Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя
(повне найменування вищого навчального закладу)
Факультет комп'ютерно-інформаційних систем і програмної інженерії
(назва факультету)
Кафедра кібербезпеки
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(освітній рівень)

на тему: " Інтелектуальні методи виявлення шкідливого програмного забезпечення методами машинного навчання в Android "

Виконав: студент (ка)

Спеціальності:
125 «Кібербезпека та захист інформації»
(шифр і назва напряму підготовки, спеціальності)
Луговський Павло Володимирович
підпис (прізвище та ініціали)

Керівник Козак Р.О.
підпис (прізвище та ініціали)

Нормоконтроль Тимощук Д. І.
підпис (прізвище та ініціали)

Завідувач кафедри Загородна Н.В.
підпис (прізвище та ініціали)

Рецензент Марценко С.В.
підпис (прізвище та ініціали)

м. Тернопіль – 2024

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«__» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека та захист інформації
(шифр і назва спеціальності)

Студенту Луговському Павлу Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи " Інтелектуальні методи виявлення шкідливого програмного забезпечення методами машинного навчання в Android

Керівник роботи Козак Руслан Орестович, к.т.н., доцент
доцент кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 22 » 11 2024 року № 4/7-1119

2. Термін подання студентом завершеної роботи 20.12.2024

3. Вихідні дані до роботи

Літературні джерела, відкритий набір даних CICMalDroid 2020

4. Зміст роботи (перелік питань, які потрібно розробити)

Теоретичні основи виявлення шкідливого програмного забезпечення

Основні підходи до виявлення шкідливого пз

Практична реалізація методів машинного навчання для виявлення шкідливого ПЗ для ANDROID

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)

Мета, задачі, Основні типи шкідливого ПЗ для мобільних пристроїв

Підходи до виявлення шкідливого ПЗ

Основні етапи статичного аналізу, Типові статичні ознаки для виявлення шкідливого ПЗ

Основні етапи динамічного аналізу Типові динамічні ознаки для виявлення шкідливого ПЗ

Набір даних CICMalDroid 2020, Переваги Python, Теплова карта (heatmap)

Результати тестування моделей для різної кількості ознак

Візуалізація метрик точності для різних моделей

Висновки

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Охорона праці	Осухівська Г.М., к.т.н., доцент		
Безпека в надзвичайних ситуаціях	Клепчик В.М., проректор з адміністративно-господарської роботи та будівництва		

7. Дата видачі завдання 23.09.2024 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	23.09 – 25.09	*Виконано*
2.	Підбір наукових джерел про шкідливе ПЗ та способи його виявлення	26.09-13.10	*Виконано*
3.	Переклад та опрацювання наукових джерел про теоретичні підходи виявлення шкідливого ПЗ	14.10-25.10	*Виконано*
4.	Виконання дослідження щодо пошуку шляхів підвищення ефективності виявлення шкідливого ПЗ	26.10-10.11	*Виконано*
5.	Виконання експериментальних досліджень	11.11-20.11	*Виконано*
6.	Оформлення розділу «Теоретичні основи виявлення шкідливого програмного забезпечення»	21.11-25.11	*Виконано*
7.	Оформлення розділу «Основні підходи до виявлення ПЗ»	26.11-30.11	*Виконано*
8.	Оформлення розділу «Практична реалізація методів машинного навчання для виявлення шкідливого ПЗ»	01.12-05.12	*Виконано*
9.	Виконання завдання до підрозділу «Охорона праці»	04.12-08.12	*Виконано*
10.	Виконання завдання до підрозділу «Безпека в надзвичай них ситуаціях»	04.12-08.12	*Виконано*
11.	Оформлення кваліфікаційної роботи	01.12-08.12	*Виконано*
12.	Нормоконтроль	09.12 – 11.12	*Виконано*
13.	Перевірка на плагіат	12.12 – 15.12	*Виконано*
14.	Попередній захист кваліфікаційної роботи	16.12 – 20.12	*Виконано*
15.	Захист кваліфікаційної роботи	23.12.2024	

Студент _____ Луговський П.В.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Козак Р.О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Інтелектуальні методи виявлення шкідливого програмного забезпечення методами машинного навчання в Android // ОР «Магістр» // Луговський Павло Володимирович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2024 // С. 71, рис. – 18, табл. – 4 , кресл. – __, додат. – 1.

Ключові слова: ШКІДЛИВЕ ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, ANDROID, СТАСТИЧНІ, ДИНАМІЧНІ МЕТОДИ, СИГНАТУРА, АНОМАЛІЯ.

Кваліфікаційна робота присвячена оцінці ефективності методів машинного навчання в задачах виявлення шкідливого програмного забезпечення в Android.

В першому розділі кваліфікаційної роботи здійснено огляд типів та способів поширення шкідливого програмного забезпечення у мобільних додатках, наведено реальні кейси такого програмного забезпечення та рекомендації щодо захисту.

В другому розділі кваліфікаційної роботи наведено класифікацію методів виявлення шкідливого ПЗ інтелектуальними методами, висвітлено особливості статичного, динамічного та гібридного підходу та описано моделі машинного навчання для виявлення аномалій, як одного зі способів ідентицікації шкідливо ПЗ.

В третьому розділі описано деталі практичної реалізації методів машинного навчання для виявлення шкідливого програмного забезпечення, проведено оцінку точності побудованих моделей.

ABSTRACT

Intelligent Methods for Detecting Malware Using Machine Learning in Android Systems // Thesis of educational level "Master"// Pavlo Luhovskyi // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cybersecurity, group СБм-61 // Ternopil, 2024// P. 71 figs. 18, tables 4 , drawings __ added. – __.

Keywords: MALWARE, ANDROID, STATIC, DYNAMIC METHODS, SIGNATURE, ANOMALY

The qualification paper is devoted to the evaluation of the effectiveness of machine learning methods in detecting malware in Android.

The first section of the thesis provides an overview of the types and methods of malware spreading in mobile applications, real cases of such software and recommendations for protection.

The second section of the qualification work describes a classification of methods for detecting malware using intelligent methods, highlights the features of static, dynamic and hybrid approaches, and describes machine learning models for detecting anomalies as one of the ways to identify malware.

The third section describes the details of the practical implementation of machine learning methods for detecting malware, and evaluates the accuracy of the built models.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

ML	—	Machine Learning
kNN	—	k Nearest Neighbors
SVM	—	Support Vector Machine
ПЗ	—	Програмне забезпечення
IT	—	Inforamtion Technologies
PCA	—	Principal Component Analysis
ОС	—	Операційна система
VPN	—	Virtual private network
API	—	Application Programming Interface
TLS	—	Transport Layer Security
ICC	—	Inter-Component Communication
APK	—	Android Package Kit
RNN	—	Recurrent Neural Network
LOF	—	Local Outlier Factor
SGD	—	Stochastic Gradient Descent
DoS	—	Denial of Service

ВСТУП

Актуальність теми. Актуальність дослідження теми виявлення шкідливого програмного забезпечення для мобільних платформ, зокрема Android, зумовлена стрімким зростанням кількості мобільних пристроїв і додатків, що використовуються у повсякденному житті. За статистикою Statcounter [1], станом на 2024 рік понад 70% мобільних пристроїв у світі працюють на Android, що робить цю платформу найпопулярнішою та водночас найбільш привабливою ціллю для кіберзлочинців.

Шкідливе програмне забезпечення для Android постійно еволюціонує, використовуючи все більш складні методи, такі як експлойти у системних компонентах, соціальна інженерія для поширення фальшивих додатків або приховані механізми монетизації через крадіжку персональних даних. Зокрема, у 2023 році було виявлено серйозну хвилю атак через Play Store, де шкідливі програми завантажили мільйони користувачів до їх виявлення. Це свідчить про недостатню ефективність традиційних методів перевірки, які не встигають за швидкістю поширення нових загроз.

Традиційні системи виявлення шкідливого ПЗ, що базуються на сигнатурах і правилах, часто виявляються неефективними для сучасних атак. Вони не здатні оперативно реагувати на нові загрози, зокрема програми, що використовують поліморфізм або самозашифровування для обходу стандартних механізмів захисту. Це створює необхідність впровадження більш інтелектуальних систем захисту, здатних до виявлення шкідливих програм на основі поведінкових характеристик та аналізу великих обсягів даних. Машинне навчання, яке демонструє високу ефективність у виявленні складних патернів, є перспективним рішенням для подолання цих викликів.

Тема роботи набуває ще більшої значущості в умовах постійного зростання кількості атак на мобільні фінансові додатки, системи електронної комерції та особисті акаунти користувачів. Наприклад, лише у першій половині 2023 року кількість атак на мобільні банківські додатки зросла на 50%, порівняно з попереднім роком. Це підкреслює потребу у створенні надійних методів

виявлення, які можуть ефективно захистити дані користувачів і зменшити ризик фінансових втрат. Отже, розробка інтелектуальних систем виявлення шкідливого ПЗ на основі методів машинного навчання є не лише актуальним, а й критично важливим завданням для забезпечення кібербезпеки у сучасному світі.

Мета кваліфікаційної роботи. Розробка і дослідження ефективних методів виявлення шкідливого програмного забезпечення для Android-середовища з використанням інтелектуальних методів машинного навчання.

Досягнення мети передбачало виконання наступних *завдань*:

- Провести огляд проблеми виявлення шкідливого ПЗ, типів та реальних кейсів шкідливого ПЗ для мобільних додатків.
- Проаналізувати сучасні підходи до виявлення шкідливого ПЗ за допомогою машинного навчання.
- Знайти та проаналізувати доступні набори даних для навчання моделей виявлення шкідливого програмного забезпечення (ПЗ).
- Дослідити ефективність різних алгоритмів машинного навчання для задачі виявлення шкідливого програмного ПЗ
- Провести експериментальну оцінку вибраних моделей машинного навчання на тестових даних та порівняти їх ефективність за ключовими метриками, такими як точність, повнота, F1-міра.
- Надати рекомендації щодо впровадження та вдосконалення запропонованого рішення.

Об’єкт дослідження: процеси виявлення та попередження шкідливого програмного забезпечення в Android-середовищі.

Предмет дослідження: методи інтелектуального аналізу для підвищення ефективності виявлення шкідливого ПЗ.

Наукова новизна одержаних результатів кваліфікаційної роботи полягає в проведенні порівняльного аналізу ефективності різних моделей машинного навчання для задачі виявлення шкідливого програмного забезпечення в Android з врахуванням специфіки мобільної платформи та дослідженні впливу кількості ознак на точність моделей.

Практичне значення одержаних результатів полягає в створенні прототипу системи виявлення шкідливого ПЗ, яка може бути інтегрована у мобільні додатки або системи захисту даних. Надані рекомендацій щодо застосування методів машинного навчання в реальних умовах для виявлення шкідливих програм у Android-середовищі, які можуть бути використані для захисту мобільних пристроїв користувачів від нових видів кіберзагроз.

Отримані результати можуть бути використані в освітньому процесі для підготовки спеціалістів у галузі кібербезпеки та машинного навчання, а також слугувати основою для подальших наукових досліджень у цій сфері.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на XII науковій-технічній конференції «Інформаційні моделі, системи та технології» (м.Тернопіль, Україна).

Публікації. Окремі результати кваліфікаційної роботи опубліковано у працях конференції (див. Додаток А).

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

За даними компанії Statista [2], у 2023 році приблизно вісім із десяти атак зловмисного програмного забезпечення були спрямовані на мобільні пристрої по всьому світу (див. рис.1). Ця цифра зростає з кожним роком. Ці загрози завдають серйозних фінансових збитків як приватним користувачам, так і компаніям, які використовують мобільні пристрої у своїй діяльності.

