вказаний процес є обов'язковим при визначенні операційної системи з використанням активного методу визначення. Модель процесу можна описати як:

$$I_1 = \langle P_{1i}, P_{2i} \rangle_{i=1\ldots n},$$

де P_1, P_2 – надіслані відповідні до дій пакети, n – кількість вузлів, для яких необхідно визначити встановлену операційну систему.

В разі використання пасивного методу визначення, процес визначення операційної системи розпочинається із процесу збору трафіку.

2) Процес збору трафіку. Процес включає в себе збір трафіку у мережі (надіслані вузлом пакети та відповіді). Схему процесу наведено на рис. 2.3.

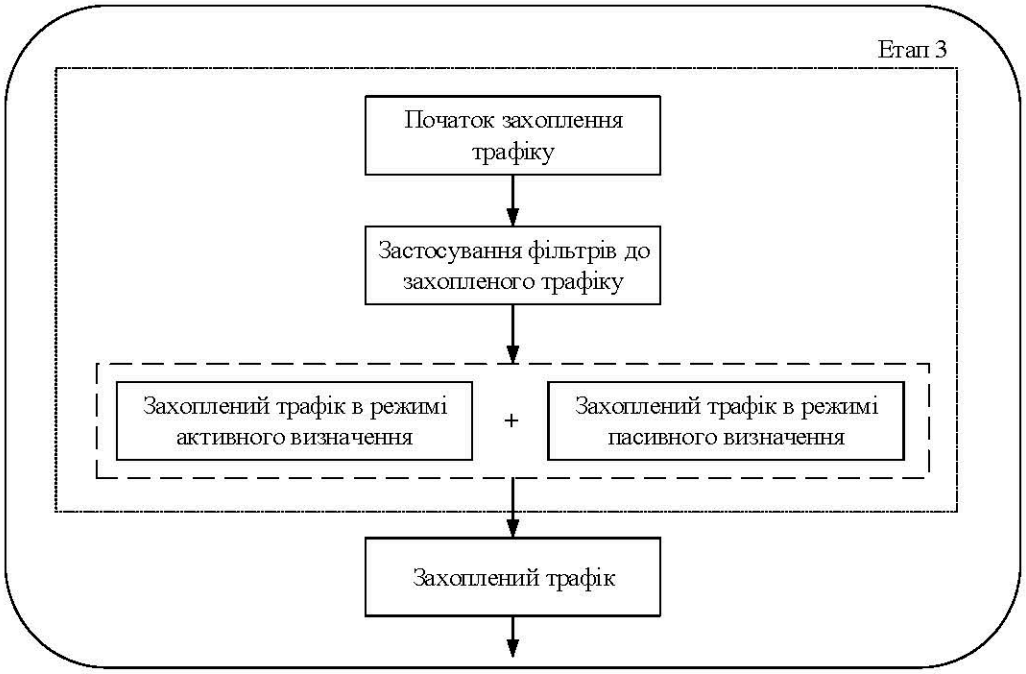


Рисунок 2.3 – Схема процесу збору трафіку

Модель процесу можна описати як:

$$I_2 = R_1^{(I_1)} \cup R_2,$$

де $R_1^{(I_1)}$ – відповіді на згенеровані у попередньому процесі пакети трафіку операційних систем під час виконання визначення активним методом, R_2 – отримані пакети від вузла під час виконання визначення пасивним методом.

3) Процес аналізу трафіку. Процес включає в себе відбір пакетів, де мережева адреса пристрою співпадає з мережевою адресою вузла, який аналізується, та отримання значень параметрів з відібраних пакетів. Схему процесу наведено на рис. 2.4. Результатом виконання процесу є значення наступних параметрів:

1. IP – ttl.
2. ICMP – ident, seq_le, data_len.
3. TCP – window_size_value, window_size_scalefactor.

Отримані значення параметрів з пакетів перетворюються у сигнатури. Сигнатури мають наступну структуру:

$$F = addr,\ ip,\ icmp,\ tcp,$$

де *addr* – мережева адреса вузла, *ip, icmp, tcp* – значення заголовків вказаних протоколів.

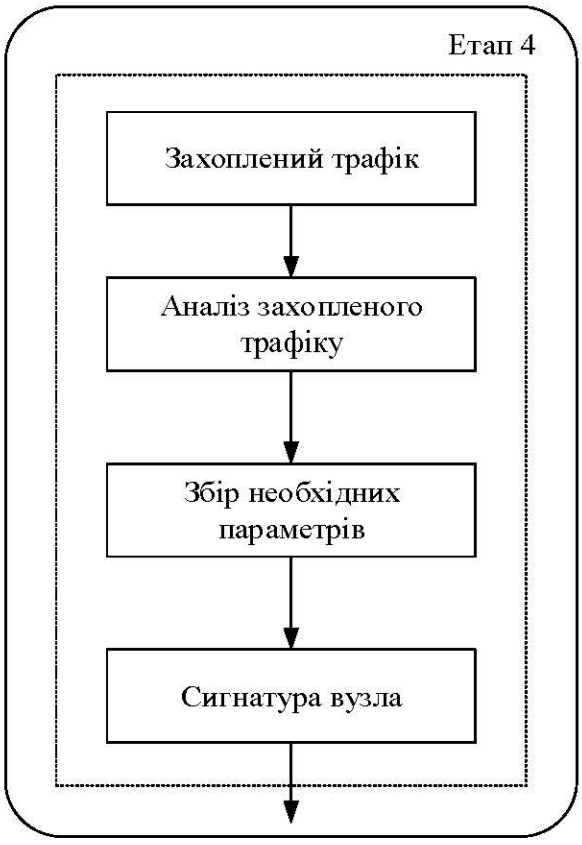


Рисунок 2.4 – Схема процесу аналізу трафіку

Модель процесу можна описати як:

$$I_3 = \langle \langle P_{IP}^{I_2}, P_{ICMP}^{I_2}, P_{TCP}^{I_2} \rangle, F \rangle,$$

де $\langle P_{IP}^{I_2}, P_{ICMP}^{I_2}, P_{TCP}^{I_2} \rangle$ – множини значень заголовків IP, ICMP та TCP протоколів, F – сформована сигнатура вузла.

4) Процес попередньої обробки. Процес включає в себе нормалізацію отриманих параметрів протоколів для підготовки цих даних до визначення моделлю машинного навчання. Схему процесу наведено на рис. 2.5.

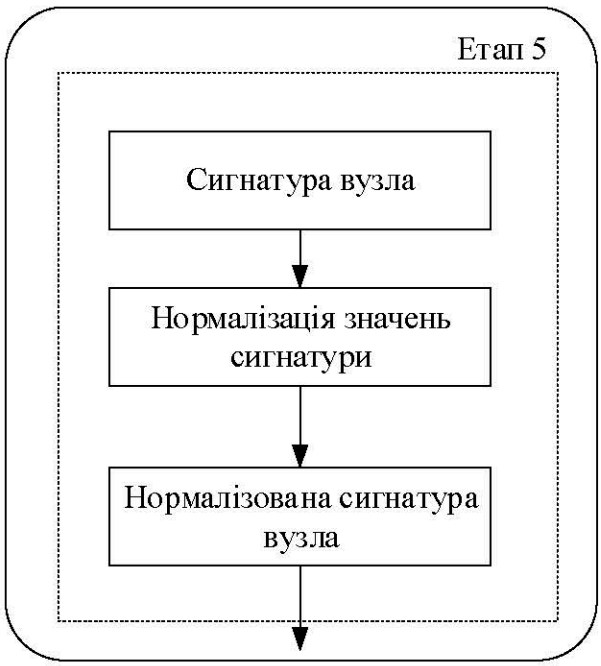


Рисунок 2.5 – Схема процесу попередньої обробки

Модель процесу можна описати як:

$$I_4 = \langle N, F^{(I_3)}, F_N \rangle,$$

де N – функція нормалізації, $F^{(I_3)}$ – сформована сигнатура вузла з попереднього процесу, F_N – нормалізована сигнатура вузла.

5) Процес інтелектуального аналізу. Процес відбувається у 2 етапи:

1. класифікація операційної системи на основі нормалізованої сигнатури;
2. пошук вразливостей у базі вразливостей NVD.

Під час першого етапу інтелектуального аналізу моделі машинного навчання буде надано сигнатуру операційної системи. В результаті інтелектуального аналізу модель надає відповідь у вигляді назви операційної системи, що відповідає сигнатурі із зазначенням точності цього рішення.

Під час другого етапу інтелектуального аналізу результати класифікації використовуються для пошуку вразливостей операційних систем у міжнародні базі вразливостей NVD. В результаті буде отримано перелік вразливостей, які пов'язані із виявленої операційною системою.

Загальну схему процесу наведено на рис. 2.6. Модель процесу можна описати так:

$$I_5 = \langle S^{(I_4)}, L, P(S^{(I_4)}, L)\rangle,$$

де $S^{(I_4)}$ – сигнатура операційної системи, яку було отримано з попереднього процесу, L – перелік операційних систем, що може визначати модель, $P(S^{(I_4)}, L)$– рішення моделі машинного навчання, яка включає в себе ймовірність віднесення сигнатури $S^{(I_4)}$ до значення операційної системи L.

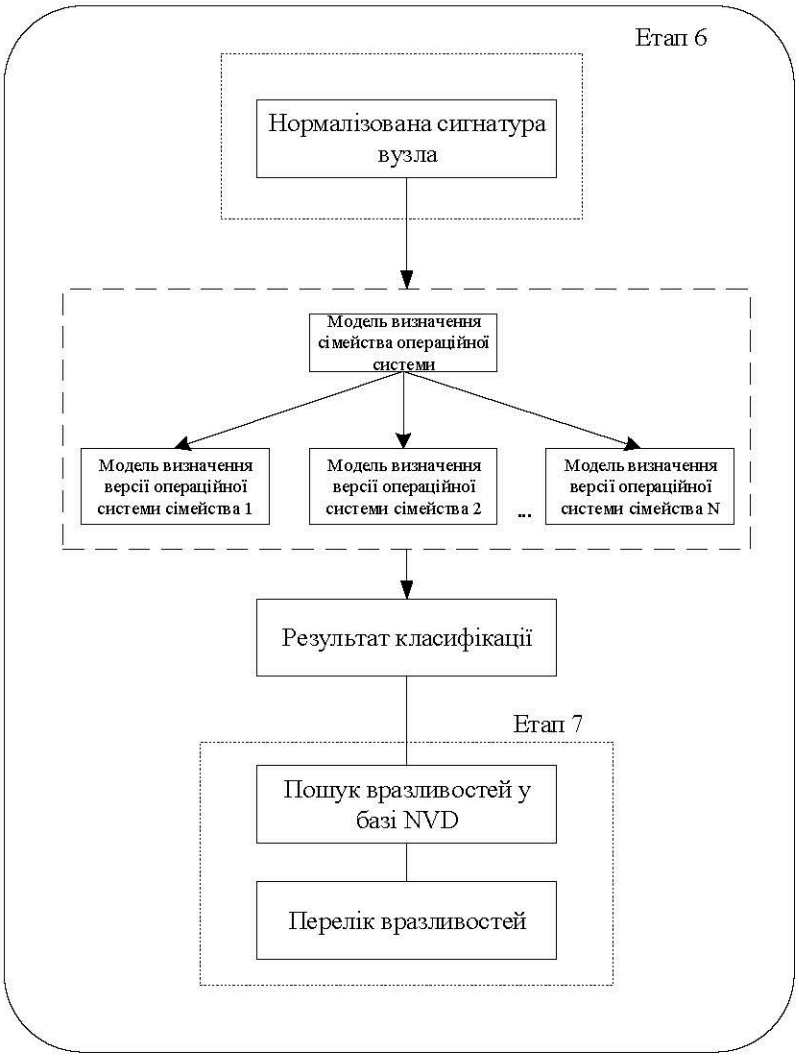


Рисунок 2.6 – Схема процесу інтелектуального аналізу

В результаті створено моделі та схеми процесів інтелектуальної технології визначення операційної системи мережевих вузлів із застосуванням машинного навчання. Наступним кроком є розробка архітектури системи для реалізації інтелектуальної технології.

2.2 Розробка архітектури програмного засобу

Інформаційна технологія визначення операційної системи віддаленого вузла включає в себе комплекс методів і задач. Архітектура програмного засобу, який реалізовує дану технологію, наведена на рис. 2.7.

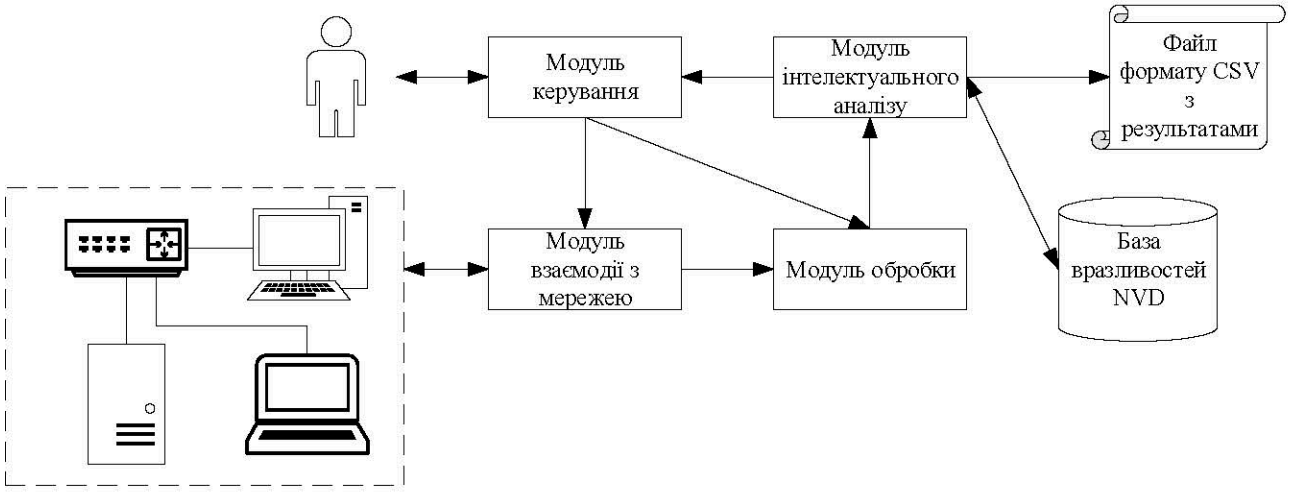


Рисунок 2.7 – Архітектура програмного засобу, який реалізовує інформаційну технологію

Розглянемо кожен модуль системи окремо.

1) Модуль керування. Даний модуль починає роботу одразу після запуску програмного засобу, і надає користувачеві графічний інтерфейс для взаємодії з програмним засобом через поля вводу та інших елементів керування. Модуль також виконує функції відображення діалогових вікон та результатів роботи програми.

2) Модуль взаємодії з мережею виконує наступні функції:

– прослуховування трафіку в мережі;

- відправлення мережевих пакетів трафіку на цільовий мережевий вузол (в разі визначення операційної системи мережевого вузла в активному режимі);
- фільтрація трафіку та отримання трафіку, що йду лише від вузла, операційна система якого визначається.

3) Модуль обробки. Даний модуль виконує наступні операції:

- отримання полів заголовків протоколів, які необхідні для формування сигнатури операційної системи;
- нормалізація значень параметрів протоколів;
- формування сигнатури операційної системи мережевого вузла.

Сигнатура представляє собою набір значень: мережева адреса вузла, значення обраних параметрів протоколів IP, ICMP, TCP. Усі значення розділені між собою комою, а мережева адреса вузла не бере участі під час класифікації операційної системи, а лише зазначається у результуючому звіті програми.

4) Модуль інтелектуального аналізу вирішує задачу класифікації операційної системи, отримуючи на вході сигнатуру вузла від модуля обробки, та класифікує операційну систему, використовуючи навчені моделі машинного навчання. Для вибору методу машинного навчання, який буде використано для класифікації операційних систем, необхідно провести моделювання різних методів машинного навчання, виконати аналіз результатів та відповідно до цього сформувати висновок.

Модуль інтелектуального аналізу використовує навчену модель машинного навчання для вирішення задачі класифікації операційної системи мережевого вузла . Для вирішення задач існує декілька методів. Для того, щоб обрати оптимальний метод, варто виконати моделювання різних методів, провести аналіз результатів і на основі висновків обрати метод класифікації, який буде використовуватись у модулі інтелектуального аналізу.

2.3 Збір та підготовка даних для моделювання

Для роботи модуля інтелектуального аналізу для визначення операційної системи віддаленого вузла необхідно мати базу сигнатур для навчання у вигляді набору підготовлених даних. Можна використовувати готовий набір даних, проте даний набір містить лише типи операційних систем, проте користувача може цікавити підтип та версія операційної системи [20]. Для цього потрібно самостійно формувати набір даних. В даному випадку даними виступають пакети та заголовки різних мережевих протоколів.