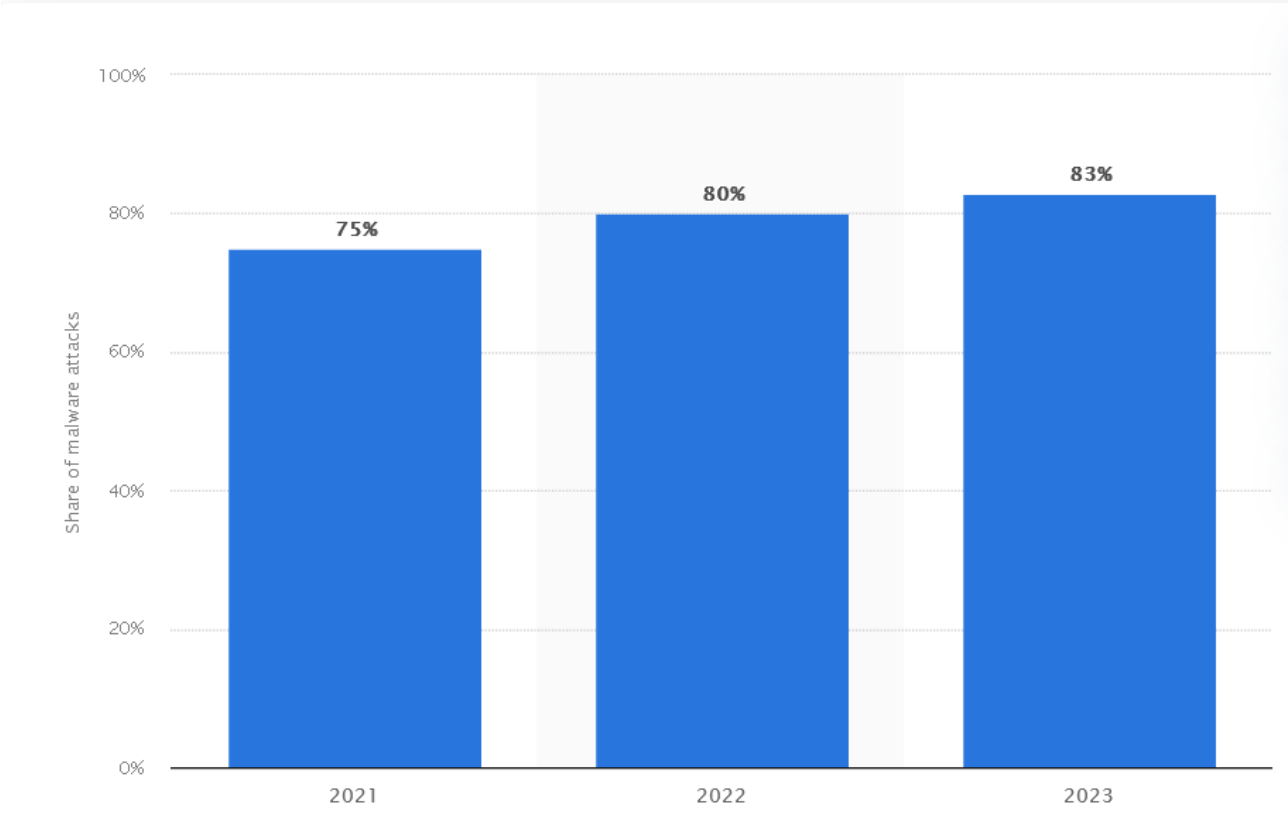

Рисунок 1.1 – Відсоток шкідливого програмного забезпечення на мобільні пристрої

Виявлення шкідливого програмного забезпечення у мобільних додатках є критично важливим через стрімке зростання ролі мобільних пристроїв у житті людей та бізнесі. Мобільні додатки широко використовуються для фінансових операцій, комунікації, зберігання персональних даних та доступу до корпоративних ресурсів. Шкідливі програми можуть викрадати конфіденційну інформацію, завдавати фінансових збитків або створювати умови для подальших атак, таких як фішинг чи розсилання спаму. Наприклад, атаки на банківські

додатки можуть призводити до прямих втрат коштів користувачів, а зараження корпоративних мобільних пристроїв може спричинити витік комерційної таємниці. У зв’язку з цим забезпечення безпеки мобільних додатків є пріоритетним завданням у сфері кіберзахисту.

Окрім фінансових і репутаційних ризиків, шкідливі програми також становлять загрозу для загальної стабільності мобільних екосистем. Велика кількість заражених пристроїв створює додаткове навантаження на сервери, послаблює довіру користувачів до мобільних платформ і гальмує розвиток цифрових сервісів. Виявлення шкідливого ПЗ на ранніх етапах його поширення дозволяє запобігти значним втратам і забезпечити стабільність роботи мобільних пристроїв. У поєднанні з сучасними технологіями, такими як машинне навчання, це дозволяє підвищити ефективність захисту, виявляючи навіть нові види загроз, які не можуть бути розпізнані традиційними методами.

1.1 Основні типи шкідливого програмного забезпечення в мобільних додатках

Огляд шкідливого програмного забезпечення (ПЗ) в Android-середовищі є важливим для розуміння, як саме мобільні пристрої стають вразливими до атак і які типи загроз існують.

Android через свою відкриту архітектуру та значну частку ринку став основною мішенню для кіберзлочинців, які створюють різноманітні типи шкідливих програм. Шкідливе ПЗ створюється для отримання несанкціонованого доступу до конфіденційної інформації користувачів, крадіжки коштів, викрадення персональних даних, а також для розповсюдження фальшивих додатків, які можуть спричинити значні фінансові втрати та інші небезпечні наслідки.

Однією з причин популярності цієї платформи серед зловмисників є легкість завантаження додатків із неофіційних джерел, що підвищує ймовірність зараження пристроїв.

Шкідливі програми для Android можна класифікувати за їх функціональністю:

- троянські програми (Trojans);
- шкідливі програми, що викрадають дані (Data Stealers);
- програми-вимагачі (Ransomware);
- рекламне ПЗ (Adware);
- шпигунське ПЗ (Spyware);
- фейкові додатки преміум-контенту (Fake Apps);
- мобільні боти (Botnets);
- руткити (Rootkits).

Основні типи шкідливого програмного забезпечення для мобільних платформ наведені на рисунку 1.2 [3].

Рисунок 1.2 - Основні типи шкідливого програмного забезпечення для мобільних платформ

Найпоширенішими типами є трояни, які маскуються під легітимні додатки, але виконують небезпечні дії. До них належать:

- Банківські трояни, що викрадають фінансову інформацію, таку як номери кредитних карт, паролі для онлайн-банкінгу.
- SMS-трояни, які автоматично відправляють платні SMS-повідомлення.

Data Stealers можуть записувати всі натискання клавіш, щоб отримати паролі та іншу конфіденційну інформацію (keyloggers), робити знімки екрана (скріншоти), щоб отримати доступ до введених даних або ж знімки з камери пристрою, щоб отримати доступ до особистих даних.

Іншим популярним типом є рекламне ПЗ (adware), яке нав’язливо показує рекламу, що приносить прибуток зловмисникам і може значно знизити продуктивність пристрою.

Також існують шпигунські програми (spyware), які збирають інформацію про користувача. Цей тип шкідливого ПЗ може передавати зловмисникам такі дані, як геолокація, історія дзвінків, повідомлення та навіть записи розмов. Шпигунське ПЗ часто використовується для стеження за користувачами та викрадення конфіденційної інформації, що може призвести до порушення конфіденційності та безпеки особистих даних.

Значну загрозу становлять програми-вимагачі (ransomware), які блокують пристрій або шифрують дані, вимагаючи викуп за їх розблокування. Це тип програмного забезпечення, який шифрує файли або блокує пристрій користувача, вимагаючи викуп за відновлення доступу. Випадки атак програм-вимагачів почастішали через використання соціальної інженерії та фішингових атак, які спрямовані на обман користувачів для завантаження шкідливих додатків. Впровадження програм-вимагачів стає дедалі складнішим, оскільки вони використовують новітні методи шифрування, що ускладнює відновлення даних навіть після видалення шкідливого ПЗ. В останні роки ці типи шкідливого ПЗ набули особливої популярності серед кіберзлочинців через їх високу прибутковість.

Також слід зазначити існування ботнетів (botnets) – шкідливих програм, які об'єднують пристрої у мережу для виконання атак на інші системи або для використання в розподілених атаках типу DDoS.

Руткит (rootkit) – це злісний програмний код, який здатний приховати свою присутність в операційній системі та надавати зловмиснику повний контроль над нею. Іншими словами, це невидимий для звичайних засобів захисту "троянський кінь", який відкриває задні двері в вашу систему. Руткити – це одні з найнебезпечніших видів шкідливого програмного забезпечення. Вони здатні завдати серйозної шкоди як окремим користувачам, так і цілим організаціям.

Важливим аспектом захисту від шкідливого ПЗ є розуміння типів атак, що здійснюються з його використанням. Наприклад, атаки на фінансові додатки

спрямовані на викрадення облікових даних або перехоплення транзакцій. Інші атаки можуть включати віддалене управління пристроєм, яке дозволяє зловмисникам переглядати особисті дані, фотографії та інші файли. Усі ці типи загроз вимагають постійного вдосконалення систем захисту та адаптації антивірусного програмного забезпечення.

1.2 Приклади кейсів шкідливого ПЗ на Android

Операційна система Android розділена на чотири рівні, як показано на малюнку 1, ядро Linux є нижнім рівнем, відповідальним за абстракцію апаратного забезпечення пристрою. Рівень бібліотек містить набір бібліотек, включаючи WebKit, Libc і SSl. Бібліотеки Android включають бібліотеки на основі Java, такі як android view і android widget. Рівень інфраструктури програми надає послуги вищого рівня програми з точки зору класів Java. Верхній рівень називається прикладним рівнем, де додатки написані для встановлення (рисунок 1.3).

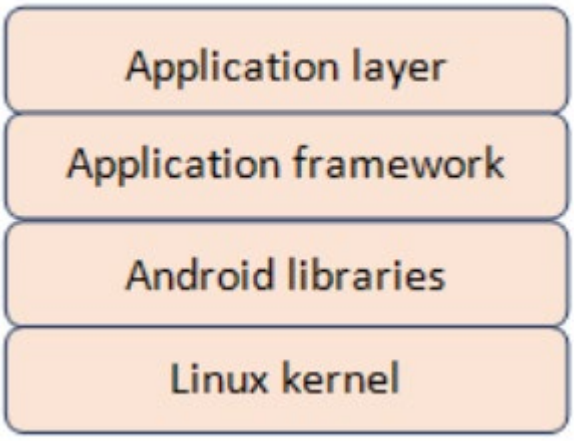

Рисунок 1.3 – Архітектура Android

Багато користувачів вважають, що оскільки ОС Android побудова на ядрі Linux, то це дає йому перевагу і робить стійким до кібератак. Від першого в історії вірусу зловмисне програмне забезпечення для мобільних пристроїв значно розвинулося. Компанія F-Secure [4] повідомила, що перший вірус, названий трояном для Palm, був представлений у 2000 році. Натомість поява

першого Android вірусу датується 2010 роком [5]. Проте, слід відзначити, що еволюція зловмисного програмного забезпечення Android відбувається прискореними темпами. В таблиці 1.1 наведено окремі приклади шкідливого ПЗ та приклади реальних кейсів та сімейств ПЗ лише для операційної системи Android з 2016 по 2020 [6,7]:

Таблиця 1.1 – Приклади шкідливого ПЗ для Android:

Рік	Назва	Тип
2016	Xbot	Ransomware
2016	Gazon	Virus
2016	Godless	Root Exploit
2017	Bad Rabbit	Ransomware
2017	Judy	Adware
2017	GantSpy	Trojan
2017	ToastAmigo	BackDoor
2018	RedDrop	Spyware
2018	Chamois	BackDoor
2019	Cerberus	Trojan
2019	TimpDoor	Spyware
2019	XHelper	Trojan
2020	Ghimob	Trojan

Кількість шкідливого програмного забезпечення для Android постійно зростає, оскільки зловмисники розробляють все нові та складніші способи зараження пристроїв. Розглянемо детальніше способи поширення шкідливого ПЗ на Android.

1.3 Способи поширення шкідливого ПЗ на Android

Серед методів поширення шкідливого ПЗ для Android найбільш популярними є соціальна інженерія та фішингові атаки. Соціальна інженерія –

це тактика, коли зловмисники переконують користувача завантажити шкідливий додаток, маскуючи його під відому програму або гру. Часто такі програми мають додаткові дозволи, які не відповідають їх заявленій функціональності, що дозволяє зловмисникам отримувати доступ до критичних функцій пристрою.

Фішингові атаки. використовують обманні методи для переконання користувачів у завантаженні та встановленні шкідливих додатків. Зловмисники також можуть використовувати фальшиві оновлення для відомих додатків або системних компонентів, щоб ввести користувачів в оману та змусити їх завантажити шкідливий код. Це підкреслює важливість регулярного оновлення систем безпеки і використання офіційних джерел для завантаження додатків.

Нарешті, варто зазначити багатостадійні атаки, коли шкідливе ПЗ завантажує інші компоненти після початкового встановлення. Такі атаки часто використовуються для встановлення бекдорів або ботнетів, які можуть контролюватися віддалено. Зловмисники отримують можливість зламувати пристрої, використовувати їх для проведення DDoS-атак або викрадати дані на вимогу. Така архітектура атак ускладнює їх виявлення, адже вони можуть тривалий час залишатися в системі без видимих ознак активності. Це підтверджує необхідність постійного моніторингу та використання інтелектуальних методів виявлення шкідливого ПЗ.

Крім того, деякі шкідливі програми використовують уразливості операційної системи Android для ескалації привілеїв, що дозволяє їм діяти з правами адміністратора. Це ускладнює їх видалення та дозволяє їм виконувати широкий спектр небезпечних дій, включаючи зміну системних файлів, перехоплення даних і встановлення додаткового шкідливого ПЗ без відома користувача.

Атакуючі також використовують методи стеганографії для приховування шкідливих компонентів у файлах, таких як зображення або документи. Це дозволяє шкідливому ПЗ уникати виявлення антивірусними програмами, які сканують файли на наявність відомих сигнатур. Окрім цього, зловмисники використовують поліморфізм і шифрування для створення динамічних і

важковловимих версій своїх програм, що робить їх непомітними для традиційних систем виявлення.

1.4 Базові методи захисту від шкідливого ПЗ на мобільних платформах

Хоча світ кіберзагроз постійно еволюціонує, існують перевірені часом методи, які допоможуть захистити ваш мобільний пристрій від шкідливого програмного забезпечення.

1.4.1 Використання офіційних магазинів додатків

Офіційні магазини додатків, такі як Google Play та App Store, мають строгі правила для розробників. Перед публікацією додаток проходить перевірку на наявність шкідливого коду. Це значно знижує ризик завантажити шкідливу програму.

Кожен додаток, який ви бачите в магазині, пройшов певну модерацію. Розробники повинні надавати детальну інформацію про свій додаток, а користувачі можуть залишати відгуки та оцінки, що допомагає іншим користувачам приймати обґрунтовані рішення.

Крім безпеки, офіційні магазини пропонують зручний пошук, автоматичні оновлення та систему відшкодування коштів у разі проблем з додатком.

1.4.2 Регулярні оновлення операційної системи та додатків

Зловмисники постійно шукають нові вразливості в операційних системах та додатках. Оновлення виправляють ці вразливості, роблячи ваш пристрій більш безпечним.

Розробники операційних систем та додатків постійно працюють над покращенням безпеки своїх продуктів. Коли ви встановлюєте оновлення, ви отримуєте найновіші виправлення безпеки та нові функції.

Автоматичне встановлення оновлень гарантує, що ваш пристрій завжди буде захищений. Не відкладайте встановлення оновлень, оскільки це може зробити ваш пристрій вразливим до атак.

1.4.3 Обережність при встановленні додатків та відкритті посилань

Не всі додатки в магазинах є безпечними. Деякі можуть містити шкідливий код або збирати вашу персональну інформацію без вашої згоди.

Перед встановленням додатку ознайомтеся з його описом, відгуками користувачів та дозволами, які він запитує. Якщо щось здається підозрілим, краще відмовитися від встановлення.

Не потрібно встановлювати додатки з невідомих джерел, навіть якщо вони здаються безкоштовними та цікавими.

Шкідливе програмне забезпечення часто поширюється через фішингові електронні листи, SMS-повідомлення та соціальні мережі. Відкриття підозрілих посилань або завантаження файлів з невідомих джерел може призвести до зараження вашого пристрою.

Не потрібно відкривати посилання від невідомих відправників, навіть якщо вони виглядають довірливо. Потрібно перевіряти адреси веб-сайтів перед введенням особистої інформації та уникати завантаження файлів з торрентів або інших піратських сайтів.

Розширення для браузера, які блокують рекламу та шкідливі сайти, можуть додатково захистити від онлайн-загроз.

1.4.4 Створення резервних копій даних

Створення резервних копій даних мобільних додатків є важливим заходом для захисту вашої інформації. Це дозволяє відновити дані у випадку втрати або пошкодження пристрою, а також при перевстановленні додатків.

Більшість сучасних мобільних операційних систем (Android, iOS) пропонують вбудовані функції для автоматичного створення резервних копій даних. Ці копії зазвичай зберігаються в хмарному сховищі, пов’язаному з вашим обліковим записом (Google Drive для Android, iCloud для iOS). Принцип роботи простий: система періодично сканує ваш пристрій та синхронізує обрані дані з хмарним сховищем.

Залежно від операційної системи та налаштувань, ви можете створювати резервні копії різних типів даних: контакти, календар, фотографії, відео,

повідомлення, налаштування додатків тощо. Деякі додатки також пропонують власні функції резервного копіювання, дозволяючи зберегти прогрес у грі, налаштування програми та інші індивідуальні дані.

Регулярне створення резервних копій дозволяє:

- Відновити дані після втрати або пошкодження пристрою.
- Зберегти прогрес у додатках.
- Захиститися від випадкового видалення даних.

Дотримуючись цих рекомендацій, ви зможете захистити свої дані від втрати та забезпечити їх безпеку:

- Вмикайте автоматичне резервне копіювання.
- Перевіряйте правильність створення резервних копій.
- Створюйте резервні копії на зовнішній носій.

1.4.5 Обмеження доступу до кореня пристрою

Root-доступ може зробити ваш пристрій більш вразливим до атак. Якщо ви не є досвідченим користувачем, краще не отримувати root-доступ до свого пристрою.

Отримання root-доступу до мобільного пристрою відкриває перед користувачем широкі можливості для налаштування системи. Однак, разом з тим, це суттєво підвищує ризики безпеки. Кореневий доступ дозволяє встановлювати неперевірене програмне забезпечення, змінювати системні файли та налаштування, що може призвести до нестабільної роботи пристрою, втрати даних або навіть зараження шкідливим програмним забезпеченням.

Більшість сучасних смартфонів вже оптимізовані для ефективної роботи. Отримання root-доступу часто призводить до втрати гарантії виробника, а також може ускладнити отримання оновлень операційної системи. Крім того, зловмисники можуть скористатися вразливостями, які з'являються після отримання root-доступу, для проникнення в систему і викрадення ваших даних.

Root-доступ, як правило, необхідний розробникам, які створюють кастомні прошивки або додатки, що вимагають низькорівневого доступу до системи. Для звичайних користувачів, які хочуть лише налаштувати свій смартфон під свої

потреби, root-доступ, як правило, не потрібен. Більшість необхідних налаштувань можна здійснити за допомогою сторонніх додатків без отримання root-доступу.

1.4.6 Використання VPN

VPN шифрує ваш інтернет-трафік, що ускладнює зловмисникам перехоплення ваших даних. Крім того, VPN може приховати вашу IP-адресу, що дозволяє обійти географічні обмеження та захистити вас від фішингу.

VPN створює безпечне з'єднання між вашим пристроєм та сервером VPN. Весь ваш інтернет-трафік проходить через цей сервер, шифруючись перед відправкою.

VPN особливо корисний при використанні публічних Wi-Fi мереж, при доступі до чутливих веб-сайтів (наприклад, онлайн-банкінгу) та при подорожах.

Жоден метод захисту не є на 100% ефективним. Комбінація кількох методів допоможе значно знизити ризик зараження вашого пристрою шкідливим програмним забезпеченням. Будьте обережні, дотримуйтесь правил кібербезпеки і ваш смартфон завжди буде під надійним захистом.

1.5 Висновок до розділу 1

Світ кіберзагроз розвивається стрімко, постійно народжуючи нові, більш витончені види шкідливого програмного забезпечення. Традиційні методи захисту, які базуються на порівнянні з відомими зразками шкідливого коду, все частіше виявляються безсилими перед цим потоком. Швидка мутація вірусів, використання складних шифрувань та інших обхідних маневрів роблять такі методи неефективними. Крім того, стрімке зростання кількості мобільних додатків ускладнює ручний аналіз кожного з них, що створює додаткові труднощі для забезпечення безпеки.

Завдання розробників і спеціалістів з кібербезпеки – створення ефективних методів виявлення та протидії, здатних відповідати сучасним викликам. Це включає впровадження технологій машинного навчання для аналізу

поведінкових характеристик програм та виявлення нових видів шкідливого ПЗ на основі їхньої поведінки, а не лише за відомими сигнатурами.

Інтелектуальні методи виявлення, що використовують машинне навчання, стають усе більш затребуваними для виявлення нових типів атак та забезпечення належного рівня безпеки в мобільних екосистемах.

РОЗДІЛ 2 ОСНОВНІ ПІДХОДИ ДО ВИЯВЛЕННЯ ШКІДЛИВОГО ПЗ

Методи виявлення шкідливих програм для мобільних пристроїв використовуються як засоби боротьби з існуючими загрозами. Однак їхня ефективність залежить від різних факторів, зокрема від того, на чому зосереджений кожен метод. У цьому розділі ми систематизуємо наявні дослідження, виходячи з технік виявлення, описаних авторами, та аналізуємо їх функціональність і ефективність.

Виявлення шкідливої активності на мобільних пристроях може відбуватися за різними схемами, але дослідники ще не дійшли єдиної думки щодо класифікації. Одна з точок зору полягає в тому, що існують два основні типи методів аналізу шкідливих програм: статичний і динамічний.

Статичний аналіз передбачає дослідження програмного коду без його виконання. Він дозволяє виявити потенційно шкідливі фрагменти коду, але не може точно визначити, чи спрацюють вони в конкретній ситуації.

Динамічний аналіз передбачає спостереження за поведінкою програми під час її виконання. Він дозволяє виявити шкідливі дії, такі як несанкціонований доступ до даних, відправка SMS, встановлення з'єднань з підозрілими серверами.

Частина дослідників пропонує ще гібридний підхід в класифікації шкідливого ПЗ, який поєднує в собі переваги статичного та динамічного аналізу [6]. Він дозволяє досягти високої точності виявлення шкідливого програмного забезпечення. Прикладом гібридного аналізу є спочатку проводиться статичний аналіз для виявлення потенційно шкідливих фрагментів коду, а потім динамічний аналіз для підтвердження їх шкідливої діяльності

Інші дослідники пропонують інший підхід до класифікації, в якому статичне та динамічне виявлення виступають як підкатегорії для методів на основі сигнатур та аномалій [8].

Рисунок 2.1 ілюструє класифікацію методів виявлення шкідливих програм, що використовується в даній роботі.

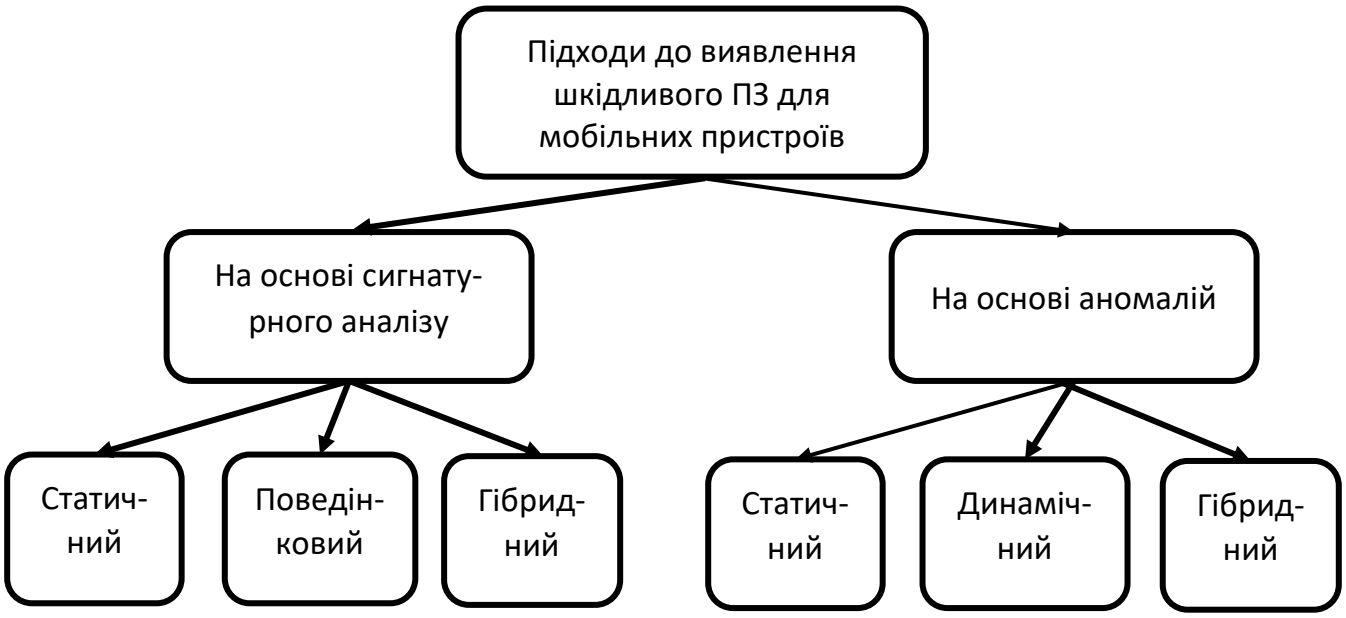

Рисунок 2.1 – Запропонована класифікація підходів до виявлення шкідливого ПЗ для мобільних пристроїв

За основний вектор категоризації методів виявлення шкідливих програм вибрано тип виявлення. Два основні типи виявлення — це методи, що базуються на сигнатурах, та методи на основі аномалій. Розглянемо детальніше кожен із запропонованих підходів.