Для захоплення пакетів із мережі використовувався програмний засіб Wireshark, який дозволяє записувати сеанс захоплення у файл для подальшої обробки без необхідності бути приєднаним до мережі [21].

Для створення необхідного трафіку для кожної операційної системи виконувались наступні дії:

- виконання команди «ping» з досліджуваної операційної системи;
- надсилання TCP SYN та TCP ACK пакетів на досліджувану операційну систему.

Оскільки необхідно створити набір даних з параметрами протоколів пакетів маршрутизаторів, варто використати середовище віртуалізації мережевих пристроїв. Одним із таких середовищ є GNS3 [22]. GNS3 пропонує простий спосіб проектувати та будувати мережі будь-якого розміру без потреби в апаратному забезпеченні. Середовище надає інструменти для симуляції мережі в режимі реального часу, запуск операційних систем, що емулюють реальну поведінку мережевого обладнання, а також дозволяє поєднувати реальні пристрої з емульованими в межах мережі.

Під час створення набору даних було досліджено наступні операційні системи: Linux (версія 5.9.0, 5.16.0, 20.04.1), MacOS (версія 10.12.4 та 11.4), IOS (версія 15.5), Windows (10 Home 20h2, 10 Pro 20h2, 8.1, 7 Professional, XP Professional SP3), Cisco (CSR1000v 16.12.3, Cisco IOU L2 15.1g, Cisco 3725 124-

25.T14), Juniper (vSRX 20.4R1, vMX vCP 20.2R1.10-KVM, Space 17.2R1.4), HP (HPE VSR1001 7.10.E0519L0), Dell (Dell OS9 9.8.0) та Huawei (WS5200 10.0.2.19) на 20 різних вузлах, формуючи таким чином 20 конкретних класів (версія операційної системи сімейства) та 6 загальних класів (Windows, MacOS, Unix, Routers, Cisco, Juniper).

Далі, за допомогою бібліотеки PyShark для мови програмування Python, було захоплено трафік від кожного вузла, а саме відповіді мережевого вузла. Потім було виконано аналіз файлів захопленого трафіку, а саме - пошук пакетів необхідних протоколів (IP, ICMP, TCP). За результатами попередніх досліджень, обрано наступні значення полів:

1) IP – ttl

2) ICMP – ident, seq_le, data_len

3) TCP – window_size_value, window_size, window_size_scalefactor

Далі отримані параметри було перетворено на сигнатури операційних систем. Сигнатури мають наступну структуру:

class, ip, icmp, tcp

де *class* – точна назва і версія операційної системи, *ip, icmp, tcp* – значення заголовків вказаних протоколів.

Опис полів протоколів та приклад їх значень наведено у табл. 2.1.

Таблиця 2.1 – Опис параметрів протоколів

Протокол	Назва параметру	Опис	Приклад значення
IP	ttl	час «життя» пакету	64
ICMP	ident	ідентифікація частин пакета протоколу	1
ICMP	seq_le	поточна довжина послідовності	142848
ICMP	data_len	довжина повідомлення в пакеті	32
TCP	window_size_value	значення розміру «вікна»	513
TCP	window_size	повне обраховане значення вікна	131328
TCP	window_size_scalefactor	обрахований модифікатор розміра вікна (2^n)	256

Для навчання моделей машинного навчання, які будуть визначати сімейство операційних систем, застосовувався зібраний набір даних, де замість назви операційної системи зазначалося сімейство операційної системи.

Отриманий перелік сигнатур зберігається у форматі «csv» (Comma separated values), тобто кожне значення відокремлене від іншого роздільником (в даному випадку – комою).

Було сформовано 7 наборів даних: повний загальний набір (у ньому містяться усі сигнатури операційних систем, проте замість точної версії операційної системи зазначалося сімейство) та 6 наборів даних сигнатур (Windows, MacOS, Unix, Generic routers (HP, Dell, Huawei), Cisco, Juniper), які містять сигнатури операційних систем із зазначенням версії. Кількість записів у наборах даних наведено у табл. 2.2.

Таблиця 2.2 – Кількість записів у наборах даних

Назва ОС	Кількість записів		Сімейство ОС	Кількість записів	
Windows 10 Home 20h2	4717	23.36%	Windows	20197	29.64%
Windows 10 Pro 20h2	3413	16.9%			
Windows 7 Professional	4385	21.7%			
Windows XP Professional SP3	5281	26.15%			
Windows 8.1	2401	11.89%			
Cisco CSR1000v 16.12.3	2499	27.7%	Cisco	9023	13.24%
Cisco IOU L2 15.1g	3931	43.56%			
Cisco 3725 124-25.T14	2593	28.74%			
Linux 5.9.0	1855	26.14%	Unix	7096	10.42%
Linux 5.16.0	1629	22.96%			
Ubuntu 20.04.1	3612	50.9%			
Mac OS 10.12.4	3997	32.09%	MacOS	12454	18.28%
MacOS 11.4	4350	34.93%			
IOS 15.5	4107	32.98%			
Juniper vSRX 20.4R1	4035	34.95%	Juniper	11546	16.95%
Juniper vMX vCP 20.2R1.10-KVM	3377	29.25%			
Junos Space 17.2R1.4	4134	35.8%			
HPE VSR1001 7.10.E0519L03	2310	29.56%	Generic routers	7815	11.47%
Dell OS9 9.8.0	2276	29.12%			
Huawei WS5200 10.0.2.19	3229	41.32%			

Повна загальна вибірка буде використовуватись для навчання моделі, яка буде визначати сімейство операційної системи. Спеціалізовані набори даних будуть використовуватись для навчання спеціалізованих моделей.

2.4 Моделювання та вибір класифікатора для вирішення задачі

Існують різні задачі, які можна вирішувати, використовуючи машинне навчання. До них відносять:

- задача регресії – прогнозування, яке базується на об'єктах з різними ознаками;
- задача класифікації – вирішення, до якого класу належить певний об'єкт;
- задача кластеризації – групування об'єктів зі схожими характеристиками;
- задача виявлення аномалій – виявлення значень або подій, які не характерні для системи.

В даному випадку потрібно вирішити задачу класифікації, а саме – багатокласова класифікація (потрібно віднести систему до одного з класів, аналізуючи ознаки системи).

Для вирішення задачі класифікації існує багато методів. Для аналізу було обрано наступні [23]:

- дерево рішень (Decision Tree) – встановлення мітки класу, використовуючи набір правил у вигляді дерева;
- багатошаровий персептрон (Multilayer Perceptron) – нейромережа, що має один або декілька прихованих шарів, мета якої – встановлення ваг зв'язків і створення таким чином алгоритму вирішення задачі;
- гауссовський наївний баєсовський класифікатор (Gaussian Naïve Bayes) – вирішення задачі класифікації, базуючись на незалежності кожної пари ознак (дані для кожного представника обрані з простого розподілу Гаусса);
- метод k найближчих сусідів (K-Nearest Neighbors) – встановлення мітки класу, використовуючи схожість ознак» найближчих k сусідів;

- логістична регресія (Logistic Regression) – ймовірності описують можливі результати, використовуючи логістичну функцію;
- випадковий ліс (Random Forest) – ансамблевий класифікатор, до якого входять кілька дерев рішень, що дозволяє підвищити достовірність визначення (кількість правильних випадків класифікації).

Для моделювання також необхідно визначити найкращий набір гіперпараметрів для кожного з класифікаторів. Гіперпараметри – параметри моделі, які встановлюються до етапу навчання моделі. Одним із методів вирішення даної задачі є алгоритм GridSearch. Суть алгоритму – перебрати перелік гіперпараметрів моделі, виконати аналіз оцінки і отримати перелік параметрів, при яких модель має найкращі результати.

Після виконання алгоритму було отримано наступний перелік гіперпараметрів для кожного з класифікаторів:

- DecisionTree(criterion='entropy', max_depth=7);
- MLP(alpha=0.06, hidden_layer_sizes=(25, 30, 20));
- KNearestNeighbors(leaf_size=6, n_neighbors=3);
- GaussianNB(var_smoothing=1e-8);
- LogisticRegression(C=7, max_iter=750, solver='newton-cg');
- RandomForest(criterion='entropy', max_features='sqrt', min_samples_leaf=3, n_estimators=1850).

Вказані гіперпараметри будуть використовуватись при моделювання класифікаторів.

Наступним кроком є власне моделювання (навчання та перевірка результатів) усіх класифікаторів. Для навчання було використано проміжну вибірку даних, де міститься по тисячі сигнатур кожної операційної системи. Для навчання було обрано 70% даних, для тестування – 30%. Результати моделювання наведено у табл. 2.3. Для оцінки класифікатора було використано:

- accuracy – доля правильно визначених класів;
- f1 – середньозважена оцінка похибок першого та другого роду;

- precision – відношення кількості правильно визначених класів до суми кількості правильно визначених класів та помилок першого роду (false positive);
- recall – відношення кількості правильно визначених класів до суми кількості правильно визначених класів та помилок другого роду (false negative);
- confusion_matrix – матриця, що відображає кількість правильних визначень та кількість помилкових визначень;
- fp – кількість помилок першого роду (false positive);
- fn – кількість помилок другого роду (false negative).

Таблиця 2.3 – Результати моделювання класифікаторів

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	**0,98135**	**0,98143**	**0,98143**	**0,98098**	**161**	**220**
Logistic regression	0,81999	0,81875	0,8201	0,81789	1452	2227
Multilayer Perceptron	0,75329	0,7542	0,7542	0,8624	3076	1966
K-Nearest Neighbors	0,97445	0,9756	0,9782	0,9728	185	337
GaussianNB	0,78963	0,78842	0,78435	0,7896	2037	2262
Random Forest	0,97054	0,97123	0,97123	0,9732	425	177

Далі наведено графік важливості ознак (рис. 2.8), матриця помилок дерева рішень (рис 2.9) та фрагмент структури дерева рішень (рис. 2.10).

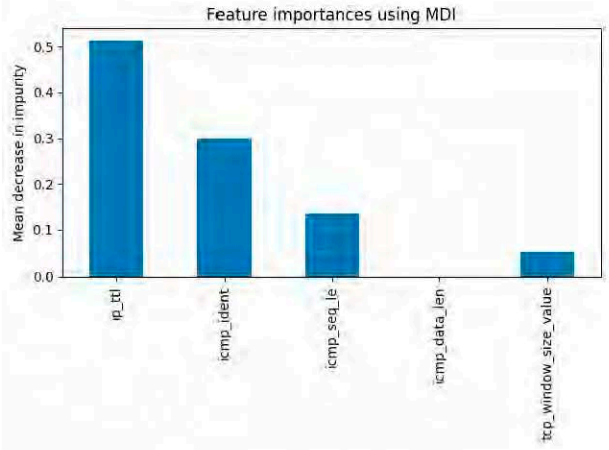


Рисунок 2.8 – Графік важливості ознак

Рисунок 2.9 – Матриця помилок дерева рішень

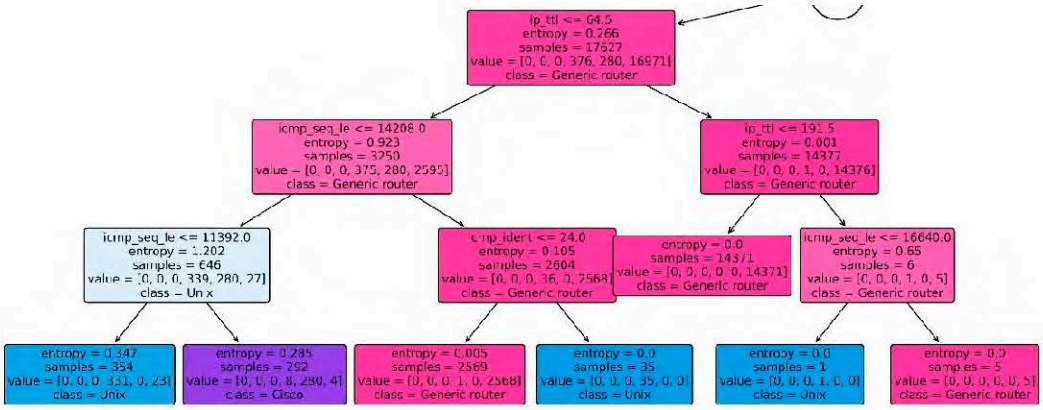


Рисунок 2.10 – Фрагмент структури дерева рішень

Наступним кроком є моделювання класифікаторів другого рівня, які будуть вирішувати задачу класифікації операційної системи в межах сімейства для більш точного визначення. Результати моделювання наведено у табл 2.4-2.9. Матриці помилок обраних класифікаторів наведено на рис. 2.11 – 2.16.

Таблиця 2.4 – Результати моделювання класифікатора версії Windows

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	0.9140	0.91346	0.91346	0.9131	342	179
Logistic regression	0.8038	0.8109	0.8109	0.8099	513	675
Multilayer Perceptron	0.773	0.7764	0.7764	0.7727	490	867
K-Nearest Neighbors	0.8350	0.8364	0.8364	0.8344	725	274
GaussianNB	0.7498	0.7501	0.7501	0.7487	985	530
Random Forest	**0.9515**	**0.9524**	**0.9524**	**0.9519**	**149**	**144**

Рисунок 2.11 – Матриця помилок класифікатора версії Windows

Таблиця 2.5 – Результати моделювання класифікатора версії Unix

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	**1.0**	**1.0**	**1.0**	**1.0**	**0**	**0**
Logistic regression	0.9252	0.9276	0.9278	0.9281	91	68
Multilayer Perceptron	0.8867	0.8873	0.8876	0.8901	107	134
K-Nearest Neighbors	0.99902	0.9989	0.99907	0.9995	0	2
GaussianNB	0.8569	0.8670	0.8681	0.8600	60	244
Random Forest	1.0	1.0	1.0	1.0	0	0

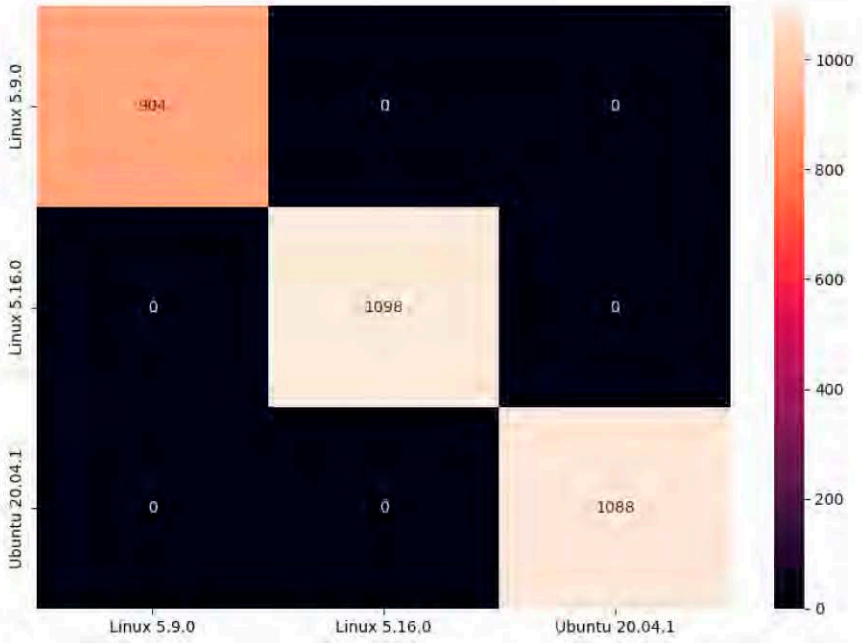


Рисунок 2.12 – Матриця помилок класифікатора версії Unix

Таблиця 2.6 – Результати моделювання класифікатора версії Cisco

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	0.9602	0.9599	0.9602	0.9614	60	47
Logistic regression	0.9493	0.9484	0.9493	0.9420	67	70
Multilayer Perceptron	0.8940	0.8924	0.8940	0.8946	119	167
K-Nearest Neighbors	0.9308	0.9312	0.9308	0.9350	29	158
GaussianNB	0.8232	0.8224	0.8232	0.8245	263	215
Random Forest	**0.9788**	**0.9776**	**0.9775**	**0.9734**	**39**	**18**

Рисунок 2.13 – Матриця помилок класифікатора версії Cisco

Таблиця 2.7 – Результати моделювання класифікатора версії MacOS

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	0.99974	0.9989	0.9991	0.9992	0	1
Logistic regression	0.9567	0.9547	0.9525	0.9558	75	86
Multilayer Perceptron	0.9953	0.9953	0.9953	0.9943	17	0
K-Nearest Neighbors	0.9990	0.9985	0.9990	0.9971	1	3
GaussianNB	0.9989	0.9957	0.9975	0.9970	3	1
Random Forest	**0.9999**	**0.9999**	**0.9999**	**0.9999**	**1**	**0**