2.1 Методи виявлення шкідливого ПЗ на основі сигнатурного аналізу

Виявлення, засноване на сигнатурах, збирає патерни та сигнатури відомих шкідливих програм і порівнює їх із підозрілими фрагментами коду, щоб визначити, чи є вони шкідливими або безпечними. Техніки виявлення на основі сигнатур поділяються на підкатегорії, що базуються на статичному, поведінковому та гібридному аналізі. Статичні методи виявлення на основі сигнатур використовуються більшістю комерційних антивірусних програм.

2.1.1 Статичне виявлення на основі сигнатур

Статичний аналіз (структурний аналіз) - це методика перевірки програмних додатків шляхом аналізу їхнього коду без його виконання. Цей процес дозволяє зрозуміти структуру коду і допомагає виявити функціональність, яку програма

буде виконувати. Покриття коду максимізується в цьому підході, оскільки аналізу піддається лише вихідний код. Іншою перевагою статичного аналізу є те, що він виявляє злісні наміри без ризику бути поміченим під час фактичного виконання програми і без втрат. Однак цей підхід має низьку ефективність у разі використання обфускації коду та динамічного завантаження коду [9]. Обфускація коду - це процес навмисного ускладнення або змінювання програмного коду таким чином, щоб його було важко зрозуміти або проаналізувати, зберігаючи при цьому функціональність програми. Мета обфускації — ускладнити реверс-інжиніринг, тобто перетворення скомпільованого коду назад у зрозумілий вихідний код.

Статичне виявлення на основі сигнатур використовує базу даних, що містить записи сигнатур зразків шкідливих програм, і порівнює об'єкти, які знаходяться в оперативній пам'яті або на SD-карті пристрою, з відповідними патернами. Енк та ін. [10] запропонували сервіс безпеки для операційної системи Android під назвою Kirin. Kirin сертифікує додаток під час встановлення, використовуючи набір правил безпеки, які є шаблонами, розробленими для виявлення підозрілих властивостей у конфігураціях безпеки додатків.

2.1.2 Динамічне виявлення на основі сигнатур

У статичному методі на основі сигнатур, отримання сигнатур відбувається під час розбору та аналізу вихідного коду шкідливої програми. Метою динамічного аналізу є визначення поведінки конкретного додатка під час його виконання. Додаток спостерігається шляхом його фактичного виконання на реальному пристрої (як реальна виконувана програма) або у віртуальному середовищі, наприклад, на віртуальному Android-пристрої (Android Virtual Device). Сигнатури в методах виявлення шкідливого ПЗ на основі динамічної поведінки отримуються після виконання шкідливого коду. Це здійснюється за допомогою попередньо налаштованих і визначених атакувальних патернів, які експерти надають заздалегідь для створення бази даних сигнатур або набору патернів.

Чен та ін. [11] запропонували підхід виявлення, який ідентифікує патерни загроз. Він аналізує виклики функцій, а також потік даних для виявлення шкідливої поведінки на пристроях Android. Більш конкретно, їхня схема використовує реверс інжиніринг для відтворення вихідного коду та клас-файлів з кожного додатка і створення відповідних викликів API та графів залежностей. На основі цих двох графів їхня система може виявляти патерни загроз, які можуть показати, чи намагається додаток отримати доступ до конфіденційної інформації або виконати будь-який незаконний доступ. Їхні експерименти показали 91,6% рівень виявлення на 252 шкідливих зразках.

2.1.3 Гібридне виявлення на основі сигнатур

Гібридне виявлення на основі сигнатур поєднує статичне та поведінкове виявлення на основі сигнатур. Папамарціванос та ін. [12] запропонували хостову та хмарну систему, яка працює за принципом краудсорсингу. Їхня система включає три основні сервіси: відстеження потоку приватності, краудсорсинг та виявлення й реагування на порушення конфіденційності. Клієнт взаємодіє з хмарними сервісами через TLS-з'єднання, щоб зняти навантаження з ресурсів пристрою. Більш конкретно, клієнт складається з трьох модулів: перевірка приватності, реагування та сенсор подій. Хмарна частина також включає три модулі: краудсорсинг, виявлення та оновлення підключень.

2.2 Методи виявлення шкідливого ПЗ на основі аналізу аномалій

Методи на основі аномалій використовують менш строгий підхід, який реалізовується шляхом спостереження за нормальною поведінкою пристрою протягом певного періоду часу та використання метрик цієї нормальної моделі як вектору порівняння для відхилень. Що стосується частини аналізу, то застосовуються як статичні, так і динамічні методи. Статичний підхід аналізує додаток до його встановлення шляхом розбирання коду, тоді як динамічний аналіз виконується під час роботи додатка, збираючи дані, такі як системні виклики та події. Як у динамічному, так і в статичному варіанті методи

виявлення на основі аномалій складаються з двох етапів: етапу навчання та етапу виявлення. Під час етапу навчання на неінфікованій системі нормальна робота спостерігається та відслідковується. З іншого боку, етап виявлення служить для тестування, коли відхилення від моделі навчального періоду вважаються аномаліями.

Методи статичного виявлення на основі аномалій не вимагають виконання шкідливого коду. Їхня функція полягає у перевірці коду потенційно шкідливого додатка на наявність специфічних фрагментів коду, підозрілої функціональності та інших поведінкових характеристик.

Існують різні джерела вилучення ознак в рамках цього підходу, зокрема автори [13] витягують інформацію про додаток з його маніфест-файлу та декомпільованого коду. Під час декомпіляції можна отримати вихідний код, який зазвичай включає виклики API, інтерфейси, змінні та логіку додатку. Аналізуючи декомпільований код, можна виявити, які саме API-виклики та функції використовуються додатком, а також з'ясувати, яку небажану або шкідливу поведінку здійснює додаток. Це дозволяє розпізнати потенційно небезпечні операції, навіть якщо додаток ще не був виконаний. Ще одним варіантом є аналіз дозволів додатку. В рамках комбіновано підходу можна використовувати поєднання дозволів додатків та викликів API. На першому етапі відбувається розпаковка APK-файлу додатку, щоб витягти маніфест і файли класів. Потім програма характеризується на основі запитуваних дозволів і викликів API.

2.2.1 Статичний підхід до виявлення аномалій

Статичний підхід виявлення аномалій здатний не тільки виявляти невідомі шкідливі програми, але й вказувати на потенційні вразливості у вихідному коді.

У методі статичного дослідження багато додатків стикаються з проблемами, такими як події, що керують виконанням додатків для Android. Проблеми, які потрібно враховувати для досягнення ефективних результатів, є наступними:

- Наявність кількох точок входу для кожного додатку.

- Можливість одночасного та асинхронного виконання кількох компонентів додатку.
- Часті зворотні виклики через етапи розробки додатку.
- Залучення як внутрішніх (intra-app), так і міждодаткових (inter-app) механізмів міжкомунікацій (ICC)
- Використання Java Reflection, обфускації та нативного коду також становить велику проблему [14],[15]

На рисунку 2.2 представлені представлені відомі елементи статичного аналізу.

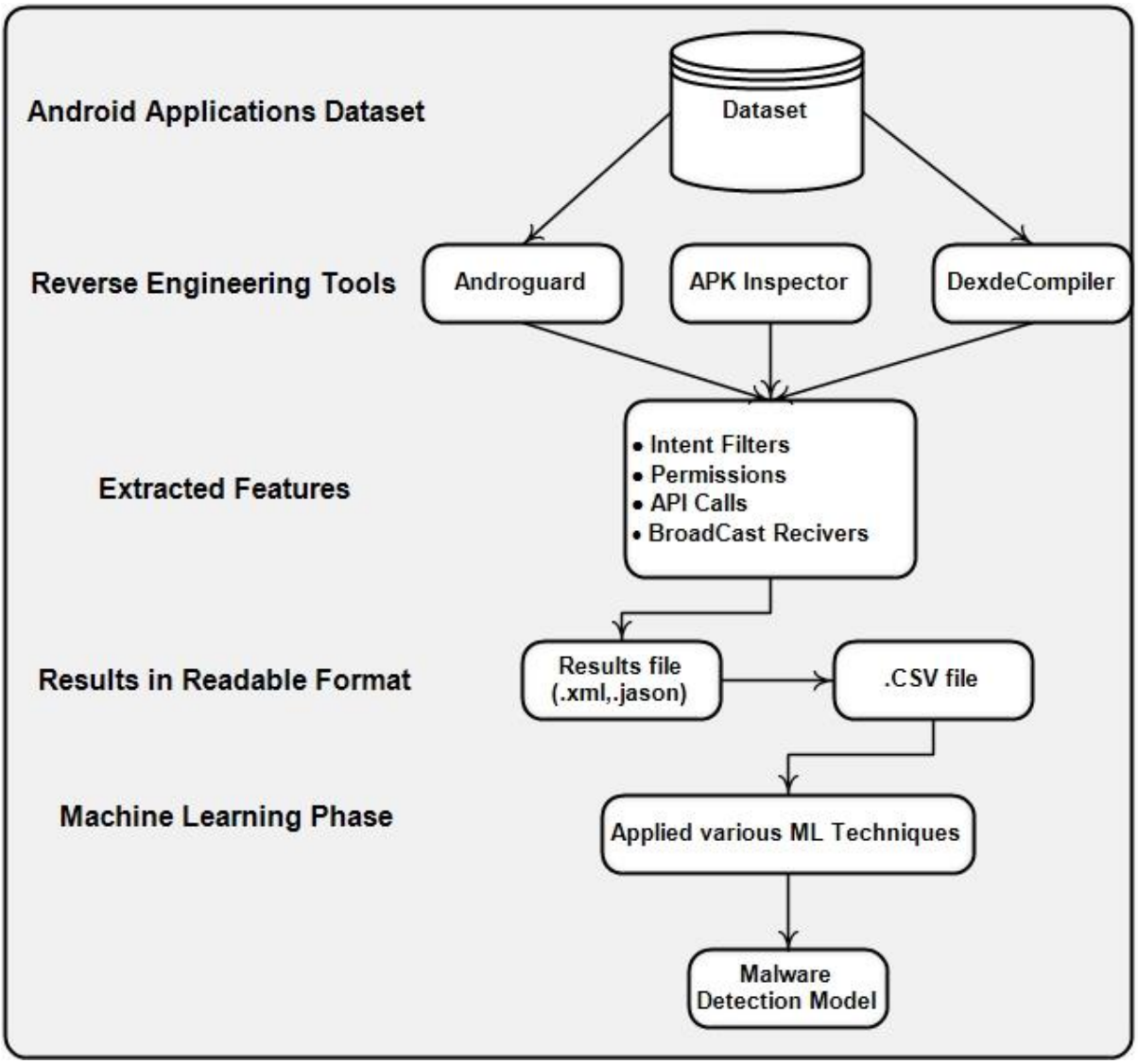

Рисунок 2.2 – Основні етапи статичного аналізу

До елементів статичного аналізу належать:

- набори даних,

- інструменти зворотного інжинірингу,
- виділені характеристики,
- результати файлів,
- етап машинного навчання
- модель виявлення шкідливих програм.

Розглянемо деякі приклади відомих рішень на основі статичного аналізу коду.