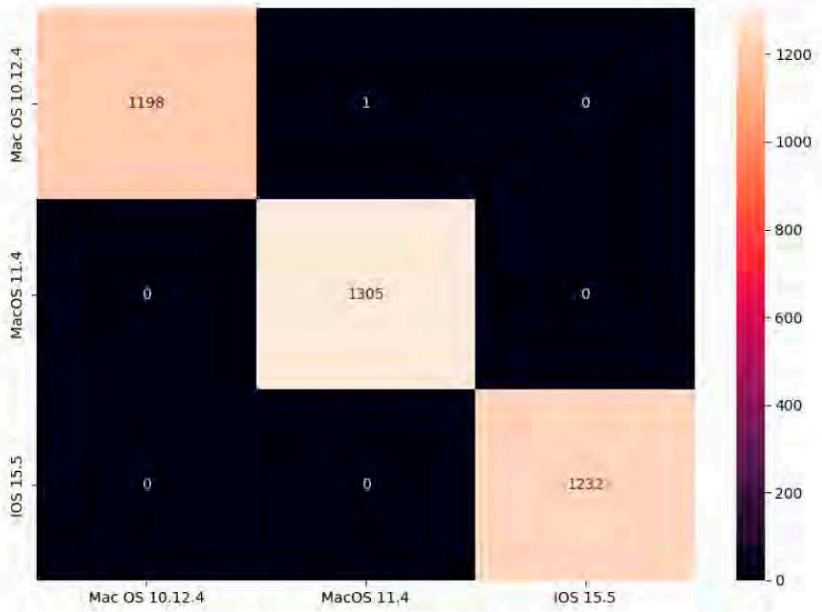


Рисунок 2.14 – Матриця помилок класифікатора версії MacOS

Таблиця 2.8 – Результати моделювання класифікатора версії Juniper

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	0.9507	0.9531	0.9541	0.9507	85	85
Logistic regression	0.9476	0.9446	0.9451	0.9476	114	67
Multilayer Perceptron	0.7924	0.7967	0.7946	0.7924	520	119
K-Nearest Neighbors	0.9634	0.9645	0.9612	0.9634	98	28
GaussianNB	0.7763	0.7765	0.7734	0.7763	386	388
Random Forest	**0.976**	**0.9734**	**0.9748**	**0.976**	**60**	**23**

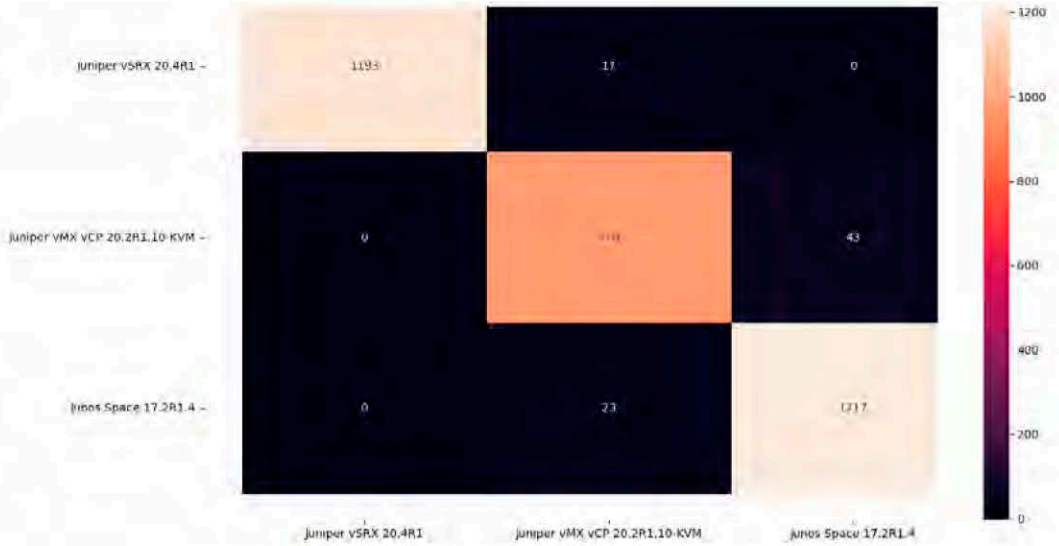


Рисунок 2.15 – Матриця помилок класифікатора версії Juniper

Таблиця 2.9 – Результати моделювання класифікатора версії Generic router

Назва класифікатора	Accuracy	Precision	Recall	F1	FP	FN
Decision Tree	0.9201	0.9276	0.9207	0.9201	124	63
Logistic regression	0.9143	0.9164	0.9145	0.9143	77	123
Multilayer Perceptron	0.7305	0.7362	0.7301	0.7305	549	82
K-Nearest Neighbors	0.8676	0.8624	0.8613	0.8676	72	238
GaussianNB	0.7691	0.7654	0.7623	0.7691	478	63
Random Forest	**0.9515**	**0.9578**	**0.9534**	**0.9515**	**71**	**42**

Рисунок 2.16 – Матриця помилок класифікатора версії Generic Router

За результатами моделювання було обрано класифікатор дерево рішень (Decision Tree) та класифікатор Випадковий ліс (Random Forest). Класифікатори мають високу точнсть визначення. завдяки високому рівню точності та швидкості навчання моделей. Класифікатор дерево рішень буде використано для класифікації сімейства операційної системи, тоді як Випадковий ліс буде використано для визначення точної версії операційної системи в межах сімейства. Вказані класифікатори будуть використані в ансамблевій моделі машинного навчання (рис. 2.17).

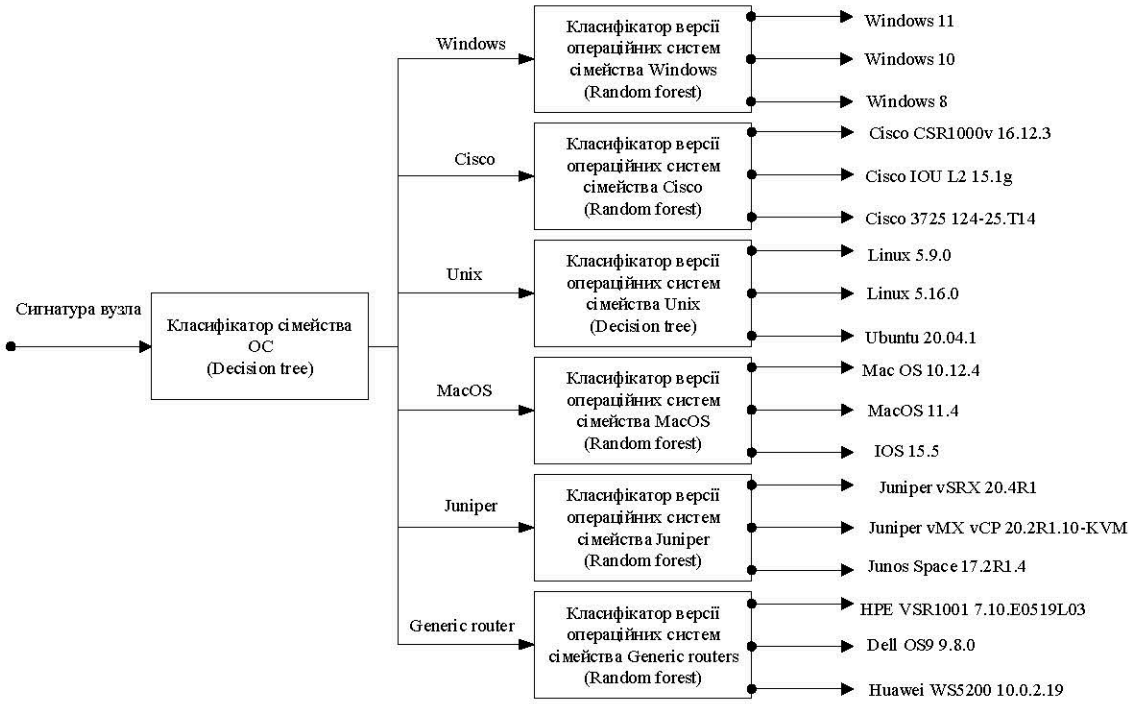


Рисунок 2.17 – Ансамблева модель машинного навчання

На вхід ансамблевої моделі надається сигнатура мережевого вузла. Далі класифікатор сімейства операційної системи визначає сімейство операційної системи як 1 серед списку: Windows, Cisco, Unix, MacOS, Juniper, Generic routers. Далі за результатами визначення сигнатура передається відповідній сімейству моделі, яка визначає точну версії операційної системи серед тих, які пропонувались під час навчання.

2.5 Розробка алгоритмів функціонування системи

Програмний засіб, що реалізує інформаційну технологію визначення операційної системи мережевих вузлів із застосуванням машинного навчання, виконує наступні функції:

- визначення операційної системи мережевого вузла;
- інформування користувача про вразливості, що пов’язані із визначеною операційною системою;

– формування звіту сканування, який може бути використаний для формування подальшої стратегії підвищення рівня захищеності мережі.

Після запуску програмного засобу, користувачі пропонується меню у вигляді вікна програми із кнопками «Обрати файл», «Розпочати визначення ОС», «Завантажити список адрес вузлів», поля для введення мережевих адрес вузлів та назви файлу трафіку, та кнопки типу radiobutton з назвами «Активний режим» та «Пасивний режим», які використовуються для вибору режиму виявлення операційної системи.

Розглянемо детальніше порядок дій при натисканні на кожну кнопку у програмному засобі:

1) «Обрати файл для аналізу»:
    а) користувачу пропонується діалогове вікно для обрання файлу типу .pcap, який буде використано для аналізу програмним засобом у пасивному режимі, користувач також може самостійно ввести шлях до файлу;
    б) у текстове поле користувач вводить IP-адресу вузла, операційну систему якого необхідно визначити -> виконується перевірка правильного введення IP-адреси, інакше користувачу повідомляється, що IP-адресу введено неправильно.
2) «Розпочати визначення ОС»:
    а) у текстове поле користувач вводить IP-адресу вузла, операційну систему якого необхідно визначити -> виконується перевірка правильного введення IP-адреси, інакше користувачу повідомляється, що IP-адресу введено неправильно;
    б) користувач обирає бажаний режим виявлення: активний або пасивний -> в разі, якщо обрано файл, доступний лише пасивний режим визначення;
    в) програма виконує визначення операційної системи віддаленого вузла з використанням модуля інтелектуального аналізу;

г) користувач отримує повідомлення із виявленою операційною системою вузла -> користувач отримує значення ймовірності приналежності операційної системи до вказаного значення, а також пропонується обрати місце збереження файлу формату CSV, який містить у собі мережеву адресу вузла, виявлену операційну систему та перелік вразливостей, які пов'язані із виявленою операційною системою.

3) «Завершити роботу»:

а) Користувач натискає на кнопку «Завершити роботу» -> програма завершує роботу із програмним засобом.

Алгоритм роботи програми у вигляді блок-схеми з розгалуженнями наведено на рис. 2.18.

Модуль інтелектуального аналізу даних за допомогою машинного навчання виконує основні операції обчислення, в результаті якого отримується результат класифікації.

Перевагою машинного навчання при вирішенні задач класифікації полягає у можливості її навчання. Суть навчання – знайти закономірності між параметрами.

У процесі навчання модель здатна виявляти складні залежності між вхідними й вихідними даними. В разі успішного навчання, модель матиме змогу надавати правильний результат на підставі даних, які були відсутні в навчальній вибірці [23].

У якості ознаки використовуються параметри пакетів протоколів, які було обрано раніше під час підготовки даних для навчання. Під час навчання класифікатору пропонуються різні зразки сигнатур операційних систем із зазначенням того, до якого класу вони відносяться (Windows, Linux тощо). Зразок, як правило, представляється як вектор значень ознак. При цьому сукупність усіх ознак повинна однозначно визначати клас, до якого належить зразок.

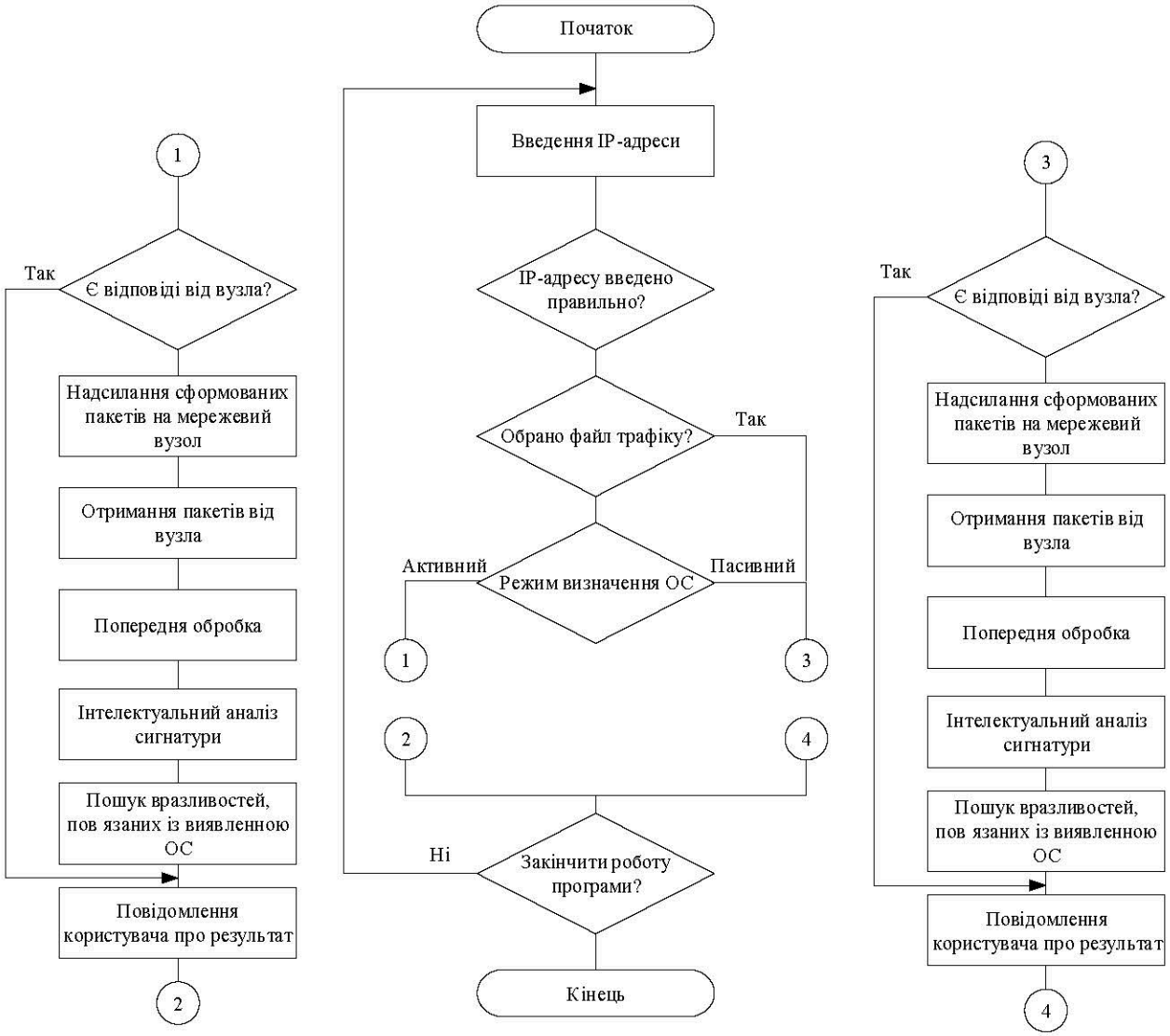


Рисунок 2.18 – Схема алгоритму роботи програми

Після закінчення моделі класифікатора можна пред'являти невідомі раніше образи і отримувати відповідь про належність до певного класу [23].

У даному розділі на основі вимог та задач, що були висунуті до проектованої інформаційної технології, було розроблено структурну схему роботи програмного засобу, що реалізує інформаційну технологію, визначено склад технічних засобів та програмного забезпечення.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ

3.1 Обґрунтування вибору інструментальних засобів розробки

Наступним етапом програмної реалізації інформаційної технології є вибір інструментальних засобів (бібліотеки мови програмування, модулі, програмні засоби), використовуючи які, будуть реалізовані основні модулі та спроектована система в цілому.

Для розробки програмного засобу буде використано мову програмування високого рівня Python. Перевагами вказаної мови є наявність великої кількості бібліотек, зокрема для роботи з моделями машинного навчання, простий синтаксис.

Наступним кроком є обрання середовища розробки. Запуск програмного засобу буде відбуватися на персональному комп'ютері користувача (фахівець з проведення тестування на проникнення, адміністратор мережі тощо).

Серед найбільш популярних середовищ варто відокремити наступні, оскільки вони передбачають комплексну та швидку розробку програмного забезпечення [24]:

1) Sublime Text 3 – умовно безкоштовне середовище розробки, яке підтримує велику кількість мов програмування, включаючи Python. За замовчуванням має базову підтримку Python, проте для полегшення та пришвидшення розробки необхідно встановлювати пакети доповнень.
2) Atom – безкоштовне середовище розробки, яка необхідно налаштувати для початку роботи. Головним недоліком є зупинка офіційної підтримки у грудні 2022 року.
3) PyCharm – найпопулярніша середа розробки. Має набір додатків та функцій, які прискорюють розробку програмного забезпечення, які встановлюютсья із середовищем за замовчуванням. Має безкоштовну та платну версії.
4) Visual Studio Code – загальне середовище розробки, яке підтримує різні мови програмування. Постачається з системою автодоповнення. Для розробки

програмних засобів мовою Python, необхідно завантажити розширення та виконати налаштування середовища розробки.

Під час розробки програмного засобу також будуть використані наступні бібліотеки мови Python:

1) pandas – програмна бібліотека, яка призначена для маніпуляції із даними, їх аналізу. Додатково бібліотека надає структури даних, можливість виконання операцій із таблицями. Особливості pandas:
   - наявність об'єкту DataFrame для швидкої та ефективної обробки даних;
   - наявність інструментів, які призначені для ефективного читання та запису даних між структурами даних;
   - інструменти для вирівнювання даних та обробки ситуацій, коли у структурі відсутні певні дані [24].
2) scikit-learn – бібліотека мови Python, яка побудована на NumPy, SciPy та Matplotlib. Призначення бібліотеки: надання ефективних інструментів для роботи із різними моделями машинного навчання, які включають у себе моделі для класифікації, кластеризації [26].
3) matplotlib – бібліотека Python, яка призначеня для побудови візуалізації різного роду дій у мові Python. Бібліотека була використана під час етапу моделювання моделей машинного навчання для наочної візуалізації результатів навчання та тестування моделей, та їх структури [27].
4) PyQt5 – комплексний набір бібліотек мови Python для створення графічного інтерфейсу програмного засобу для різних операційних систем, включаючи Windows, Linux, iOS та Android. Має вбудовний редактор вікон, який дозволяє пришвидшити розробку програмних засобів із графічним інтерфейсом [28].
5) nvdlib – бібліотека, яка дозволяє надсилати параметризовані запити до міжнародної бази вразливостей NVD. Для збільшення частоти та кількості запитів, які можна надсилати, необхідно використовувати API ключ, який можна безкоштовно отримати на сайті NVD [29].

Наявність розширень, які встановлені за замовчуванням, власного переліку доповнень, які не вимагають додаткових дій користувача для встановлення, доступ до термналів Bash та Python є основними чинниками, які роблять середовище PyCharm вибором для розробки програмного засобу, що буде реалізовувати інформаційну технологію.

3.2 Програмна реалізація

Відповідно до алгоритму роботи програмного засобу було створено ряд класів, кожен з яких виконує необхідні для роботи функції:

1) active_analysis – клас, який реалізує модуль сканування, а саме – надсилання пакетів на вказаний вузол та виклик методу прослуховування для отримання відповідей;

2) int_analysis – клас, який реалізує модуль інтелектуального аналізу, а саме – обробка сигнатури та класифікація операційної системи за наданою сигнатурою;

3) design – клас, що містить в собі інформацію про тип та розміщення елементів графічного інтерфейсу користувача;

4) file_extractor – клас, що відповідає за отримання ознак для формування сигнатури із файлу захопленого трафіку. Має 2 методи: check_all, який перевіряє наявність усіх необхідних ознак у сигнатурі, та extract_from_file, який отримує потрібні ознаки із файлу захопленого трафіку;

5) main – головний клас програми, відповідає за ініціалізацію головного вікна програми та виклик усіх інших класів із головного вікна, а також перетворює отримані результати виявлення операційної системи у зручний для користувача вигляд;

6) passive_analysis – клас, який має методи для виявлення операційної системи пасивним методом, а саме: passive_file_analysis (для виявлення операційної системи, аналізуючи наявний файл) та passive_analysis (для пасивного виявлення, прослуховуючи вказаний інтерфейс);

7) traffic_analyzer – клас, що має метод для прослуховування трафіку у мережі на наявність пакетів від вказаного вузла.

8) nvd_query – клас, що має метод для отримання переліку пов'язаних із визначеною операційною системою вразливостей.

9) reporter – клас, що має метод створення звіту сканування мережевих вузлів.

На рисунку 3.1 наведено структуру проекту, де scaler.pkl – збережена модель для масштабування даних, яка використовувалась при навчанні, model_family.pkl – збережена навчена модель класифікатора дерева рішень, що визначає сімейство операційної системи, файли з назвою model_*.pkl – збережені моделі класифікатора дерева рішень, що визначають операційну систему в межах визначеного сімейства (Windows, Unix, MacOS тощо).

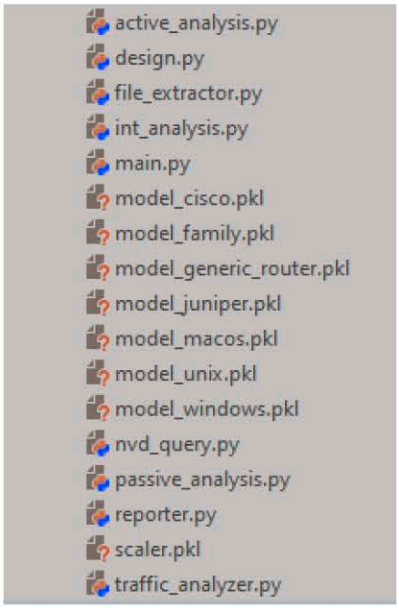


Рисунок 3.1 – Структура проекту

Текст програмного засобу наведено у додатку Б. Для перевірки розробленого програмного засобу необхідно провести тестування.

3.3 Тестування програмного засобу

Тестування буде проведено з використанням двох сценаріїв:

- перевірка правильної роботи програмного засобу;

– перевірка роботи програмного засобу при використанні неправильних даних.

Для запуску програмного засобу необхідно відкрити файли OS-Detect. Користувачеві буде відображено головне вікно програми (рис. 3.2).

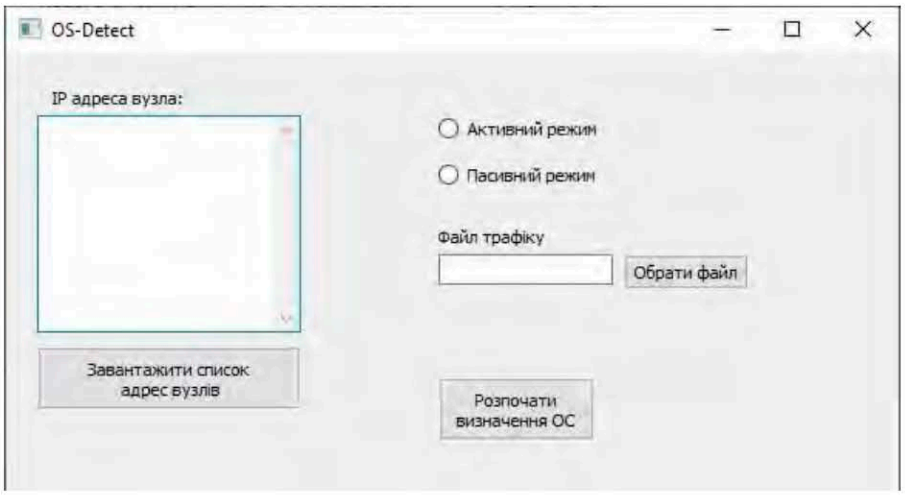


Рисунок 3.2 – Головне вікно програми

У вікні є поле для введення IP-адреси мережевого вузла, операційну систему якого потрібно виявити, кнопки керування (завантаження списку адрес вузлів, розпочати визначення операційної системи, обрати файл для аналізу, вибір режиму) та кнопка завершення роботи. Після введення IP-адреси потрібно обрати режим визначення та натиснути кнопку «Розпочати визначення ОС».

Для перевірки роботи програмного засобу в активному режимі розпочнемо виявлення операційної системи для вузла з IP-адресою 192.168.3.1. Оскільки IP-адресу введено правильно, через деякий час отримуємо результат сканування, а саме – операційна система та достовірність того, що виявлена операційна система належить визначеному сімейству та достовірність визначення операційної системи всередині сімейства (рис 3.3).

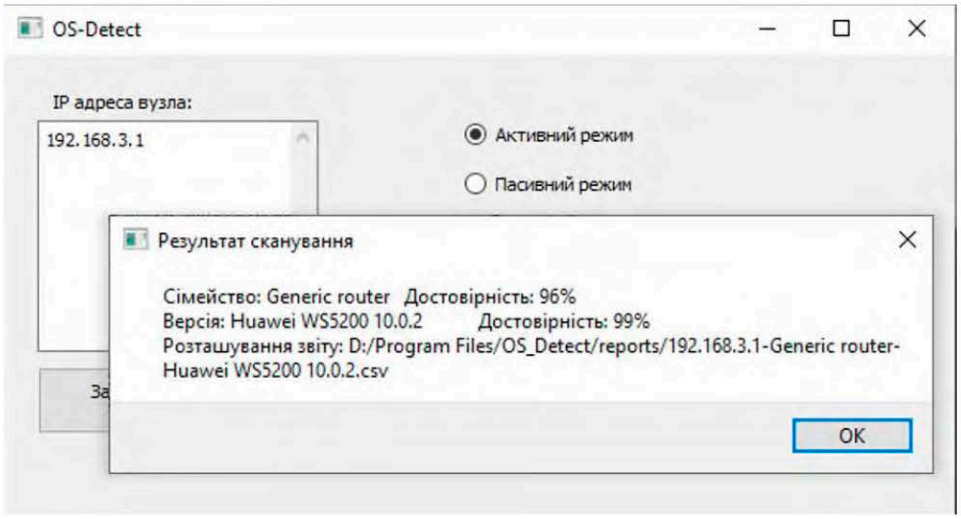


Рисунок 3.4 – Результат визначення операційної системи в активному режимі

Для перевірки роботи програмного засобу в пасивному режимі розпочнемо виявлення операційної системи для вузла з IP-адресою 192.168.3.92, де встановлено Windows 7 Professional. Результат виявлення наведено на рис. 3.5.

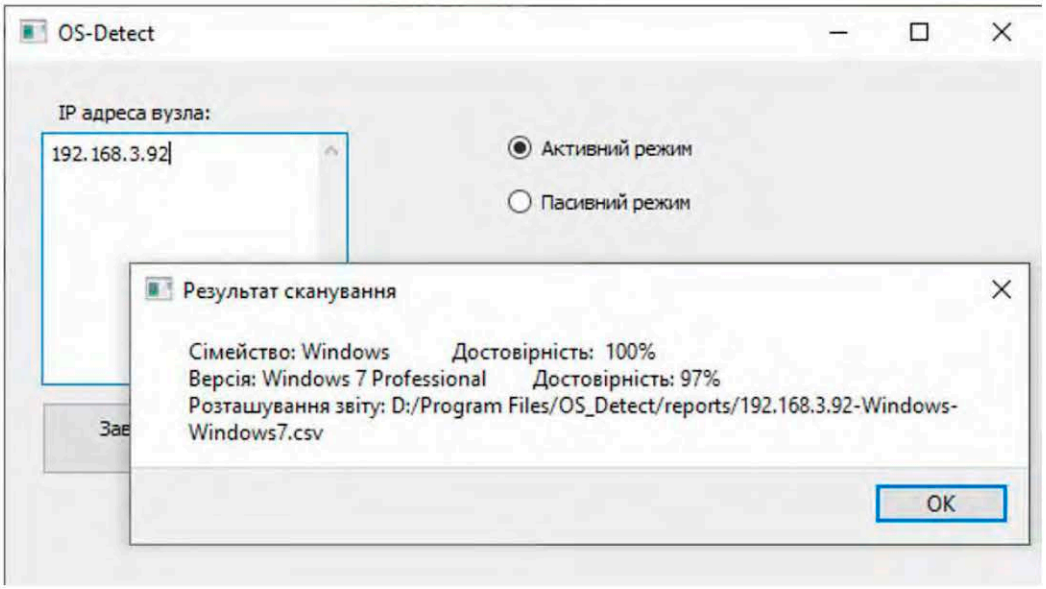
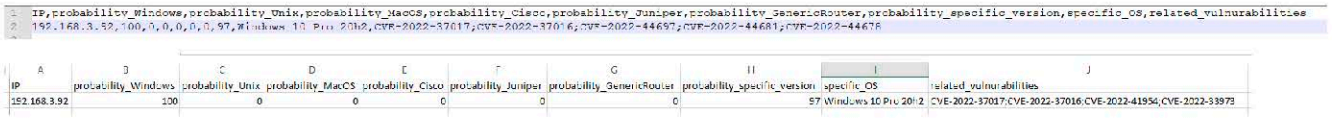


Рисунок 3.5 – Результат визначення операційної системи в пасивному режимі

Після кожного сканування зберігається звіт у папці з програмним засобом у форматі файлу CSV. Звіт з сканування (вміст файлу та приклад завантаження звіту у програмний засіб Microsoft Excel) наведено на рис. 3.6. Деталі вразливостей можна переглянути на сайтах CVE та NVD, здійснивши пошук за вказаними значеннями (рис. 3.7).



IP	probability_Windows	probability_Unix	probability_MacOS	probability_Cisco	probability_Juniper	probability_GenericRouter	probability_specific_version	specific_OS	related_vulnurabilities
192.168.3.92	100	0	0	0	0	0	97	Windows 10 Pro 20h2	CVE-2022-37017;CVE-2022-37016;CVE-2022-41954;CVE-2022-33973

Рисунок 3.6 – Вміст звіту та приклад завантаження звіту у Microsoft Excel

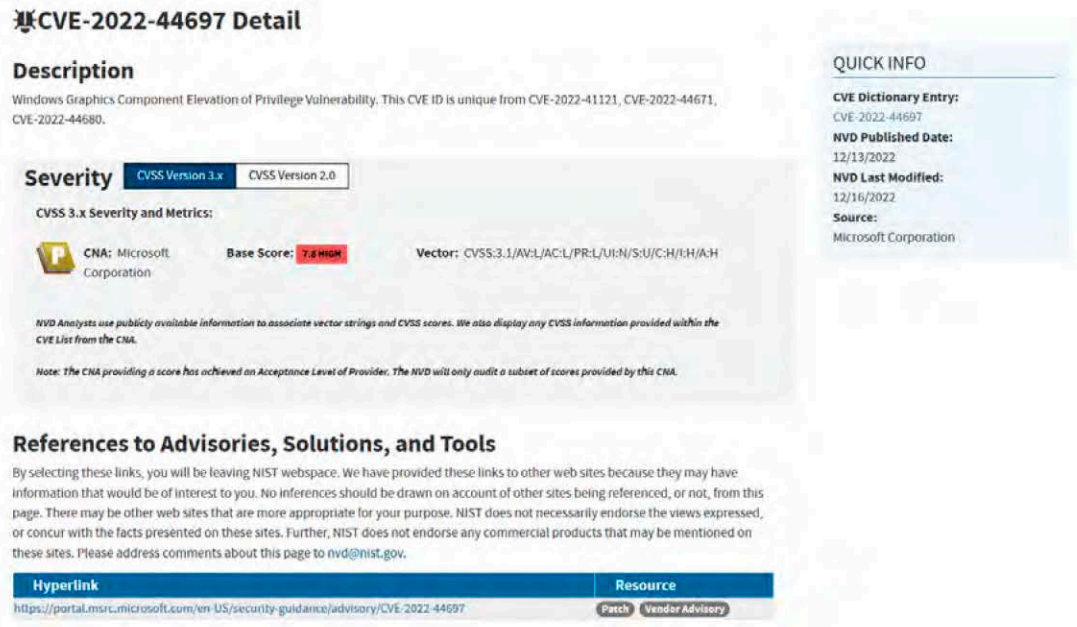


Рисунок 3.7 – Деталі про вразливість з сайту NVD

Для перевірки обробки помилок, необхідно ввести неправильні дані. Спробуємо ввести наступну IP-адреси в поле: 300.49.85.12, імітуючи таким чином випадкові помилки користувача. На рис. 3.6 зображено відповідне повідомлення, що виводиться користувачу при введенні неправильної IP-адреси.