Грейс та ін. представили RiskRanker [16] з проактивною схемою для виявлення шкідливих програм нульового дня. Цей інструмент виконує двоетапний аналіз ризиків. На першому етапі його мета — визначити не обфусковані реалізації, що активують відомі root-діри, незаконні витрати на створення, та атаки на витік приватності. RiskRanker сканує рідні бінарні файли на наявність підписів експлойтів root для виявлення відомих root-експлойтів. Потім перевіряється витік приватної інформації шляхом застосування методу нарізки (slicing), щоб перевірити будь-яку інформацію, що передається до будь-яких підключених вузлів, яка може розкривати особисту інформацію. Крім того, він використовує набір евристичних технік, які допомагають виявляти процеси ухилення, такі як шифрування, динамічне завантаження коду та рефлексія Java, які не можуть бути виявлені на першому етапі запропонованого методу. Експерименти цього підходу базуються на 118 318 додатках, зібраних з різних сторонніх ринків додатків. В результаті RiskRanker успішно виявив 718 шкідливих програм, що належать до двадцяти дев'яти сімей шкідливих програм, зокрема 322 атаки нульового дня. Однак у RiskRanker є деякі обмеження, оскільки евристики застосовуються на другому етапі, а не на першому, через що шкідливі програми можуть легко уникнути виявлення на першому етапі.

Арц та ін. представили FlowDroid [84], який виконує статичний аналіз для виявлення шкідливих програм в Android-додатках. Він забезпечує правильну обробку зворотних викликів, ініційованих Android-фреймворком, при цьому аналіз потоку даних, контексту, об'єктів і даних, чутливих до полів, призводить до зменшення кількості хибних спрацьовувань. Подібно до цього, він моделює стани життєвого циклу Android-додатків і обробляє поширення зараження через

об'єкти користувацького інтерфейсу та зворотні виклики. Крім того, інноваційні алгоритми за запитом допомагають FlowDroid зберігати високу ефективність і точність, що дозволяє реалізувати повністю відкритий вихідний код. Експерименти з використанням FlowDroid на більше ніж 500 добросовісних додатках з Google Play Store та близько 1000 шкідливих додатків з virusShare підтверджують високу точність і повноту виявлення на рівні 86% та 93% відповідно. Під час застосування міжпроцедурного аналізу потоку даних FlowDroid не відстежує потік даних, що ґрунтується на міжкомпонентній комунікації (ICC) для додатків.

У таблиці 2.1 узагальнено типові статичні ознаки, які використовуються які використовуються в літературі для ефективного виявлення шкідливих програм для Android

Таблиця 2.1 – Типові статичні ознаки для виявлення шкідливого ПЗ

Категорія	Файли -джерело ознак	Ознаки	Літературні джерела
Дозволи	AndroidManifest.xml	Packages, permissions, strings	[17]
Наміри	-	intent_filter, broadcast_receivers	[17]
Ресурси	Resources.arsc	Interesting strings	[18]
Хеші	Other files	Hash keys, md5	[19]
Криптографічні операції	.xml	Data encryption, cp_traffic, uses_crypto, uses_reflection,java packages	[19]
Bytecode, opcode	.class	-	[20]

Однак статичний метод аналізу також має свої недоліки. Рівень помилкових спрацьовувань продовжує бути високим, а перевірка коду може бути ресурсозатратною, як за часом, так і за обчислювальною потужністю.

2.2.2 Динамічний підхід до виявлення аномалій

Динамічний аналіз спрямований на виявлення поведінкових ознак для Android-додатків, але існує багато викликів, таких як спрацьовування подій,

керовані ресурси Android та Binder-базована міжкомпонентна комунікація (ICC). Ось деякі з найбільших складнощів у контексті динамічного аналізу, які застосовуються до додатків та аналізуються у віртуальному середовищі з огляду на такі фактори:

- Обмеження часу на виконання та спостереження за поведінкою.
- Імітація відповіді графічного інтерфейсу користувача та реальних системних подій.
- Шкідливі додатки намагаються уникнути віртуалізації Android
- Велика кількість додатків, які потрібно оцінити системою виявлення.

На рисунку 2.3 наведено основні елементи динамічного аналізу

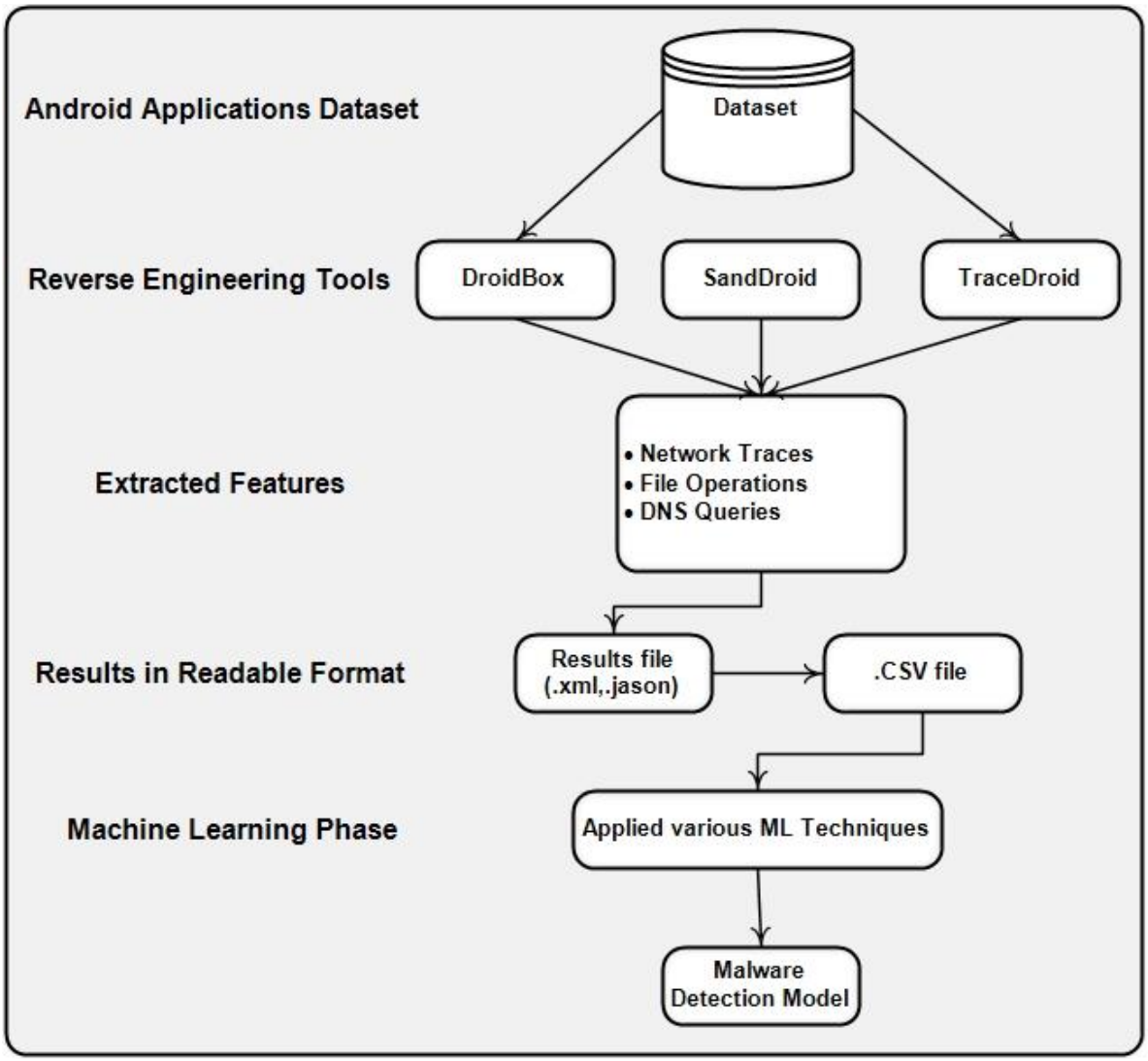

Рисунок 2.3 – Основні елементи динамічного аналізу

Рисунок 2.3 ілюструє загальні кроки та існуючі інструменти для динамічного аналізу шкідливих програм, які часто використовуються дослідниками в механізмах виявлення.

В таблиці 2.2 зібрані основні ознаки, що використовуються для подальшої класифікації для динамічного підходу

Таблиця 2.2 – Типові ознаки для виявлення шкідливого ПЗ у випадку динамічного підходу

Категорія	Файли -джерело ознак	Ознаки	Літературні джерела
API Calls	.json, .xml	getConnect(),getWifiState(), getConnectionInfo(), getCellLocation(), getDeafult(),getSubscriberId()	[21]
Network Operations	.xml,.json	TCP size, Duration, outgoing, ingoing and complete data flow	[21]
File Operations	.xml,.json	File read, write, file leaks	[22]
Running Services	.xml,.json	started_services	-
Network traces	.pcaps	Source IP, destination IP, host, port,path	[23]
System Calls	Classes.dex	classes, fields, methods, prototypes, types, strings	[21]

У динамічному виявленні на основі аномалій етапи навчання та виявлення відбуваються під час виконання додатка. Окрім здатності виявляти невідомі шкідливі програми, цей підхід також дозволяє виявляти атаки нульового дня. Однак, як уже згадувалося, проблема високого рівня помилкових спрацьовувань є досить актуальною, особливо для технік виявлення на основі аномалій. Для того, щоб пом'якшити цей недолік, під час етапу навчання необхідно створювати точні моделі нормальної поведінки.

2.2.3 Гібридний підхід до виявлення аномалій

Гібридне виявлення на основі аномалій поєднує як статичне, так і динамічне виявлення на основі аномалій. Основна ідея гібридного аналізу полягає в

поєднанні характеристик статичного та динамічного аналізу, що включає вивчення коду та поведінки додатка, як показано на рисунку 2.4.

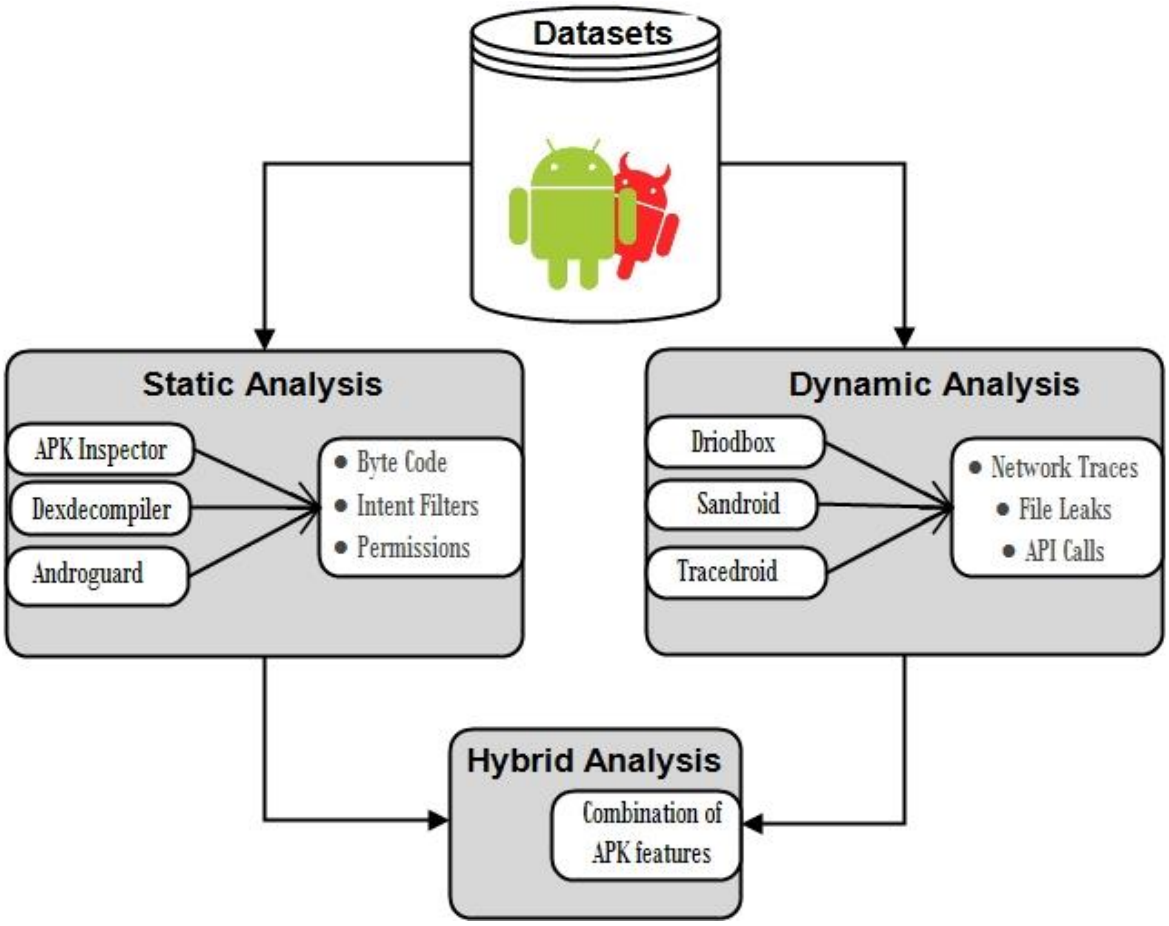

Рисунок 2.4 – Гібридний аналіз, як комбінація статичного та динамічного підходів

Також показано, що гібридний аналіз накладає параметри статичного та динамічного аналізу. Дослідники використовували статичні та динамічні інструменти, такі як androguard, APK inspector, Droidbox, Sandroid та Tracedroid, зокрема для витягування характеристик з додатків, як ми обговорювали у підходах до виявлення мобільного шкідливого ПЗ. Хоча в літературі є менше досліджень, які використовують метод гібридного аналізу для виявлення мобільних шкідливих програм, точність виявлення шкідливих програм в гібридному аналізі вища порівняно з використанням лише статичних чи динамічних методів.