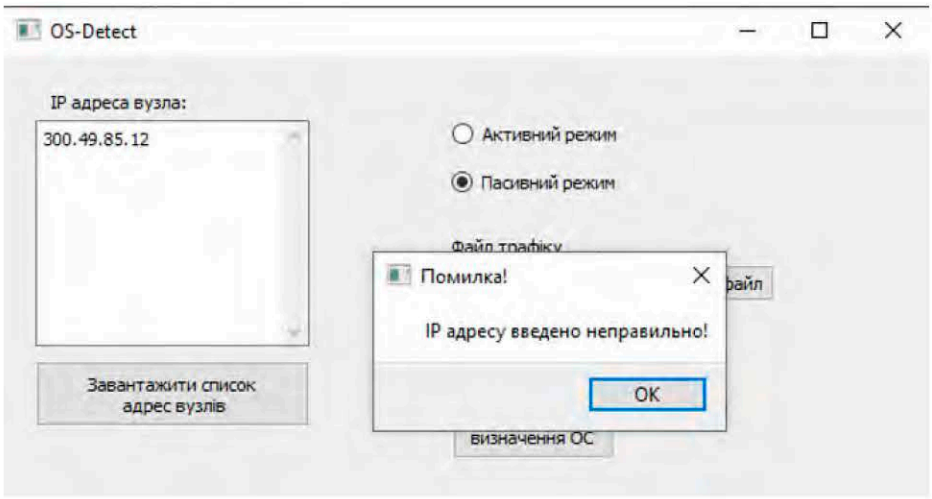


Рисунок 3.6 – Повідомлення про помилку при введенні IP-адреси

Порівняємо результати виявлення програмного засобу та популярного засобу Nmap. Для порівняння було виконано визначення операційної системи вузлів, які розглядались раніше (машрутизатор Huawei та персональний комп'ютер із встановленною Windows 7 Professional). Результати сканування Nmap наведено на рис. 3.7 та рис. 3.8.

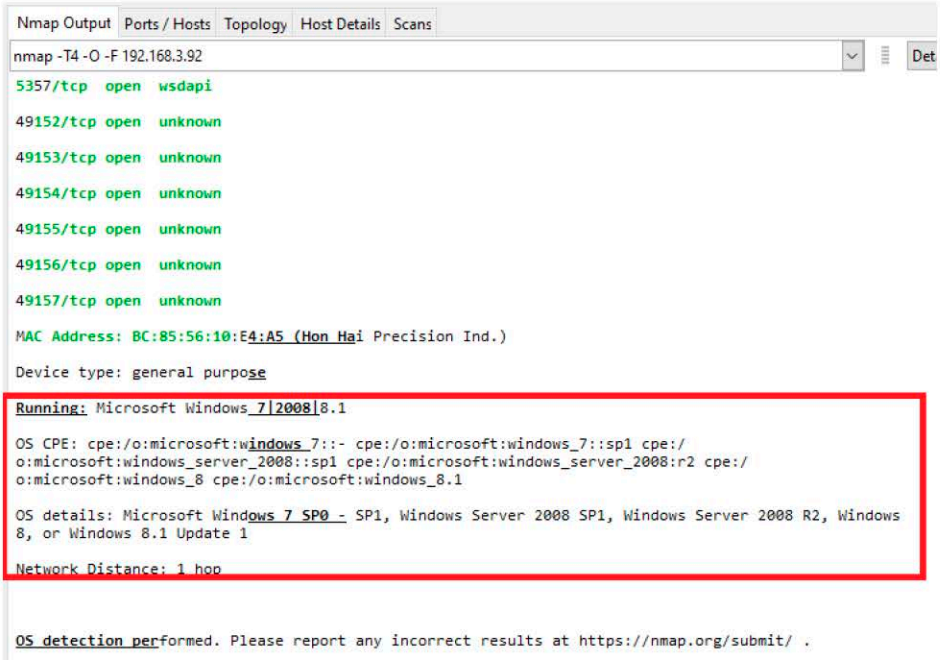


Рисунок 3.7 – Результат виявлення операційної системи, використовуючи Nmap

Рисунок 3.8 – Результат виявлення операційної системи, використовуючи Nmap

Отримані результати свідчать про те, що Nmap не зміг точно визначити встановлені операційні системи (через відсутніх знайдених закритих портів), надавши користувачу результат у вигляді 3 операційних систем Windows (без зазначення достовірності визначення.

Проаналізувавши усі отримані результати можна зробити висновок, що розроблений програмний засіб дозволив підвищити достовірність виявлення операційної системи мережевого вузла, надаючи достовірність визначення конкретної операційної системи та приналежності операційної системи до одного із загальних класів, а також покращує процес виконання тестування на проникнення, оскільки користувач має змогу отримати перелік пов'язаних із виявленою операційною системою вразливостей, на які варто звернути увагу.

Було виконано тестування програмного засобу, використовуючи два сценарії (введення правильних і неправильних даних). За результатами тестування можна стверджувати, що програмний засіб працює правильно і виконується обробка помилок в разі їх появи.

4 ЕКОНОМІЧНЕ ОБҐРУНТУВАННЯ

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» відноситься до науково-технічних робіт, які орієнтовані на виведення на ринок (або рішення про виведення науково-технічної розробки на ринок може бути прийнято у процесі проведення самої роботи), тобто коли відбувається так звана комерціалізація науково-технічної розробки. Цей напрямок є пріоритетним, оскільки результатами розробки можуть користуватися інші споживачі, отримуючи при цьому певний економічний ефект. Але для цього потрібно знайти потенційного інвестора, який би взявся за реалізацію цього проекту і переконати його в економічній доцільності такого кроку.

Для наведеного випадку нами мають бути виконані такі етапи робіт:

1) проведено комерційний аудит науково-технічної розробки, тобто встановлення її науково-технічного рівня та комерційного потенціалу;

2) розраховано витрати на здійснення науково-технічної розробки;

3) розрахована економічна ефективність науково-технічної розробки у випадку її впровадження і комерціалізації потенційним інвестором і проведено обґрунтування економічної доцільності комерціалізації потенційним інвестором.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» є оцінювання науково-технічного

рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1 [30].

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних
Ринкові переваги (недоліки)					
	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
	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
	Технічні та споживчі властивості продукту значно гірші, ніж в	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	Технічні та споживчі властивості продукту значно кращі,
	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
Ринкові перспективи					
	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
	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає

Продовження таблиці 4.1

Практична здійсненність					
	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
	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
0	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
1	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
2	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 повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Для проведення технологічного аудиту залучені 3 незалежних експерти: керівник магістерської роботи доцент кафедри захисту інформації Вінницького національного технічного університету Куперштейн Л. М.; доцент кафедри захисту інформації Вінницького національного технічного університету Баришев Ю. В.; доцент кафедри захисту інформації Вінницького національного технічного університету Войтович О. П. Результати оцінювання науково-

технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	Куперштейн Л. М.	Баришев Ю. В.	Войтович О. П.
	Бали:		
1. Технічна здійсненність концепції	5	5	5
2. Ринкові переваги (наявність аналогів)	1	2	2
3. Ринкові переваги (ціна продукту)	1	1	1
4. Ринкові переваги (технічні властивості)	1	2	2
5. Ринкові переваги (експлуатаційні витрати)	2	2	2
6. Ринкові перспективи (розмір ринку)	3	4	3
7. Ринкові перспективи (конкуренція)	3	2	3
8. Практична здійсненність (наявність фахівців)	5	5	5
9. Практична здійсненність (наявність фінансів)	3	4	3
10. Практична здійсненність (необхідність нових матеріалів)	5	5	5
11. Практична здійсненність (термін реалізації)	5	4	4
12. Практична здійсненність (розробка документів)	4	5	4
Сума балів	38	41	39
Середньоарифметична сума балів $СБ_c$	39,3		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3 [30].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ , розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія визначення операційної системи мережевих

вузлів із застосуванням машинного навчання» становить 39,3 бала, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

4.2 Розрахунок узагальненого коефіцієнта якості розробки

Окрім комерційного аудиту розробки доцільно також розглянути технічний рівень якості розробки, розглянувши її основні технічні показники. Ці показники по-різному впливають на загальну якість проектної розробки.

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення розрахуємо за формулою [31]:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i \,, \tag{4.1}$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

α_i – коефіцієнт, який враховує питому вагу *і-го* технічного показника в загальній якості розробки.

Коефіцієнт α_i визначається експертним шляхом і при цьому має виконуватись умова $\sum_{i=1}^{k} \alpha_i = 1$;

β_i – відносне значення *і-го* технічного показника якості нової розробки.

Відносні значення β_i для різних випадків розраховуємо за такими формулами:

для показників, зростання яких вказує на підвищення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ні}}{I_{аі}} \,, \tag{4.2}$$

де $I_{ні}$ та $I_{на}$ – чисельні значення конкретного *і*-го технічного показника якості відповідно для нової розробки та аналога;

для показників, зростання яких вказує на погіршення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ai}}{I_{ni}}; \qquad (4.3)$$

Використовуючи наведені залежності можемо проаналізувати та порівняти техніко-економічні характеристики аналогу та розробки на основі отриманих наявних та проектних показників, а результати порівняння зведемо до таблиці 4.4.

Таблиця 4.4 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Достовірність визначення	%	85	90	1,1	0,35
Масштабованість	інат	1	5	5	0,2
Універсальність	бал	6	8	1,33	0,05
Швидкість визначення	мс	8	2	4	0,25
Кількість підтримуваних операційних систем	шт	2	3	1,5	0,15

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 1{,}1 \cdot 0{,}35 + 5 \cdot 0{,}2 + 1{,}33 \cdot 0{,}05 + 4 \cdot 0{,}25 + 1{,}5 \cdot 0{,}15 = 2{,}68.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 2,68 рази, причому розроблене програмне забезпечення суттєво відрізняється від аналогів за такими параметрами як:

- швидкість визначення (в активному режимі);
- достовірність визначення;
- масштабованість;
- універсальність;

– можливість виконання визначення операційної системи, використовуючи файл захопленого трафіку.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

4.3.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою [30]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (4.4)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=21 дні.

$$З_о = 17200{,}00 \cdot 21 / 21 = 17200{,}00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 4.5 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	17200,00	819,05	21	17200,00
Інженер-розробник програмного забезпечення	16400,00	780,95	21	16400,00
Науковий консультант з проблем машинного навчання	15500,00	738,10	4	2952,38
Лаборант	6850,00	326,19	18	5871,43
Всього				42423,81

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \qquad (4.5)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}}, \qquad (4.6)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної місячної заробітної плати (в залежності від діючого законодавства), приймемо $Мм$=6700,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б) [30];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих

об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

$T_р$ – середнє число робочих днів в місяці, приблизно $T_р$ = 21 дн;

$t_{зм}$ – тривалість зміни, год.

$C_1 = 6700{,}00 \cdot 1{,}10 \cdot 1{,}65 / (21 \cdot 8) = 72{,}38$ *грн.*

$З_{р1} = 72{,}38 \cdot 8{,}30 = 600{,}79$ *грн.*

Таблиця 4.8 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Підготовка робочого місця інженера-розробника ПЗ	8,30	2	1,10	72,38	600,79
Інсталяція програмного забезпечення середовища розробки	6,20	3	1,35	88,83	550,78
Формування кодів програмних блоків	7,50	5	1,70	111,87	839,00
Формування бази даних для забезпечення машинного навчання	16,00	2	1,10	72,38	1158,14
Контроль проходження експериментів	8,00	4	1,50	98,71	789,64
Всього					3938,34

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \qquad (4.7)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 11%.

$З_{дод} = (42423{,}81 + 3938{,}34) \cdot 11 / 100\% = 5099{,}84$ *грн.*

4.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.8)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (42423,81 + 3938,34 + 5099,84) · 22 / 100% = 11321,64 грн.

4.3.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання».

Витрати на матеріали ($М$), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$М = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} В_j \cdot Ц_{вj}, \quad (4.9)$$

де $Н_j$ – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 ... 1,15*);

$В_j$ – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$М_1$ = 2,0 · 225,00 · 1,1 - 0 · 0 = 495,00 грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 4.9 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Папір офісний UP! (Underprice) A4 80 г/м білий	225,00	2,0	0	0	495,00
Папір офісний Crystal A5 80 г/м CRYSTAL PRO білий	139,00	3,0	0	0	458,70
Органайзер офісний FAX	200,00	3,0	0	0	660,00
Набір канцелярський офісний FAX	210,00	3,0	0	0	693,00
Картридж для принтера	980,00	1,0	0	0	1078,00
Диск оптичний CD-RW	23,00	2,0	0	0	50,60
USB флеш накопичувач KINGSTON 32GB DATATRAVELER 100 GENERATION 3 USB3.0 (DT100G3/32GB)	169,00	1,0	0	0	185,90
Всього					3621,20

4.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі ($К_в$), які використовують при проведенні НДР на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» відсутні.

4.3.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення, транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (4.10)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, (K_i = *1,10…1,12);*

k – кількість найменувань устаткування.

$B_{спец}$ = *4860,00 · 1 · 1,1 = 5346,00 грн.*

Отримані результати зведемо до таблиці:

Таблиця 4.11 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Оперативна пам'ять 8GB	1	4860,00	5346,00
Відеокарта AMD Radeon HD 7670M	1	9740,00	10714,00
Диск HGST Travelstar 2.5-Inch 1TB	1	5600,00	6160,00
Всього			22220,00

4.3.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення. Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.11)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу

тощо, (K_i = *1,10…1,12); k* – кількість найменувань програмних засобів.

$B_{прг}$ = *7860,00 · 1 · 1,1 = 8646,00 грн.*

Отримані результати зведемо до таблиці.

Таблиця 4.12 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Прикладне програмне забезпечення розробки	1	7860,00	8646,00
Середовище програмування PyCharm	1	850,00	935,00
Всього			9581,00

4.3.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.12)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років. Проведені розрахунки зведемо до таблиці.

$A_{обл}$ = *(52420,00 · 2) / (2 · 12) = 4368,33 грн.*

Таблиця 4.13 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер проведення розробки та моделювання HP Envy DV7, процесор AMD A8-4500M APU	52420,00	2	2	4368,33
Робоче місце інженера-розробника ПЗ	9200,00	5	2	306,67
Пристрої передачі даних	7510,00	4	2	312,92

Продовження таблиці 4.13

Оргтехніка	6750,00	4	2	281,25
Пристрій виводу інформації	6740,00	5	2	224,67
Приміщення лабораторії	632000,00	20	2	5266,67
ОС Windows 11	8570,00	2	2	714,17
Прикладний пакет Microsoft Office 2019	7825,00	2	2	652,08
Всього				12126,75

4.3.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.13)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо Ц$_e$ = 6,20 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,30 \cdot 160,0 \cdot 6,20 \cdot 0,95 / 0,97 = 297,60$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 4.14 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Персональний комп'ютер проведення розробки та моделювання HP Envy DV7, процесор AMD A8-4500M APU	0,30	160,0	297,60
Робоче місце інженера-розробника ПЗ	0,12	160,0	119,04
Пристрої передачі даних	0,01	160,0	9,92
Пристрій виводу інформації	0,42	20,0	52,08
Оргтехніка	0,50	3,0	9,30
Всього			487,94

4.3.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із

застосуванням машинного навчання» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов'язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з'їзди, конференції, наради, пов'язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (4.14)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{СВ}$ = 20%.

$В_{св}$ = (42423,81 + 3938,34) · 20 / 100% = 9272,43 грн.

4.3.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (4.15)$$

де $Н_{СП}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{СП}$= 30%.

$В_{сп}$ = (42423,81 + 3938,34) · 30 / 100% = 13908,65 грн.

4.3.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.16)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів} = 50\%$.

$I_в$ = (42423,81 + 3938,34) · 50 / 100% = 23181,08 грн.

4.3.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (4.17)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв} = 100\%$.

$В_{нзв}$ = (42423,81 + 3938,34) · 100 / 100% = 46362,15 грн.

Витрати на проведення науково-дослідної роботи на тему «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв}. \quad (4.18)$$

$В_{заг}$= 42423,81 +3938,34 +5099,84 +11321,63783 +3621,20 +0,00 +22220,00 + 9581,00 + 12126,75 +487,94 +9272,43 +13908,65 +23181,08 +46362,15 = =203544,82 грн.

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \quad (4.19)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,95.

ЗВ = 203544,82 / 0,95 = 214257,71 грн.

4.4 Розрахунок економічної ефективності науково-технічної розробки при її можливій комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

Результати дослідження проведені за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» передбачають комерціалізацію протягом 4-х років реалізації на ринку.

В цьому випадку майбутній економічний ефект буде формуватися на основі таких даних:

ΔN – збільшення кількості споживачів продукту, у періоди часу, що аналізуються, від покращення його певних характеристик;

Показник	1-й рік	2-й рік	3-й рік	4-й рік
Збільшення кількості споживачів, осіб	850	1000	1500	1200

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 5000 осіб;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 1020,00 грн;

$\pm\Delta Ц_o$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 410,03 грн.

Можливе збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ для кожного із 4-х років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховуємо за формулою [30]:

$$\Delta П_i = (\pm\Delta Ц_o \cdot N + Ц_o \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.20)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2022 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0,8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту). Приймемо $\rho = 40\%$;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2022 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$\Delta П_1 = (410,03 \cdot 5000,00 + 1430,03 \cdot 850) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 889047,50$ грн.

Збільшення чистого прибутку 2-го року:

$\Delta П_2 = (410,03 \cdot 5000,00 + 1430,03 \cdot 1850) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 1278358,87$ грн.

Збільшення чистого прибутку 3-го року:

$\Delta П_3 = (410,03 \cdot 5000,00 + 1430,03 \cdot 3350) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 1862325,92$ грн.

Збільшення чистого прибутку 4-го року:

$\Delta П_4 = (410,03 \cdot 5000,00 + 1430,03 \cdot 4550) \cdot 0,83 \cdot 0,4 \cdot (1\text{-}0,18/100\%) = 2329499,56$ грн.

Приведена вартість збільшення всіх чистих прибутків $ПП$, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.21)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,25;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$ПП = 889047,50/(1+0,25)^1+1278358,87/(1+0,25)^2+1862325,92/(1+0,25)^3+$$
$$+2329499,56/(1+0,25)^4=711238,00+818149,67+953510,87+954163,02$$
$$= 3437061,56 \text{ грн.}$$

Величина початкових інвестицій PV, які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \quad (4.22)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 214257,71 грн.

$PV = k_{інв} \cdot ЗВ = 2 \cdot 214257,71 = 428515,42$ грн.

Абсолютний економічний ефект $E_{абс}$ для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \quad (4.23)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 3437061,56 грн;

PV – теперішня вартість початкових інвестицій, 428515,42 грн.

$E_{абс} = ПП - PV$ = 3437061,56 - 428515,42 = 3008546,14 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \quad (4.24)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 3008546,14 грн;

PV – теперішня вартість початкових інвестицій, 428515,42 грн;

$T_{ж}$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+3008546,14/428515,42)^{1/4} = 0,68.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \quad (4.25)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2022 році в Україні d =0,11;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,3.

$\tau_{мін}$ = 0,11+0,3 = 0,41 < 0,68 свідчить про те, що внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені потенційним інвестором у

впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \tag{4.26}$$

де $E_в$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок} = 1 / 0{,}68 = 1{,}46$ р.

$T_{ок} <$ 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання» становить 39,3 бала, що, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього). При оцінюванні за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 2,68 рази.

При оцінюванні рівня конкурентоспроможності, згідно узагальненого коефіцієнту конкурентоспроможності розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 2,68 рази. Також термін окупності становить 1,46 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження даної розробки та виведення її на ринок.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання».

ВИСНОВКИ

У ході виконання магістерської кваліфікаційної роботи було виконано усі поставлені задачі. Було проведено аналіз мережевих пристроїв та їх особливостей. Було проаналізовано актуальність вирішення задачі визначення операційної системи мережевого вузла у кібербезпеці. Проведено аналіз методів та існуючих засобів виявлення операційної системи мережевих вузлів. Визначено, що проблема є доволі актуальною. Прийнято рішення покращити процес тестування на проникнення на етапах збору даних та сканування цільового вузла шляхом розробки інформаційної технології. Для виконання даної задачі було проведено моделювання ряду класифікаторів. Спроектовано архітектуру ансамблевої моделі машинного навчання, відображено результати навчання у вигляді матриці помилок, оцінок навчання та кількості помилок першого та другого роду. На основі отриманих результатів обрано найкращі – дерево рішень та випадковий ліс. Також було обрано технології програмування для вирішення задачі.

Виконано розробку структури інформаційної технології визначення операційної системи мережевих вузлів із застосуванням машинного навчання, що включає у себе процес генерації трафіку, процес збору трафіку, процес аналізу трафіку, процес попередньої обробки та процес інтелектуального аналізу.

Виконано обґрунтування вибору інструментальних засобів розробки, результати: мова програмування – Python, середовище розробки – PyCharm. На основі створених вимог до програмного засобу, створених алгоритмів та архітектури системи, створено програмний засіб, що реалізує інформаційну технологію визначення операційної системи мережевих вузлів. Створений програмний засіб має простий та зрозумілий інтерфейс. Для використання потрібно лише запустити виконуваний файл, який знаходиться у папці разом із необхідними файлами.

Проведено тестування розробленого програмного засобу. Тестування проводилось для двох сценаріїв роботи: при використанні правильних даних та при використанні неправильних даних. Помилки при введенні початкових даних обробляються, а відповідні повідомлення надаються користувачу для усунення проблем. Результати роботи програми порівняно із результатами роботи існуючого програмного засобу Nmap, а зокрема зазначено, що Nmap надає більш узагальнені результати визначення операційної системи без зазначення достовірності приналежності операційної системи до певної конкретної операційної системи. Також розроблений програмний засіб надає користувачеві актуальну інформацію щодо пов'язаних вразливостей у звіті сканування, який може бути використаний іншими програмами оформлення звітів.

Згідно проведених досліджень щодо рівня комерційного потенціалу розробки, результати вказують на комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки є вищим за середній). При оцінюванні рівня конкурентоспроможності, було визначено, що науково-технічна розробка переважає існуючі аналоги приблизно в 2,68 рази. Також термін окупності становить 1,46 р., що свідчить про комерційну привабливість науково-технічної розробки. Вказані результати свідчать про доцільність проведення наукових досліджень.

Даний програмний засіб є корисним застосунком, який покращує процес проведення тестування на проникнення, а саме під час збору даних про цілі проведення тестування. Також адміністратори мереж можуть використовувати розроблений програмний засіб для виявлення неавторизованих пристроїв у мережі, порівнюючи звіти сканування за певні проміжки часу.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Комп'ютерна програма «Засіб для визначення типу операційної системи віддаленого вузла» / А. В. Борусевич, Л. М. Куперштейн // Свідоцтво про реєстрацію авторського права на твір № 112069 від 22.02.2022

2. Борусевич А. В., Куперштейн Л. М. Інтелектуальна інформаційна технологія визначення типу операційної системи віддаленого вузла : Електронний ресурс. URL: https://kn.khmnu.edu.ua/wp-content/uploads/sites/18/apkn2022_corpuspaper.pdf

3. Борусевич А., Куперштейн. Л. Машинне навчання у задачах виявлення типу операційної системи віддаленого хоста : Електронний ресурс. URL: https://conferences.vntu.edu.ua/index.php/mn/mn2022/paper/view/14534

4. Борусевич А., Куперштейн. Л. Аналіз факторів у задачах визначення типу операційної системи : Електронний ресурс. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2022/paper/view/16068

5. Kupershtein L., Martyniuk T., Voitovych O., Borusevych A. Remote Host Operation System Type Detection Based on Machine Learning Approach : Електронний ресурс. URL: http://ceur-ws.org/Vol-3106/Paper_7.pdf

6. Борусевич А., Куперштейн Л. Аналіз методів та засобів виявлення типу операційної системи віддаленого вузла : Електронний ресурс. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2021/paper/view/12193

7. Cloudflare. What is a router? : веб-сайт. URL: https://www.cloudflare.com/learning/network-layer/what-is-a-router/ (дата звернення 05.09.2022)

8. Knowing the operating systems running on your network devices: веб-сайт. URL : https://www.networkworld.com/article/2299932/tech-tip--knowing-the-operating-systems-running-on-your-network-devices.html (дата звернення 10.11.2022).

9. Huawei. Evaluation on Security Risks of the router on the Network - NE20E-S V800R011C10 Configuration Guide - Security Hardening 02 : веб-сайт.

URL: https://support.huawei.com/enterprise/en/doc/EDOC1100125517/d8b85efb/evaluation-on-security-risks-of-the-router-on-the-network (дата звернення 09.09.2022)

10. Router security survey 2022 : веб-сайт. URL: https://www.broadbandgenie.co.uk/blog/20220106-router-security-survey-2022 (дата звернення 11.11.2022)

11. Field engineer blog. Top 7 Enterprise Router Vendors to Consider in 2022: веб-сайт. URL: https://www.fieldengineer.com/blogs/top-seven-enterprise-router-vendors-consider-2018 (дата звернення 10.09.2022)

12. What is penetration testing : веб-сайт. URL : https://www.imperva.com/learn/application-security/penetration-testing/ (дата звернення 10.12.2022)

13. What is penetration testing? : веб-сайт. URL : https://www.cloudflare.com/learning/security/glossary/what-is-penetration-testing/ (дата звернення 10.12.2022)

14. Fingerprinting Methods Avoided by Nmap : веб-сайт. URL: https://nmap.org/book/osdetect-other-methods.html#osdetect-passive (дата звернення 04.10.2022).

15. Nmap: the Network Mapper - Free Security Scanner : веб-сайт. URL: https://nmap.org/ (дата звернення 14.10.2022).

16. p0f v3 (version 3.09b) : веб-сайт. URL: https://lcamtuf.coredump.cx/p0f3/ (дата звернення 17.10.2022).

17. NetworkMiner : веб-сайт. URL: https://www.netresec.com/?page=NetworkMiner (дата звернення 17.10.2022).

18. CVE. Overview : веб-сайт. URL: https://www.cve.org/About/Overview (дата звернення 29.09.2022)

19. NVD. Home: веб-сайт. URL: https://nvd.nist.gov/ (дата звернення 29.09.2022)

20. OS Detection Techniques : веб-сайт. URL: https://jonathansblog.co.uk/os-detection-techniques (дата звернення 17.10.2022).

21. Wireshark Tutorial: IdentifyingHosts and Users : веб-сайт. URL: https://unit42.paloaltonetworks.com/using-wireshark-identifying-hosts-and-users/ (дата звернення 19.10.202).

22. GNS3. Software : веб-сайт. URL: https://www.gns3.com/ software (дата звернення 23.09.2022)

23. Васюра А.С., Мартинюк Т.Б., Куперштейн Л.М. Методи та засоби нейроподібної обробки даних для систем керування. Монографія. Вінниця: УНІВЕРСУМ–Вінниця, 2008. 175 с.

24. 9 Best Python IDEs and Code Editors : веб-сайт. URL : https://www.programiz.com/python-programming/ide (дата звернення 10.11.2022).

25. About pandas : веб-сайт. URL: https://pandas.pydata.org/about/index.html (дата звернення 15.11.2022).

26. Scikit Learn – Introduction : веб-сайт. URL: https://www.tutorialspoint.com/scikit_learn/scikit_learn_introduction.htm (дата звернення 05.12.2022).

27. Matplotlib: Visualization with Python : веб-сайт. URL: https://matplotlib.org/ (дата звернення 05.12.2022).

28. What is PyQt? : веб-сайт. URL: https://www.riverbankcomputing.com/software/pyqt/ (дата звернення 06.12.2022).

29. NVDLib: NIST National Vulnerability Database API Wrapper : веб-сайт. URL : https://nvdlib.com/en/latest/ (дата звернення 01.12.2022).

30. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.

31. Кавецький В. В. Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТКИ

Додаток А

ПРОТОКОЛ ПЕРЕВІРКИ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія визначення операційної системи мережевих вузлів із застосуванням машинного навчання

Автор роботи: Борусевич Артур Вячеславович

Тип роботи: магістерська кваліфікаційна робота

Підрозділ кафедра захисту інформації ФІТКІ
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність – 93,7%. Схожість – 6,3%.

Аналіз звіту подібності (відмітити потрібне):

√ 1. Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

☐ 3. Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Особа, відповідальна за перевірку _____ Каплун В. А.
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи _____ Борусевич А.В
(підпис) (прізвище, ініціали)

Керівник роботи _____ Кифорчеєнко М.Ф.
(підпис) (прізвище, ініціали)

Додаток Б
Текст програми

active_analysis.py

```
import traffic_sniffer as tf
import os
import sql
import socket
from threading import Thread


class active_analysis:
    def active_analisys(self, interface_name, ip, bool_http_ua):
        tf1 = tf.tf()
        thread_capture = Thread(target=tf1.capture, args=(interface_name, ip,
False, bool_http_ua))
        thread_capture.start()
        self.send_probes(self, ip)
        self.send_probes(self, ip)
        self.send_probes(self, ip)

        path = "signature.txt"
        f = open(path, "rt")
        signature_arr = f.read()
        print('sign...')
        print(signature_arr)
        signature_array = signature_arr.split("*")
        os, proba = sql1.search(signature_array)
        return os, proba

    def send_probes(self, ip_target):
        os.system('ping %s -n 4' % (ip_target,))
        s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        port = [135, 22, 443, 53, 23]
        for i in port:
            try:
                s.connect((ip_target, i))
                s.close()
                break
            except:
                print("Port closed: " + str(port))
```

design.py

```
from PyQt5 import QtCore, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.setWindowModality(QtCore.Qt.NonModal)
        MainWindow.setFixedSize(597, 247) #497 247
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.button_start_analysis = QtWidgets.QPushButton(self.centralwidget)
        self.button_start_analysis.setGeometry(QtCore.QRect(20, 110, 151, 51))
        self.button_start_analysis.setObjectName("button_start_analysis")
        self.radio_active = QtWidgets.QRadioButton(self.centralwidget)
        self.radio_active.setGeometry(QtCore.QRect(210, 110, 111, 17))
        self.radio_active.setObjectName("radio_active")
        self.button_exit = QtWidgets.QPushButton(self.centralwidget)
        self.button_exit.setGeometry(QtCore.QRect(360, 140, 111, 23))
```

```
        self.button_exit.setObjectName("button_exit")
        self.radio_passive      =      QtWidgets.QRadioButton(self.centralwidget)
        self.radio_passive.setGeometry(QtCore.QRect(210,    140,    111,    17))
        self.radio_passive.setObjectName("radio_passive")
        self.text_ip_input       =       QtWidgets.QLineEdit(self.centralwidget)
        self.text_ip_input.setGeometry(QtCore.QRect(20,     40,     171,     20))
        self.text_ip_input.setObjectName("text_ip_input")
        self.label_input_ip        =        QtWidgets.QLabel(self.centralwidget)
        self.label_input_ip.setGeometry(QtCore.QRect(40,     20,     131,     16))
        self.label_input_ip.setLayoutDirection(QtCore.Qt.LeftToRight)
        self.label_input_ip.setObjectName("label_input_ip")
        self.button_choose_file    =    QtWidgets.QPushButton(self.centralwidget)
        self.button_choose_file.setGeometry(QtCore.QRect(260,   20,   91,   41))
        self.button_choose_file.setObjectName("button_choose_file")
        self.label_filename        =        QtWidgets.QLabel(self.centralwidget)
        self.label_filename.setGeometry(QtCore.QRect(380,    30,    250,    16))
        self.label_filename.setObjectName("label_filename")
        self.button_reset       =      QtWidgets.QPushButton(self.centralwidget)
        self.button_reset.setGeometry(QtCore.QRect(360,     85,     111,     35))
        self.button_reset.setObjectName("button_reset")
        MainWindow.setCentralWidget(self.centralwidget)

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def                      retranslateUi(self,                      MainWindow):
        _translate             =             QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow",     "OS     Detect"))
        self.button_start_analysis.setText(_translate("MainWindow",    "Розпочати
виявлення                        \nопераційної                        системи"))
        self.radio_active.setText(_translate("MainWindow",   "Активний   режим"))
        self.button_exit.setText(_translate("MainWindow",  "Завершити  роботу"))
        self.radio_passive.setText(_translate("MainWindow",  "Пасивний  режим"))
        self.label_input_ip.setText(_translate("MainWindow", "Введіть IP адресу
вузла:"))
        self.button_choose_file.setText(_translate("MainWindow",   "Обрати   файл
\nдля                                                          аналізу"))
        self.label_filename.setText(_translate("MainWindow",           "None"))
        self.button_reset.setText(_translate("MainWindow",             "Очистити
вибір\nфайла"))
        self.radio_passive.setChecked(True)
```