Модель HybriDroid [22], яка використовує як статичний, так і динамічний аналіз для виявлення витоків даних в Android-додатках. У цьому випадку статичний аналіз перевіряє джерела додатків, які можуть спричиняти втрату даних. З іншого боку, динамічна техніка моніторить поведінку додатка під час виконання для виявлення шкідливого ПЗ. Наприкінці автори порівнюють цю модель з інструментами IccTa, DroidGuard та DroidBox, що доводить її ефективну роботу.

Отже методи виявлення спаму на основі аномалій здатні обробляти нові види спаму. Вони також можуть враховувати персональні уподобання користувача, створюючи адаптовані моделі для кожного індивідуального випадку, що робить системи виявлення спаму більш ефективними і гнучкими.

Бачимо важливу роль у класифікації шкідливого ПЗ на мобільних додатках відіграють методи машинного навчання, тож розглянемо детальніше їх роль.

2.3 Методи машинного навчання для виявлення аномалій

Як бачимо з попередніх розділів, основними інтелектуальними методами виявлення шкідливого ПЗ є методи машинного навчання для виявлення аномалій.

Загалом класичні методи машинного навчання поділяються на керовані (з вчителем) та некеровані (без вчителя). У керованому навчанні модель навчається на основі промітькованих даних, тобто на вхідних даних, які вже мають відомі відповідні чисельні результати або категорійні мітки. Алгоритм використовує ці мітки для того, щоб навчитися знаходити залежності між вхідними характеристиками та вихідними значеннями. Мета такого підходу - навчити модель передбачати або класифікувати нові невідомі дані, ґрунтуючись на знайдених закономірностях. Наприклад, у задачах класифікації (як шкідливе/не шкідливе ПЗ) або регресії (як прогнозування ціни нерухомості за характеристиками) модель отримує чітке уявлення про те, що є правильними відповідями, і намагається застосувати ці знання до нових даних. Перевагою

цього підходу є висока точність прогнозів, оскільки модель має чіткі орієнтири для навчання.

Натомість, у некерованому навчанні модель працює з неміткованими даними, де немає заздалегідь визначених відповідей. Алгоритм у такому випадку намагається знайти закономірності або структуру в даних без наявності чітких міток. Мета некерованого навчання виявити приховані шаблони або групи в даних, наприклад, через кластеризацію або зменшення розмірності. Це може бути корисно для вивчення структури даних без потреби у зовнішніх мітках, як у випадку з групуванням клієнтів за схожістю в покупках або спрощенням складних даних за допомогою методів зменшення розмірності, таких як PCA. Перевагою цього підходу є можливість знаходити невідомі закономірності, які можуть бути корисними для подальшого аналізу, навіть без точних міток для даних.

Основними методами керованого навчання є класифікація та регресія.

Класифікація — це тип задачі машинного навчання, при якому алгоритм навчається на мітках даних, щоб відносити нові, невідомі дані до однієї з попередньо визначених категорій або класів. Наприклад, класифікація може використовуватися для виявлення спаму в електронній пошті (спам/не спам) або для виявлення атаки в мережі (атака/нормальний трафік). Класифікаційні алгоритми намагаються знайти залежність між вхідними характеристиками (ознаками) та класами, що дозволяє автоматично розпізнавати нові об'єкти за цими класами.

Регресія — це тип задачі машинного навчання, де метою є прогнозування числового значення на основі вхідних даних. Відмінність регресії від класифікації полягає в тому, що результатом є неперервна величина, а не категорія. Наприклад, регресія може використовуватися для прогнозування ймовірності атаки, або кількості підключень до сервера за певний проміжок часу. Алгоритми регресії намагаються знайти функціональну залежність між входом і виходом, яка мінімізує помилку передбачення.

Кластеризація — це техніка машинного навчання, що використовується для групування подібних об'єктів або даних в класи або кластери без попередніх

міток або класів. Метою є виявлення природних груп в даних, де об'єкти в одному кластері мають більшу схожість між собою, ніж з об'єктами з інших кластерів. Кластеризація часто застосовується для вивчення структури даних, наприклад, у маркетингу для сегментації споживачів або в кібербезпеці для виявлення різних груп атак.

Методи машинного навчання для виявлення аномалій є важливими інструментами в кібербезпеці, оскільки вони дозволяють автоматично виявляти незвичайну або потенційно шкідливу поведінку в інформаційних системах, мережах та додатках. Виявлення аномалій засноване на ідеї, що відхилення від нормальної поведінки можуть вказувати на атаки, зловмисні дії або системні збої. Нижче наведено кілька основних методів машинного навчання, які використовуються для виявлення аномалій в контексті кібербезпеки.

2.3.1 Методи виявлення аномалій на основі класифікації (Supervised Learning)

У цьому випадку модель навчається на міткованих даних, де кожен зразок даних має позначку «нормальний» або «аномальний». За допомогою цих міток модель вчиться розрізняти між нормальним і аномальним трафіком, поведінкою або подіями. Найпоширеніші алгоритми включають логістичну регресію, метод опорних векторів та випадкові ліси.

Логістична регресія використовується для класифікації даних як «нормальні» або «аномальні» на основі вхідних ознак. Наприклад, у мережевій безпеці логістична регресія може передбачити те, що конкретний мережевий запит є шкідливим.

Метод опорних векторів (SVM) використовується для поділу даних на класи на основі навчальних даних. У кібербезпеці SVM може бути використаний для виявлення аномалій у поведінці користувачів або мережевому трафіку.

Випадкові ліси – це потужний інструмент машинного навчання, який ефективно використовується для виявлення аномалій. Цей метод працює шляхом створення множини дерев рішень, кожне з яких навчається на випадково вибраній підмножині даних. При класифікації нового об'єкта, кожне дерево

"голосує", відносячи його до певного класу. Якщо більшість дерев класифікує об'єкт як аномалію, то він з високою ймовірністю є таким. Випадкові ліси особливо корисні для виявлення аномалій, оскільки вони стійкі до перенавчання, добре справляються з великими обсягами даних і здатні виявляти складні нелінійні залежності.

2.3.2 Методи виявлення аномалій на основі некерованого навчання (Unsupervised Learning)

У випадку методів без нагляду модель не має міток даних, тому завдання полягає в тому, щоб сама система виявила структуру або аномалії в даних. Цей метод корисний, коли неможливо зібрати велику кількість мітованих даних або коли ми маємо справу з новими, невідомими загрозами.

В рамках некерованого підходу за допомогою методів кластеризації (наприклад, K-means або DBSCAN) можна групувати схожі зразки даних. Якщо зразки не потрапляють до жодної з груп або вони істотно відрізняються від інших, це може вказувати на аномалію. У кібербезпеці цей метод може бути використаний для виявлення нетипового мережевого трафіку або підозрілих активностей користувачів.

Проте існую спеціальні алгоритми виявлення аномалій:

- One-Class Support Vector Machine (One-Class SVM),
- One-Class SVM with Stochastic Gradient Descent (SGD),
- Isolation Forest (iForest), Local Outlier Factor (LOF),
- Robust Covariance (Elliptic Envelope).

Ці алгоритми спеціально призначені для виявлення аномалій в даних без попереднього навчання на мітках. Вони використовують методи ізоляції або обчислення відстані між точками в просторі ознак для виявлення викидів або аномалій. Вони можуть бути застосовані для виявлення таких кіберзагроз, як зловмисні програми, фішинг або атаки типу "brute-force".

2.3.3 Методи виявлення аномалій на основі глибокого навчання (Deep Learning)

Глибокі нейронні мережі (особливо рекурентні нейронні мережі (RNN) та автокодувальники (Autoencoders)) використовуються для виявлення аномалій, особливо в складних або великих наборах даних, таких як мережевий трафік або журналювання поведінки користувачів.

Автокодувальники (Autoencoders) - це тип нейронної мережі, який навчається на відновленнях вхідних даних. При виявленні аномалій, автокодувальник може мати великі помилки при відновленні аномальних зразків, оскільки він не «навчений» на таких аномальних даних. Цей метод застосовується для виявлення аномалій у мережевому трафіку або в системах моніторингу подій безпеки.

Рекурентні нейронні мережі (RNN) – це тип мереж, які добре працюють з часовими послідовностями, що є типовим для багатоаспектних даних кібербезпеки (наприклад, логів системи чи мережевого трафіку). Вони використовуються для виявлення аномальної поведінки користувачів або атак типу "denial of service" (DoS) за аномальними шаблонами трафіку.

2.4 Висновки до 2 розділу

В даному розділі наведено основні теоретичні відомості стосовно використання статичного, динамічного та гібридного підходу, сформульовано основні етапи кожного підходу, а також виділено типові для кожного підходу характеристики. Також описано моделі машинного навчання та їх використання для виявлення аномалій

РОЗДІЛ 3 ПРАКТИЧНА РЕАЛІЗАЦІЯ МЕТОДІВ МАШИННОГО НАВЧАННЯ ДЛЯ ВИЯВЛЕННЯ ШКІДЛИВОГО ПЗ ДЛЯ ANDROID

3.1 Вибір середовища

Python, зі своєю елегантною синтаксичною структурою та потужною екосистемою бібліотек, давно завоював визнання в спільноті розробників, особливо в галузі машинного навчання. Його універсальність та гнучкість роблять його незамінним інструментом для створення ефективних систем виявлення шкідливого програмного забезпечення.

Відповідно до [24] 87% респондентів використовують Python у щоденній роботі з даними (рис.3.1).Причиною цього є ряд переваг, які описані нижче.

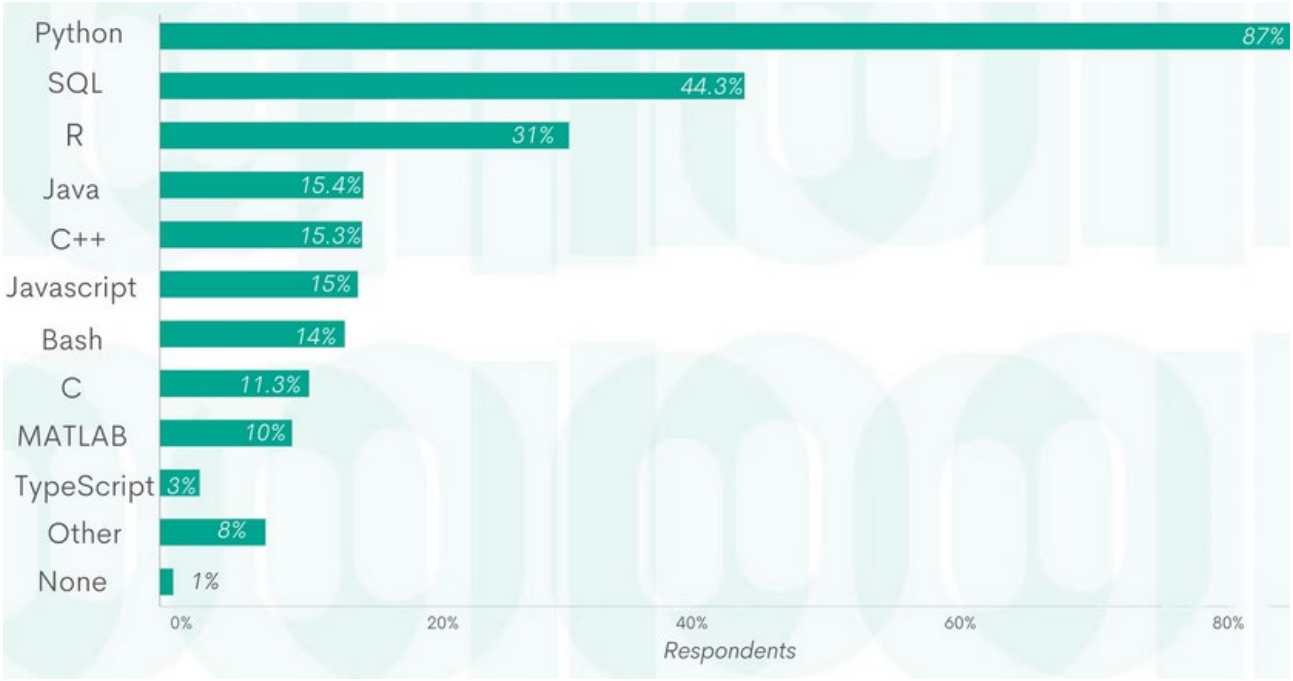

Рисунок 3.1 – Статистика використання різних мов програмування для data science

Чіткий та інтуїтивний синтаксис Python дозволяє розробникам швидко втілювати ідеї в код, а також легко розуміти та підтримувати вже написаний код. Це особливо важливо в контексті складних алгоритмів машинного навчання, де навіть невелика помилка може призвести до некоректних результатів.