file_extractor.py

```
import                  pyshark                  as                  ps


class                                                          file_extractor:
    def                                                        __init__(self):
        self.multiprocessing.set_start_method('spawn')
        self.ip_pr                              =                          False
        self.icmp_pr                             =                         False
        self.tcp_pr                              =                         False


        self.signature_array    =    ['ip_none',    'icmp_none',    'tcp_none']


def                          check_all(self,                          if_http):
    if                                                                  if_http:
        return  self.ip_pr  &  self.icmp_pr  &  self.tcp_pr  &  self.dns_pr  &
self.http_pr
    else:
```

```
        return self.ip_pr & self.icmp_pr & self.tcp_pr & self.dns_pr


def extract_from_file(self, filename, ip_source, http_yes):
    cap_file = ps.FileCapture('%s' % filename, display_filter='ip.src==%s' %
ip_source)
    for packet in cap_file:
        if not self.ip_pr:                    # ip layer
            if 'IP' in packet:
                try:
                    signature = packet['ip'].hdr_len + "," + packet['ip'].flags +
","  + packet['ip'].flags_df \
                        + "," + packet['ip'].ttl + "," +
packet['ip'].proto
                    self.signature_array[0] = signature
                    self.ip_pr = True
                    print('ip found')
                    if check_all(http_yes):
                        cap_file.close()
                        return self.signature_array
                except AttributeError:
                    pass
        if not self.icmp_pr:                  # icmp layer
            if 'ICMP' in packet:
                try:
                    signature = packet['icmp'].ident + "," +
packet['icmp'].seq_le

                    self.signature_array[1] = signature
                    self.icmp_pr = True
                    print('icmp found')
                    if check_all(http_yes):
                        cap_file.close()
                        return self.signature_array
                except AttributeError:
                    pass
        if not self.tcp_pr:                   # tcp layer
            if 'TCP' in packet:
                try:
                    signature = packet['tcp'].hdr_len + "," +
packet['tcp'].window_size_value + "," + \
                                packet['tcp'].window_size + "," +
packet['tcp'].window_size_scalefactor
                    self.signature_array[2] = signature
                    self.tcp_pr = True
                    print('tcp found')
                    if check_all(http_yes):
                        cap_file.close()
                        return self.signature_array
                except AttributeError:
                    pass
        if not self.dns_pr:                   # dns layer
            if 'DNS' in packet:
                try:
                    signature = packet['dns'].count_labels
                    self.signature_array[3] = signature
                    dns_pr = True
                    print('dns found')
                    if check_all(http_yes):
                        cap_file.close()
                        return self.signature_array
                except AttributeError:
                    pass
```

```
        if                              not                             self.http_pr:
            if                  'HTTP'                  in                  packet:
                if       'user_agent'       in       packet['http'].field_names:
                    try:
                        signature          =          packet['http'].user_agent
                        self.signature_array[4]           =            signature
                        self.http_pr                      =                  True
                        print('http                                      found')
                        if                                      check_all(http_yes):
                            cap_file.close()
                            return                              self.signature_array
                    except                                          AttributeError:
                        pass

    cap_file.close()
    return self.signature_array
```

main.py

```
    import                                                                  sys
from                    PyQt5                   import                  QtWidgets
from                    PyQt5.QtWidgets                 import                  *
import                                                              active_analysis
import                                                             passive_analysis
import                                                                      design
import                                                                      socket

sys._excepthook                         =                       sys.excepthook


def             exception_hook(exctype,             value,              traceback):
    print(exctype,                          value,                          traceback)
    sys._excepthook(exctype,                    value,                      traceback)
    sys.exit(1)


sys.excepthook                          =                       exception_hook


class           OS_Detect(QtWidgets.QMainWindow,            design.Ui_MainWindow):
    def                                                                 __init__(self):
        super().__init__()
        self.setupUi(self)
        self.button_choose_file.clicked.connect(self.browse_file)
        self.button_start_analysis.clicked.connect(self.start_analysis)
        self.button_exit.clicked.connect(self.exit_prog)
        self.button_reset.clicked.connect(self.reset_name)

    #                                   вибір                                   файлу
    def                                                             browse_file(self):
        filename = QtWidgets.QFileDialog.getOpenFileName(self, "Оберіть файл для
аналізу")
        if                                                                  filename:
            self.label_filename.setText(str(filename[0]))
    #                                   початок                                 виявлення
    def                                                             start_analysis(self):
        alert                               =                           QMessageBox()
        if                                      ip_check(self.text_ip_input.text()):
            if                                      self.radio_active.isChecked():
                if          self.label_filename.text()          ==          'None':
                    interface           =           'Беспроводная           сеть'
                    ip                  =                   self.text_ip_input.text()
                    aa              =               active_analysis.active_analysis
```

```
                    os,      probability      =      aa.active_analisys(self=aa,
interface_name=interface,                ip=ip,              bool_http_ua=False)
                    os,    probability    =    format_result(os,    probability)
                    alert.setWindowTitle("Результат")
                    alert.setText('ОС:   '   +   str(os)   +'\nВірогідність:   '   +
str(probability))
                    alert.exec()
                else:
                    alert.setWindowTitle("ПОМИЛКА")
                    alert.setText('Будь    ласка,    очистіть\nвибір    файлу')
                    alert.exec()
            if                                    self.radio_passive.isChecked():
                interface            =            'Беспроводная            сеть'
                ip                  =                 self.text_ip_input.text()
                filename               =              self.label_filename.text()
                if               filename                ==                'None':
                    pa                  =                passive_analysis.Passive
                    os,      probability      =      pa.passive_analisys(self=pa,
interface_name=interface,           ip_src=ip,           bool_time_limit=False,
                                                        bool_http_ua=False)
                    os,    probability    =    format_result(os,    probability)
                    alert.setWindowTitle("Результат")
                            alert.setWindowTitle('Результат сканування')
              alert.setText('Сімейство:                                  '+os[0]+'\t
Достовірність:'+probability[0]+'\nВерсія: ' + str(os[1]) + '\tДостовірність: ' +
str(
                  probability[1]) + '\n' + 'Звіт збережено у папці  D:/Program
Files/OS_Detect/reports/'+ip_addr       +       os[0]+'-'+os[1]       +'.csv')
alert.exec()
                else:
                    pa                  =                passive_analysis.Passive
                    os,    probability    =    pa.passive_file_analisys(self=pa,
file=filename,                ip_src=ip,               bool_time_limit=False,
                                                        bool_http_ua=False)
                    os,    probability    =    format_result(os,    probability)
                    alert.setWindowTitle("Результат")
                    alert.setText('Сімейство:                                  '+os[0]+'\t
Достовірність:'+probability[0]+'\nВерсія: ' + str(os[1]) + '\tДостовірність: ' +
str(
                  probability[1]) + '\n' + 'Звіт збережено у папці  D:/Program
Files/OS_Detect/reports/'+ip_addr       +       os[0]+'-'+os[1]       +'.csv')
alert.exec()
        else:
            alert.setWindowTitle("ПОМИЛКА")
            alert.setText('IP       адресу       введено       неправильно!')
            alert.exec()
    #                   вихід                   з                   програми
    @staticmethod
    def                                                          exit_prog():
        exit(0)

    def                                                      reset_name(self):
        self.label_filename.setText('None')


def                                                                  main():
    app                  =                  QtWidgets.QApplication(sys.argv)
    window                              =                            OS_Detect()
    window.show()
    app.exec()


def                      format_result(os,                      probability):
```

```
    texts = ['Linux 5.4.0', 'Mac OS X 10_12_4', 'MacOS 11.4', 'Windows 10 Corporate
20h2',
             'Windows  10  Home  20h2',  'Windows  7  Professional',  'Windows  XP
Professional                                                                SP3']
    i                                   =                                   os[0]
    r1                                =                                texts[int(i)]
    r2                          =                       str(probability[0][int(i)])
    return                                  r1,                                 r2


def                                                                 ip_check(ip):
    try:
        socket.inet_aton(ip)
    except                                                              socket.error:
        return                                                              False
    return                                                                   True


main()
```

passive_analysis.py

```
      import                traffic_sniffer                as                tf
import                                                                        sql
import                  file_extractor                  as                 fe


class                                                                   Passive:
    def                                                              __init__(self):
        self.texts = ['Linux 5.9.0', 'Mac OS X 10_12_4', 'MacOS 11.4', 'Windows
10                              Corporate                                   20h2',
         'Windows  10  Home  20h2',  'Windows  7  Professional',  'Windows  XP
Professional                                                                SP3']

    def     passive_file_analisys(self,     file,     ip_src,     bool_http_ua):
        signature_array = fe.extract_from_file(filename=file, ip_source=ip_src,
http_yes=bool_http_ua)
        search                              =                           sql.SQL_DB
        os,       proba       =       search.search_signature(signature_array)
        return                                os,                           proba

    def   passive_analisys(self,   interface_name,   ip_src,   bool_time_limit,
bool_http_ua):
        tf1                                 =                               tf.tf
        search                              =                           sql.SQL_DB
        signature_array     =     tf1.capture(tf1,     int_name=interface_name,
ip_source=ip_src,          limit=bool_time_limit,          http_yes=bool_http_ua)
        os,       proba       =       search.search_signature(signature_array)
        return os, proba
```

traffic_alanyzer.py

```
      import pyshark

ip_pr = False
icmp_pr = False
tcp_pr = False
dns_pr = False
http_pr = False
signature_array = ['ip_none', 'icmp_none', 'tcp_none', 'dns_none', 'http_none']


class tf:
```

```
    def __init__(self):
        self.ip_pr = False
        self.icmp_pr = False
        self.tcp_pr = False
        self.dns_pr = True
        self.http_pr = False
        self.signature_array = ['ip_none', 'icmp_none', 'tcp_none', 'dns_none',
'http_none']

    def check_all(self, if_http):
        if if_http:
            return ip_pr & icmp_pr & tcp_pr & dns_pr & http_pr
        else:
            return ip_pr & icmp_pr & tcp_pr & dns_pr

    def capture(self, int_name, ip_source, limit, http_yes):
        global ip_pr
        global icmp_pr
        global tcp_pr
        global dns_pr
        global http_pr
        global signature_array
        source = 'src ' + ip_source
        cap = pyshark.LiveCapture(interface=int_name, bpf_filter=source)
        if limit:
            cap.sniff(timeout=10)
            for packet in cap:
                if not ip_pr:  # ip layer
                    if 'IP' in packet:
                        try:
                            signature = packet['ip'].hdr_len + "," +
packet['ip'].flags + "," + packet['ip'].flags_df \
                                        + "," + packet['ip'].ttl + "," +
packet['ip'].proto
                            signature_array[0] = signature
                            ip_pr = True
                            print('ip found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                        except AttributeError:
                            pass
                if not icmp_pr:  # icmp layer
                    if 'ICMP' in packet:
                        try:
                            signature = packet['icmp'].ident + "," +
packet['icmp'].seq + "," + packet['icmp'].seq_le \
                                        + "," + packet['icmp'].data_data + "," +
packet['icmp'].data_len
                            signature_array[1] = signature
                            icmp_pr = True
                            print('icmp found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
```

```
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                            except AttributeError:
                                pass
                    if not tcp_pr:  # tcp layer
                        if 'TCP' in packet:
                            try:
                                signature = packet['tcp'].hdr_len + "," +
packet['tcp'].window_size_value + "," + \
                                            packet['tcp'].window_size + "," +
packet['tcp'].window_size_scalefactor
                                signature_array[2] = signature
                                tcp_pr = True
                                print('tcp found')
                                if self.check_all(http_yes):
                                    cap.close()
                                    path = "signature.txt"
                                    file = open(path, "w")
                                    k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                        + signature_array[3] + '*' +
signature_array[4]
                                    file.write(k)
                                    file.close()
                                    return signature_array
                            except AttributeError:
                                pass
                    if not dns_pr:  # dns layer
                        if 'DNS' in packet:
                            try:
                                signature = packet['dns'].count_labels
                                signature_array[3] = signature
                                dns_pr = True
                                print('dns found')
                                if self.check_all(http_yes):
                                    cap.close()
                                    path = "signature.txt"
                                    file = open(path, "w")
                                    k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                        + signature_array[3] + '*' +
signature_array[4]
                                    file.write(k)
                                    file.close()
                                    return signature_array
                            except AttributeError:
                                pass
                        elif 'MDNS' in packet:
                            try:
                                signature = packet['mdns'].count_labels
                                signature_array[3] = signature
                                dns_pr = True
                                print('dns found')
                                if self.check_all(http_yes):
                                    cap.close()
                                    path = "signature.txt"
                                    file = open(path, "w")
```

```
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                            except AttributeError:
                                pass
                    if not http_pr:
                        if 'HTTP' in packet:
                            if 'user_agent' in packet['http'].field_names:
                                try:
                                    signature = packet['http'].user_agent
                                    signature_array[4] = signature
                                    http_pr = True
                                    print('found http')
                                    if self.check_all(http_yes):
                                        cap.close()
                                        path = "signature.txt"
                                        file = open(path, "w")
                                        k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                            + signature_array[3] + '*' +
signature_array[4]
                                        file.write(k)
                                        file.close()
                                        return signature_array
                                except AttributeError:
                                    pass
            else:
                for packet in cap.sniff_continuously():
                    if not ip_pr:  # ip layer
                        if 'IP' in packet:
                            try:
                                signature = packet['ip'].hdr_len + "," +
packet['ip'].flags + "," + packet['ip'].flags_df \
                                            + "," + packet['ip'].ttl + "," +
packet['ip'].proto
                                signature_array[0] = signature
                                ip_pr = True
                                print('ip found')
                                if self.check_all(http_yes):
                                    cap.close()
                                    path = "signature.txt"
                                    file = open(path, "w")
                                    k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                        + signature_array[3] + '*' +
signature_array[4]
                                    file.write(k)
                                    file.close()
                                    return signature_array
                            except AttributeError:
                                pass
                    if not icmp_pr:  # icmp layer
                        if 'ICMP' in packet:
                            try:
                                signature = packet['icmp'].ident + "," +
packet['icmp'].seq + "," + packet['icmp'].seq_le \
                                            + "," + packet['icmp'].data_data + "," +
packet['icmp'].data_len
                                signature_array[1] = signature
```

```
                            icmp_pr = True
                            print('icmp found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                        except AttributeError:
                            pass
                if not tcp_pr:  # tcp layer
                    if 'TCP' in packet:
                        try:
                            signature = packet['tcp'].hdr_len + "," +
packet['tcp'].window_size_value + "," + \
                                        packet['tcp'].window_size + "," +
packet['tcp'].window_size_scalefactor
                            signature_array[2] = signature
                            tcp_pr = True
                            print('tcp found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                        except AttributeError:
                            pass
                if not dns_pr:  # dns layer
                    if 'DNS' in packet:
                        try:
                            signature = packet['dns'].count_labels
                            signature_array[3] = signature
                            dns_pr = True
                            print('dns found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                        except AttributeError:
                            pass
                    elif 'MDNS' in packet:
                        try:
                            signature = packet['mdns'].count_labels
                            signature_array[3] = signature
                            dns_pr = True
```

```
                            print('dns found')
                            if self.check_all(http_yes):
                                cap.close()
                                path = "signature.txt"
                                file = open(path, "w")
                                k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                    + signature_array[3] + '*' +
signature_array[4]
                                file.write(k)
                                file.close()
                                return signature_array
                        except AttributeError:
                            pass
                if not http_pr:
                    if 'HTTP' in packet:
                        if 'user_agent' in packet['http'].field_names:
                            try:
                                signature = packet['http'].user_agent
                                signature_array[4] = signature
                                http_pr = True
                                print('http found')
                                if self.check_all(http_yes):
                                    cap.close()
                                    path = "signature.txt"
                                    file = open(path, "w")
                                    k = signature_array[0] + '*' +
signature_array[1] + '*' + signature_array[2] + '*' \
                                        + signature_array[3] + '*' +
signature_array[4]
                                    file.write(k)
                                    file.close()
                                    return signature_array
                            except AttributeError:
                                pass
        path = "signature.txt"
        file =open(path,"w")
        k = signature_array[0] + '*' + signature_array[1] + '*' +
signature_array[2] + '*' \            + signature_array[3] + '*' +
signature_array[4]
        file.write(k)
        file.close()
        return signature_array
```

int_analysis.py

```
import pickle as pk
import pandas as pd
import numpy
from io import StringIO


def to_x(sign):
    names    =    ['ip_ttl',    'icmp_ident',    'icmp_seq_le',    'icmp_data_len',
'tcp_window_size_value',
             'tcp_window_size_scalefactor']
    str_data = StringIO(sign)
    data = pd.read_csv(str_data, names=names, sep=',')
    array = data.values
    x = array[:, 0:5]
    return x
```

```
def predict(signature):
    with open('model_DT_family.pkl', 'rb') as file:
        model = pk.load(file)
    x = to_x(signature)
    os = model.predict(x)
    proba = model.predict_proba(x)
    detailed_os, detailed_proba = predict_version(x, os)
    return detailed_os, detailed_proba


def predict_version(input_os, family):
    if family[0] == '0':  # macos
        with open('model_macos.pkl', 'rb') as file:
            model = pk.load(file)
    elif family[0] == '1':  # unix
        with open('model_linux.pkl', 'rb') as file:
            model = pk.load(file)
    elif family[0] == '3':  # cisco
        with open('model_linux.pkl', 'rb') as file:
            model = pk.load(file)
    elif family[0] == '4':  # juniper
        with open('model_linux.pkl', 'rb') as file:
        model = pk.load(file)
    elif family[0] == '5':  # generic routers
        with open('model_linux.pkl', 'rb') as file:
        model = pk.load(file)
    else:
        with open('model_windows.pkl', 'rb') as file:  # 2 == windows
            model = pk.load(file)

    os = model.predict(input_os)
    proba = model.predict_proba(input_os)
    proba = convert_proba(proba, os)
    os = convert_os(os, family)
    return os, proba


def convert_proba(proba, os):
    i = os[0]
    a = proba.tolist()
    r2 = a[0][int(i)]
    return r2
```

nvd_query.py

```
import nvdlib

nvd():
    result = ''
    r = nvdlib.searchCVE(keywordSearch='Windows 10')
    for i in (0,10);
        result += r.CVE +';'

    return result
```