Python пропонує величезну кількість спеціалізованих бібліотек, які значно спрощують роботу з даними та розробку моделей машинного навчання. Серед найбільш популярних: NumPy для числових обчислень, Pandas для аналізу даних, Matplotlib для візуалізації, Scikit-learn для традиційних алгоритмів машинного навчання та TensorFlow/Keras для глибокого навчання. Ці бібліотеки забезпечують готові інструменти для вирішення широкого спектра завдань, пов'язаних з аналізом шкідливого коду.

Активна спільнота розробників Python постійно вносить свій внесок у розвиток мови та її екосистеми. Це означає, що ви завжди матимете доступ до нових інструментів, алгоритмів та найкращих практик. Крім того, велика кількість онлайн-ресурсів, таких як форуми, блоги та навчальні матеріали, дозволяє легко знайти відповіді на будь-які питання.

Python легко інтегрується з іншими мовами програмування та інструментами, що дозволяє використовувати його в складних системах безпеки. Наприклад, можна використовувати Python для розробки моделей машинного навчання, а потім інтегрувати їх у системи на основі C++ або Java.

Python дозволяє розробляти як прості скрипти для аналізу невеликих наборів даних, так і складні розподілені системи для обробки великих обсягів інформації. Це робить його ідеальним інструментом для вирішення різноманітних задач, пов'язаних з виявленням шкідливого програмного забезпечення.

Хоча Python відомий своєю простотою, він також здатний забезпечувати високу продуктивність за рахунок використання таких технологій, як JIT-компіляція (PyPy) та обчислення на графічних процесорах (GPU). Це дозволяє ефективно обробляти великі обсяги даних та тренувати складні моделі машинного навчання.

Python є однією з найпопулярніших мов програмування у світі, що забезпечує велику кількість кваліфікованих фахівців та доступність різноманітних навчальних матеріалів. Це спрощує процес розробки та підтримки систем виявлення шкідливого програмного забезпечення.

Таким чином, Python є оптимальним вибором для розробки систем виявлення шкідливого програмного забезпечення завдяки своїй простоті, потужній екосистемі, активній спільноті та гнучкості. Використання Python дозволяє створювати ефективні, масштабовані та адаптивні рішення для боротьби з кіберзагрозами.

3.2 Опис датасету CICMalDroid 2020

CICMalDroid 2020 — це датасет, розроблений Канадським центром кібербезпеки (CIC - Canadian Institute for Cybersecurity), який призначений для виявлення шкідливих програм та аналізу безпеки мобільних додатків на платформі Android. Датасет включає в себе зібрані дані про поведінку як шкідливих, так і незловмисних додатків, що дає змогу створювати моделі для класифікації, виявлення аномалій та оцінки безпеки мобільних пристроїв.

Датасет містить 11598 зразок Android-додатків з різноманітних джерел, таких як VirusTotal, блог безпеки Contagio, AMD, MalDozer та інші датасети, що використовувалися в останніх дослідженнях у цій сфері. Дані були отримані в період з грудня 2017 року по грудень 2018 року. Для фахівців з кібербезпеки важливо класифікувати Android-додатки за категоріями шкідливих програм, щоб можна було вжити відповідних заходів для боротьби з ними. Тому наш датасет охоплює п’ять основних категорій:

- Adware,
- Banking malware,
- SMS malware,
- Riskware,
- Benign.

Adware - це рекламне програмне забезпечення, яке зазвичай приховується всередині легітимних додатків, заражених шкідливим ПЗ та розповсюджується через сторонні маркетплейси. Оскільки реклама постійно відображається через спеціалізовану бібліотеку, Adware не зупиняється навіть якщо користувач намагається закрити додаток. Це ПЗ може інфікувати пристрій, отримувати root-

доступ і змушувати пристрій завантажувати нові варіанти Adware, при цьому даючи зловмисникам доступ до особистої інформації.

Banking malware - це шкідливе програмне забезпечення, яке має на меті отримати доступ до онлайн-банківських рахунків користувачів, маскуючись під оригінальні банківські додатки або веб-інтерфейси. Більшість таких програм є троянами, які проникають у пристрій, крадуть банківські дані (логіни, паролі) і передають ці дані на сервер командування та контролю (C&C).

SMS malware використовує SMS-сервіс як канали для проведення атак, перехоплюючи повідомлення для виконання шкідливих дій. Зловмисники завантажують шкідливі програми на свої сервери, де вони інтегруються з SMS-сервісом, а через сервер C&C здійснюється контроль над атаками: відправка шкідливих повідомлень, перехоплення SMS і крадіжка даних.

Mobile Riskware - це програми, які зазвичай є легітимними, але можуть бути використані зловмисниками для шкідливих цілей. Вони можуть перетворитися на інші форми шкідливих програм, такі як Adware чи Ransomware, встановлюючи додаткові заражені додатки. Ця категорія включає лише одну основну версію, яка часто позначається як "Riskware" у VirusTotal.

Benign - усі інші додатки, які не відносяться до вищезгаданих категорій, вважаються безпечними (benign), що означає відсутність шкідливого програмного забезпечення. Для перевірки їхньої безпеки автори датасету відсканували всі безпечні зразки за допомогою VirusTotal.

CICMalDroid 2020 містить особливості і характеристики для виявлення мобільних шкідливих програм. Датасет охоплює різноманітні типи атак і загроз, що можуть бути знайдені в Android-додатках. Датасет містить ознаки як для шкідливих, так і для незловмисних Android-додатків. Для кожного додатку в датасеті надається його профіль, що включає характеристики мережевого трафіку, API-виклики, системні логи, взаємодії з пристроєм, інтеракція з пристроєм тощо.

Усі APK-файли спочатку виконувались в CopperDroid, і поведінка під час виконання записувалась в журнали. Результати аналізу, отримані від

CopperDroid, доступні у форматі JSON для зручного парсингу та додаткової допоміжної інформації. Результати аналізу класифіковані на три основні групи:

- Статично витягнуті дані, наприклад, інтенти; дозволи та сервіси; частотні підрахунки для різних типів файлів; випадки обфускації та виклики чутливих API;
- Динамічно спостережувана поведінка, яка в основному розбита на три категорії: системні виклики, виклики через binder та складні поведінки;
- PCAP-дані всього мережевого трафіку, захопленого під час аналізу.

Для збору даних використовуються методи як статичного, так динамічного аналізу додатків, що дозволяє реєструвати поведінку додатків у процесі їх виконання, а також методи статичного аналізу для дослідження коду додатків до запуску. Загалом датасет містить 471 ознаку.

3.3 Реалізація моделей машинного навчання для виявлення шкідливого ПЗ

Для обробки та аналізу результатів, зібраних на основі статичних і динамічних характеристик додатків, необхідно застосувати методи машинного навчання, зокрема для класифікації шкідливих і безпечних додатків. Для цього підходять наступні бібліотеки:

- scikit-learn - популярна бібліотека для машинного навчання, яка підтримує багато алгоритмів для класифікації (наприклад, SVM, логістична регресія, дерева рішень тощо).
- TensorFlow або Keras - бібліотеки для глибокого навчання, які можуть бути корисні для побудови нейронних мереж, зокрема для аналізу зображень або виявлення складних патернів в даних.
- pandas та numpy - для обробки даних і маніпулювання великими таблицями результатів, наприклад, для роботи з CSV та JSON-файлами результатів аналізу.

Щоб краще розуміти результати аналізу, корисно використовувати бібліотеки для візуалізації даних:

- matplotlib та seaborn - бібліотеки для побудови графіків і діаграм, що допомагають аналізувати патерни в мережевому трафіку, системних викликах і іншому.
- plotly - інтерактивні графіки для побудови більш складних візуалізацій даних.

Імпорт необхідних бібліотек наведено у лістингу 3.1

Лістинг 3.1 – Імпорт необхідних бібліотек

```
import numpy as np
import pandas as pd
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, precision_score,
recall_score, f1_score
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.linear_model import LogisticRegression
from sklearn.svm import SVC
from sklearn.model_selection import cross_val_score
import seaborn as sns
import matplotlib.pyplot as plt
import os
for dirname, _, filenames in os.walk('/malware/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

Функції для завантаження та перегляду та визначення розміру датасету наведені в лістингу 3.2

Лістинг 3.2 - Функції для завантаження та перегляду та визначення розміру датасету

```
    df    =    pd.read_csv('/malware/input/cicmaldroid-2020/feature
_vectors_syscallsbinders_frequency_5_Cat.csv')
    # Labels
    # Adware: 1,253-------1
    # Banking: 2,100------2
    # SMS malware: 3,904--3
    # Riskware: 2,546-----4
    # Benign: 1,795-------5
    df.head()
    df.shape
```

На рисунку 3.2 наведено результат виконання команди head(), а на рисунку 3.3 результат виконання команди shape.

	ACCESS_PERSONAL_INFO___	ALTER_PHONE_STATE___	ANTI_DEBUG____	CREATE_FOLDER____	CREATE_PROCESS`___
0	1	0	0	3	0
1	3	0	0	6	0
2	2	0	0	4	0
3	1	0	0	4	0
4	3	0	0	11	0

Рисунок 3.2 - Результат виконання команди df.head()

```
In [4]: df.shape

Out[4]: (11598, 471)
```

Рисунок 3.3 - Результат виконання команди df.shape

Застосування команди df.isnull().sum() підвердило відсутність нульових даних в датасеті, що значно спрощує попередню обробку. Результат виконання команди наведено на рисунку 3.4

```
Out[5]:
        ACCESS_PERSONAL_INFO___     0
        ALTER_PHONE_STATE___        0
        ANTI_DEBUG_____             0
        CREATE_FOLDER_____          0
        CREATE_PROCESS`_____        0
                                   ..
        watchRotation               0
        windowGainedFocus           0
        write                       0
        writev                      0
        Class                       0
        Length: 471, dtype: int64
```

Рисунок 3.4 – Результат виконання команди df.isnull().sum()

Лістинг 3.3 виконує візуалізацію розподілу шкідливого ПЗ для розуміння збалансованості датасету.

Лістинг 3.3 – Код для візуалізації розподілу різних типів шкідливо ПЗ