ІЛЮСТРАТИВНА ЧАСТИНА

АРХІТЕКТУРА ПРОГРАМНОГО ЗАСОБУ

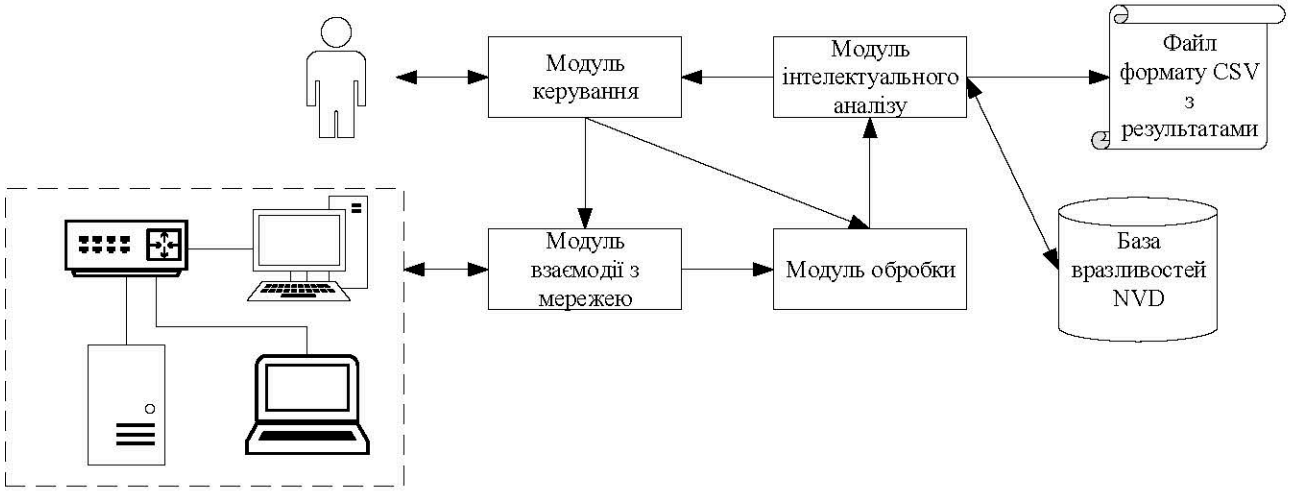
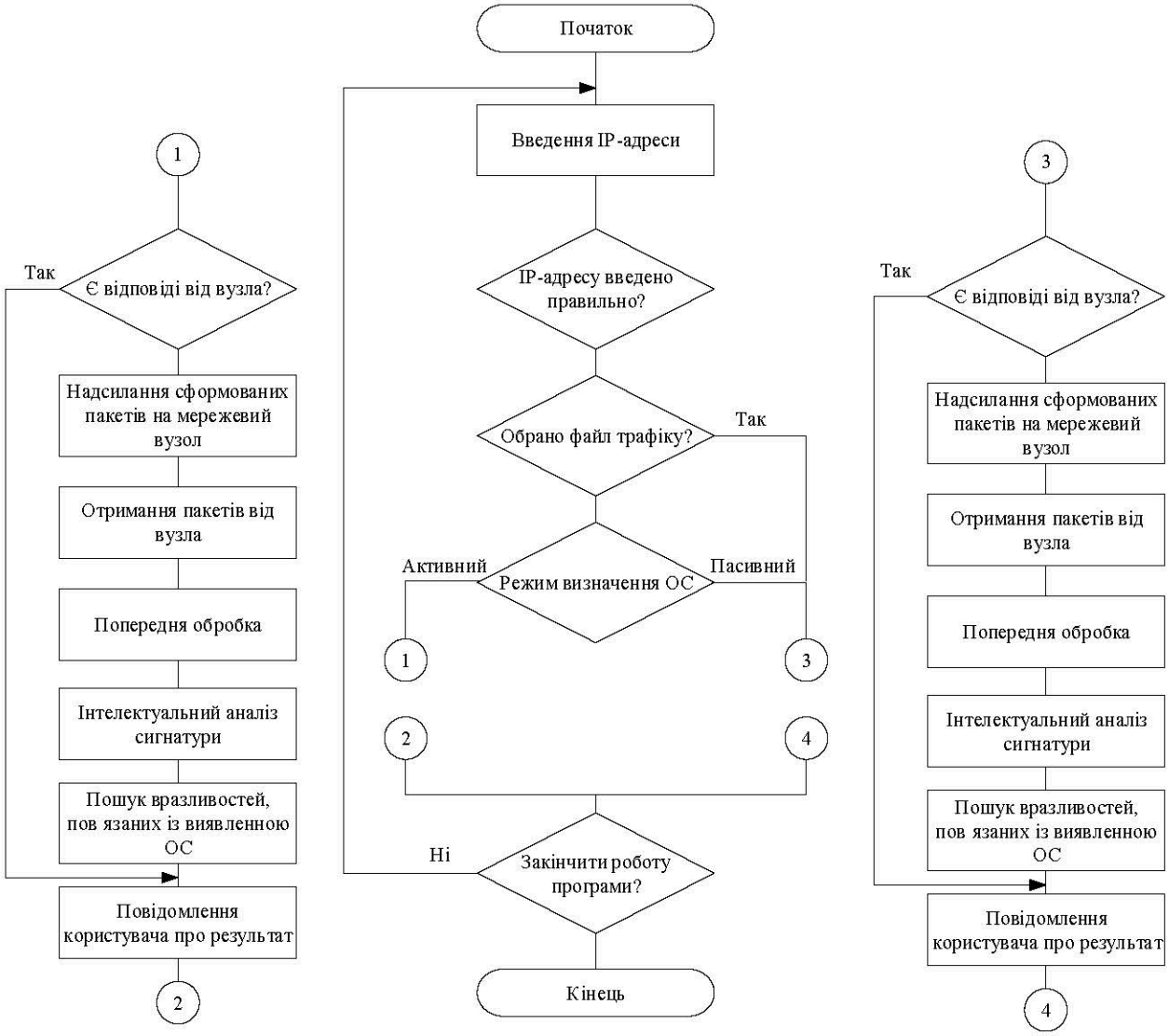
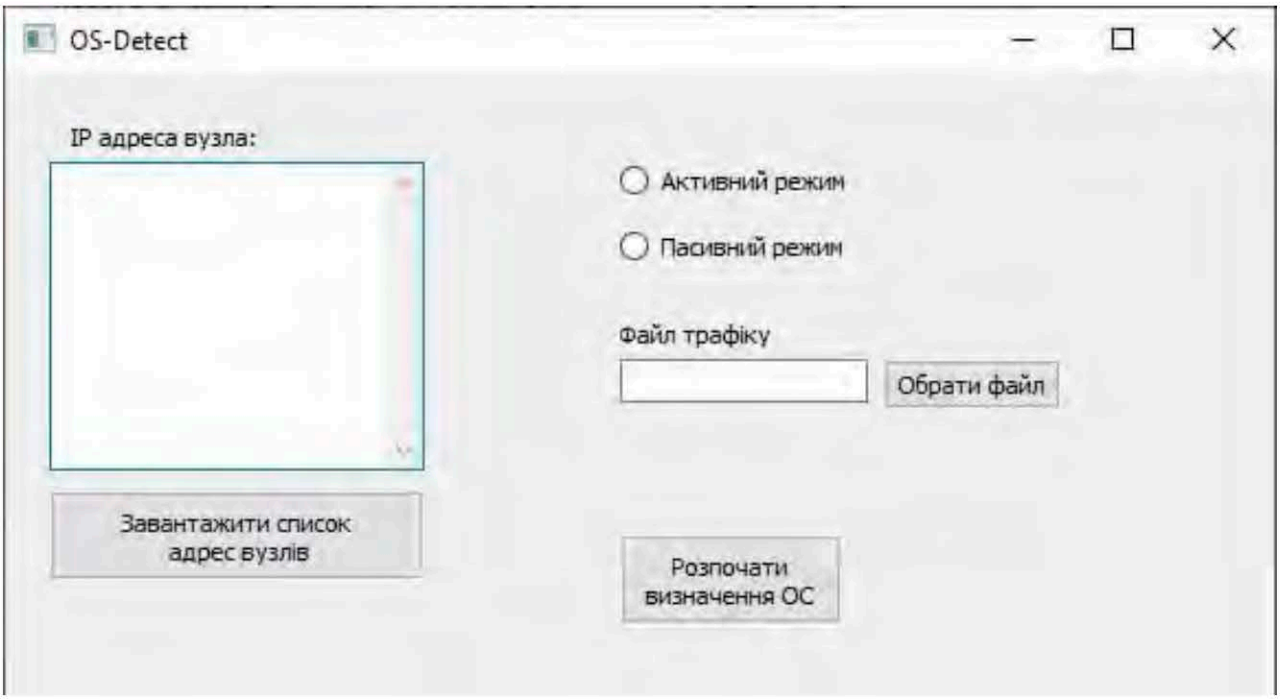


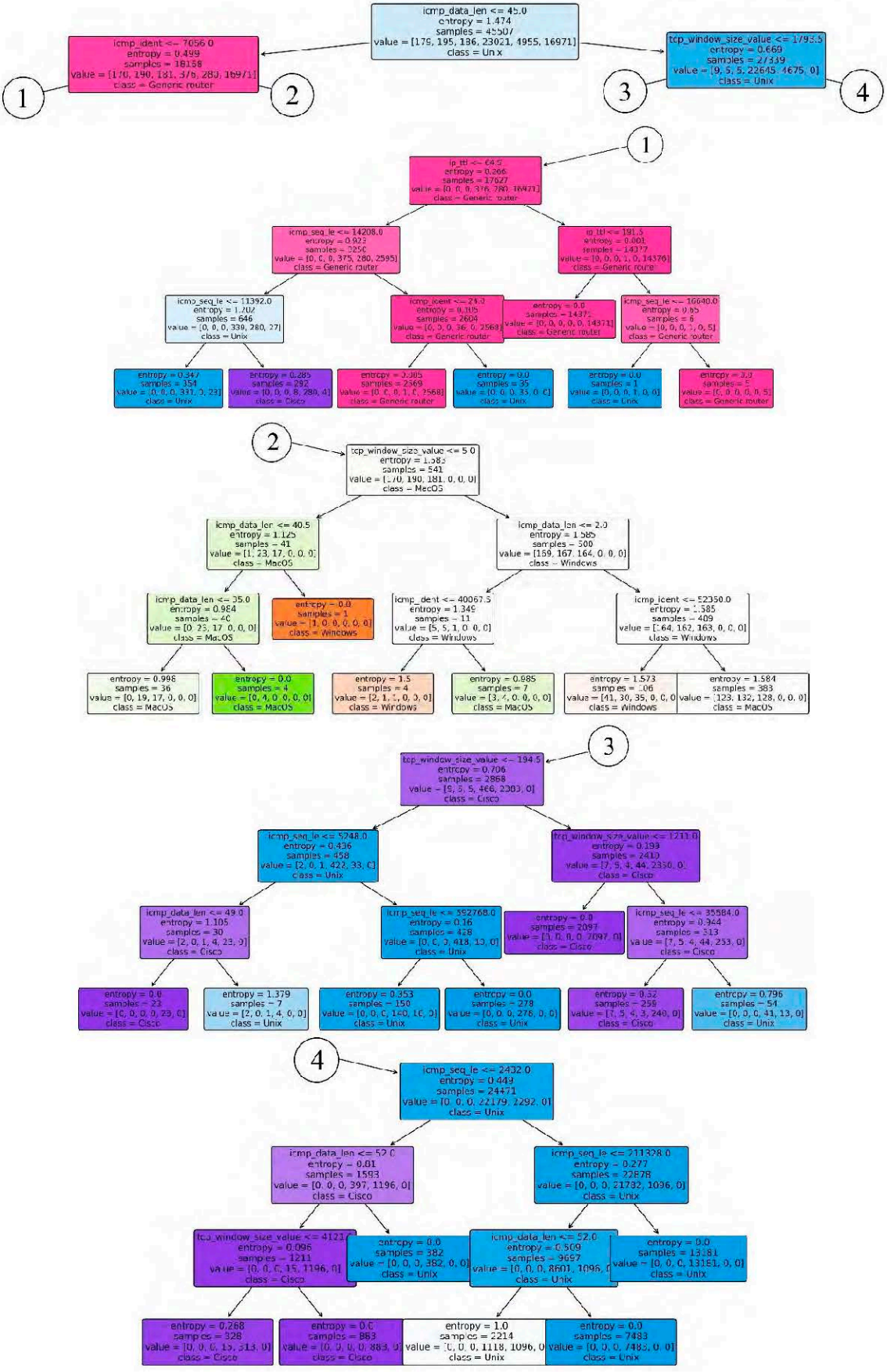
СХЕМА АЛГОРИТМУ РОБОТИ ПРОГРАМНОГО ЗАСОБУ



ІНТЕРФЕЙС ПРОГРАМНОГО ЗАСОБУ



СТРУКТУРА КЛАСИФІКАТОРА СІМЕЙСТВ ОПЕРАЦІЙНИХ СИСТЕМ



СТРУКТУРА АНСАМБЛЕВОЇ МОДЕЛІ

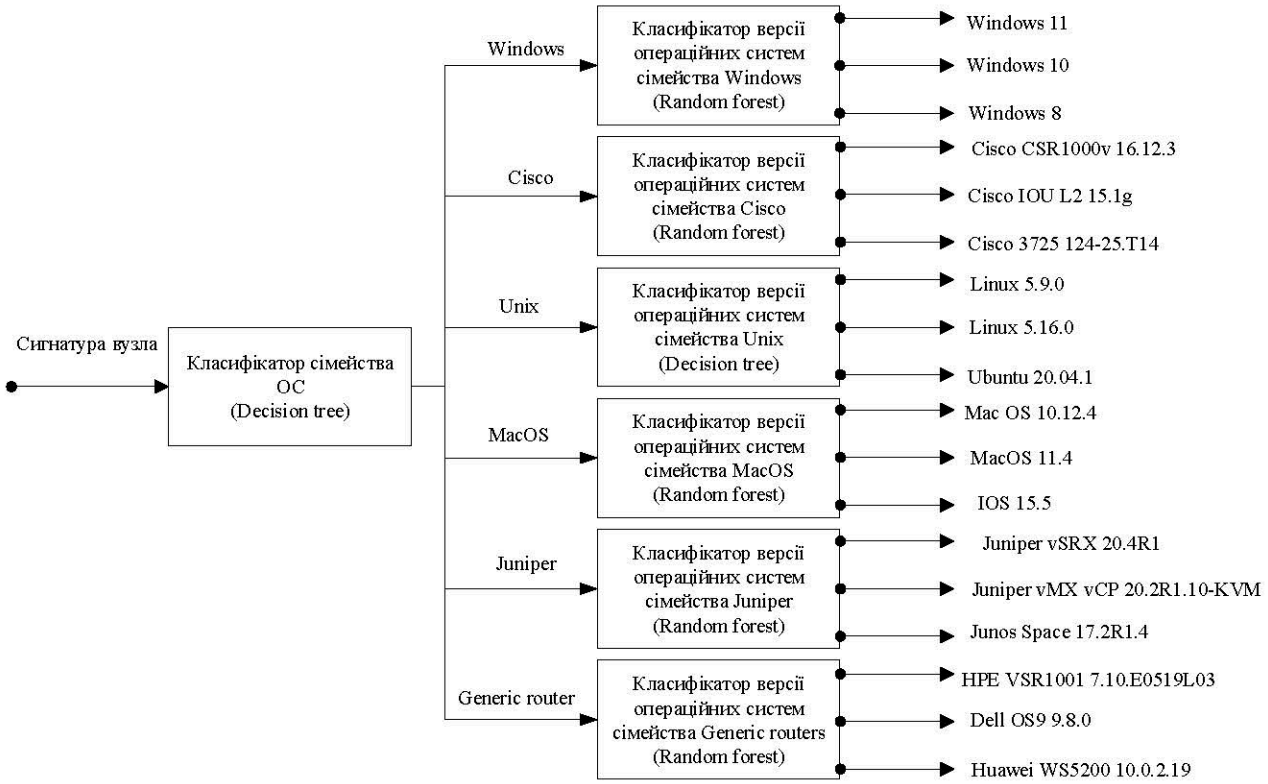


СХЕМА ПРОЦЕСУ ІНТЕЛЕКТУАЛЬНОГО АНАЛІЗУ