```
label_counts = df['Class'].value_counts()
labels = label_counts.index.tolist()
counts = label_counts.tolist()
plt.bar(labels, counts)
plt.xlabel('Malware')
plt.ylabel('Counts')
plt.title('Malware Distribution in Dataset')
plt.show()
```

Результат виконання лістингу на рисунку 3.5. З лістингу 3.2 очевидно, що міткою 1 позначено рекламне шкідливе ПЗ (Adware), міткою 2 – банківське шкідливе ПЗ (Banking), міткою 3 – шкідливі повідомлення sms, міткою 4 – Riskware, та міткою 5 – нормальне ПЗ (Benign). Очевидно, що датасет сформований таким чином, щоб навчити модель в першу чергу визначати шкідливе ПЗ. Проте невідомо, як моделі будуть працювати з реальними даними, де відсоток нешкідливих додатків перевищує 90%.

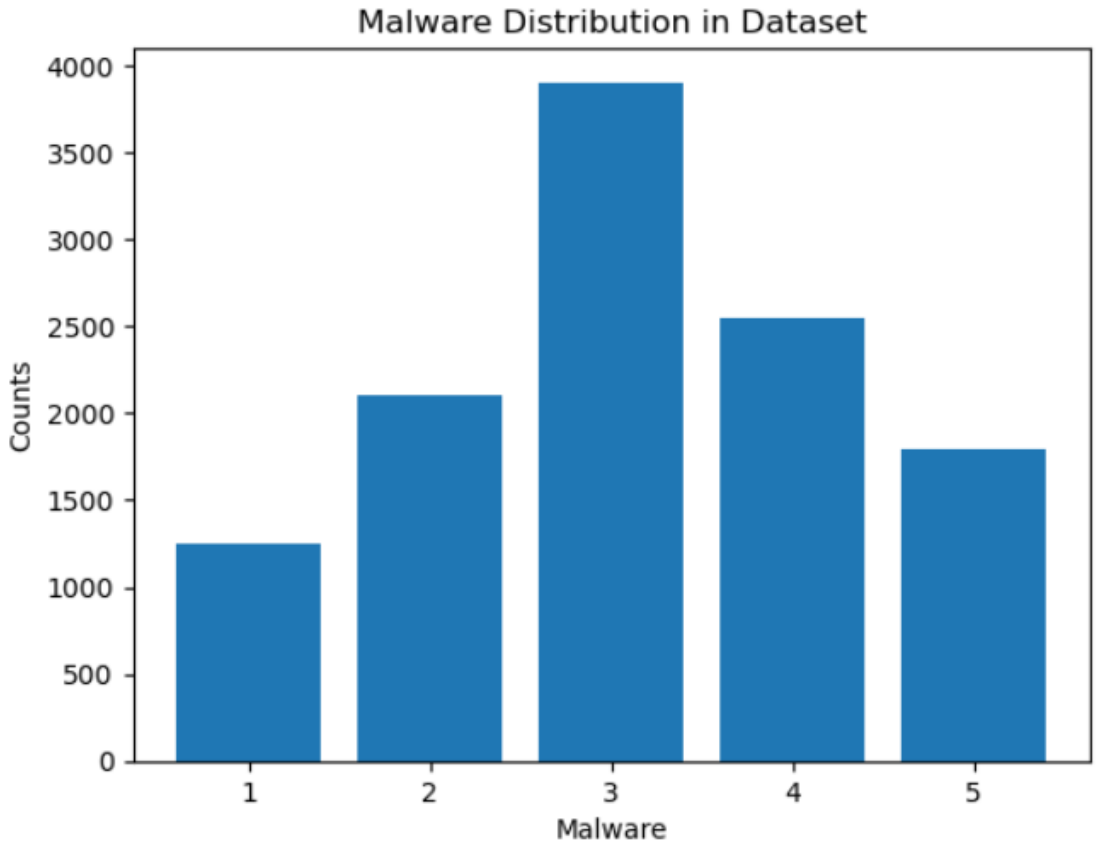

Рисунок 3.5 – Візуалізація розподілу типів шкідливого ПЗ (мітки 1-4) та нешкідливого (мітка 5)

Для встановлення якості моделей класифікації, необхідно поділити датасет на навчальний та тестовий (лістинг 3.4)

Лістинг 3.4 – Поділ датасету на навчальний та тестовий

```
X = df.drop(columns=['Class'])  # Features
y = df['Class']  # Target
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
stratify=y,test_size=0.2, random_state=42)
```

Наявність великої кількості ознак у датасеті, хоча й може здаватися перевагою, оскільки забезпечує більш повну інформацію про об'єкти, насправді може призводити до низки проблем при застосуванні машинного навчання.

Зокрема, це може призводити до того, що дані стають більш розрідженими, тобто між об'єктами з'являється більше порожніх просторів. Це ускладнює

навчання моделей, оскільки алгоритмам стає важче знаходити закономірності та узагальнювати знання на нові дані. Перевірка даних нашого набору даних справді підтверджує те, що дані є розрідженими (багато нулів у стовпцях ознак).

Велика кількість ознак збільшує ризик перенавчання моделі. Модель може занадто добре підлаштуватися під тренувальні дані, включаючи шум і випадкові особливості, і погано узагальнювати на нових даних.

Збільшення кількості ознак призводить до збільшення розміру матриці даних і, відповідно, до збільшення обчислювальних ресурсів, необхідних для навчання моделі. Це може значно сповільнити процес навчання і ускладнити використання складних алгоритмів.

В даному датасеті є 471 ознака. З великою кількістю ознак виникає проблема вибору найбільш інформативних ознак, які дійсно впливають на цільову змінну. Неправильний вибір ознак може призвести до зниження якості моделі. В даному дослідженні було обрано метод ANOVA для зниження розмірності простору ознак [25].

ANOVA - відбір ознак на основі дисперсійного аналізу, є потужним методом, який використовується для визначення найважливіших ознак у наборі даних перед застосуванням алгоритмів машинного навчання. Цей метод дозволяє оцінити, наскільки сильно кожна ознака впливає на цільову змінну, і вибрати ті ознаки, які найбільш суттєво відрізняються між різними групами даних.

Основою ANOVA є порівняння дисперсій всередині груп та між групами. Якщо дисперсія між групами значно більша за дисперсію всередині груп, це свідчить про те, що ознака має сильний вплив на цільову змінну і є важливою для побудови моделі. ANOVA дозволяє обчислити F-статистику та p-значення для кожної ознаки, що дозволяє визначити статистичну значущість відмінностей між групами.

ANOVA має ряд переваг. По-перше, він простий у розумінні та реалізації. По-друге, він дозволяє оцінити не тільки чи є ознака важливою, але й наскільки сильно вона впливає на цільову змінню. По-третє, він може бути застосований як для числових, так і для категоріальних ознак. Однак, ANOVA має також деякі

обмеження. Він найбільш ефективний для лінійних залежностей між ознаками та цільовою змінною і може бути чутливим до виключень. Крім того, він не враховує взаємодію між ознаками.

В лістингу 3.5 представлено код для застосування методу ANOVA для дослідження впливу кількості ознак на точність датасету

Лістинг 3.5 – Код для зменшення простору ознак

```
    #ANOVA-based feature selection
    num_features_to_select = 120
    selector          =          SelectKBest(score_func=f_classif,
k=num_features_to_select)
    X_train_selected = selector.fit_transform(X_train, y_train)
    X_test_selected = selector.transform(X_test)
    # Get the indices of selected features
    selected_feature_indices = selector.get_support(indices=True)
    # Convert selected features to a DataFrame
    X_train = X_train.iloc[:, selected_feature_indices]
    X_test = X_test.iloc[:, selected_feature_indices]
```

В даній роботі проведено дослідження впливу кількості ознак на точність моделей, результати якого будуть представлені нижче. В результаті застосування даного коду кількість ознак зменшилась до 120, а кількість записів до 2320.

Додатково було проведено нормалізацію даних (лістинг 3.6)

Лістинг 3.6 – Нормалізація даних

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

Після етапу попередньої обробки, необхідно провести навчання моделей. В лістингу 3.7 наведено приклад навчання моделі наївного Баєса для навчального датасету та тестування на тестовому датасеті.

Лістинг 3.7 – Навчання моделі

```
nb_model = GaussianNB()
nb_model.fit(X_train, y_train)
# Predict labels on the test set
y_pred = nb_model.predict(X_test)
```

Схожим чином, було імплементовано моделі випадкового лісу, методу опорних векторів, логістичної регресії та k-найближчих сусідів.

3.4 Тестування моделей та оцінка результатів

Перед тим, як перейти до самих метрик оцінки точності класифікації необхідно ввести концепцію для опису цих метрик – матрицю помилок (confusion matrix). На рисунку 3.6 наведено її схематичне зображення:

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Рисунок 3.6 – Матриця помилок класифікації

Для тестування використовувались типові показники оцінки якості класифікаційних моделей: точність (accuracy), влучність (precision), повнота (recall)

Точність (accuracy) – це найпростіший і найінтуїтивніший показник, який обчислюється як відношення правильно класифікованих об'єктів до загальної кількості об'єктів. Однак, точність може бути оманливою, особливо коли класи не збалансовані. Наприклад, якщо в наборі даних один клас значно переважає над іншими, то модель може досягти високої точності, просто завжди прогнозуючи найбільш поширений клас. Даний показник оцінюється за формулою:

$$\text{Accuracy} = \frac{Correct\ predictions}{All\ predictions} \tag{3.1}$$

Влучність (precision) є одним із ключових показників якості класифікаційної моделі, який характеризує точність позитивних прогнозів. Іншими словами, влучність відповідає на питання: "З усіх об'єктів, які модель класифікувала як позитивні, яка частка дійсно є позитивними?". Висока влучність означає, що модель рідко помиляється, класифікуючи негативні об'єкти як позитивні. У випадку класифікації програмного забезпечення, якщо модель зробила 80 правильних позитивних прогнозів зі 100 загальних позитивних прогнозів, влучність цієї програми складатиме 80%. Це означає, що хоча програма виявила 80% реальних загроз, вона також помилково класифікувала 20% безпечних файлів як шкідливі (помилково позитивні результати). Влучність оцінюється за формулою:

$$\text{Precision} = \frac{True\ Positives}{Total\ Predicted\ Positives} \tag{3.2}$$

Повнота (recall) показує, яку частку позитивних прикладів модель змогла правильно класифікувати. Іншими словами, це міра того, наскільки добре модель виявляє позитивні класи. Повнота є важливим показником, коли помилкове пропущення позитивного прикладу має серйозні наслідки

$$\text{Recall} = \frac{True\ Positives}{Total\ Real\ Positives} \tag{3.3}$$

F1-міра є гармонійним середнім між точністю і повнотою. Вона дозволяє об'єднати ці два показники в один і є корисною, коли необхідно досягти балансу між точністю і повнотою. F1-міра особливо корисна, коли класи не збалансовані:

$$\text{F1} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.4}$$

Побудуємо матриці помилок для всіх моделей, що були використані в роботі та порахуємо на їх основі вище наведені метрики для оцінки якості моделей.

На рисунку 3.7 наведено теплова карта (heatmap) після застосування моделі Байеса. По стовпцях наведено реальні дані, а по стрічках – прогнозовані.

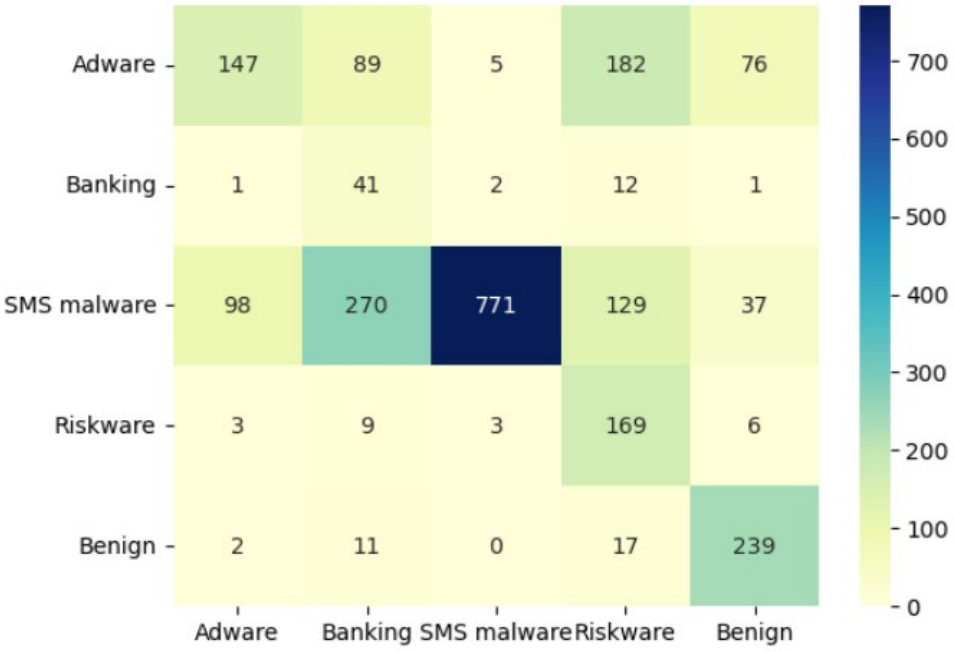

Рисунок 3.7 –Теплова карта для моделі Байеса

Очевидно, що найкраще модель спрацювала для sms malware, мабуть тому, що цих даних було найбільше в датасеті, але загалом класифікація пройшла погано.

На рисунку 3.8 наведено теплову картку для моделі випадкового лісу

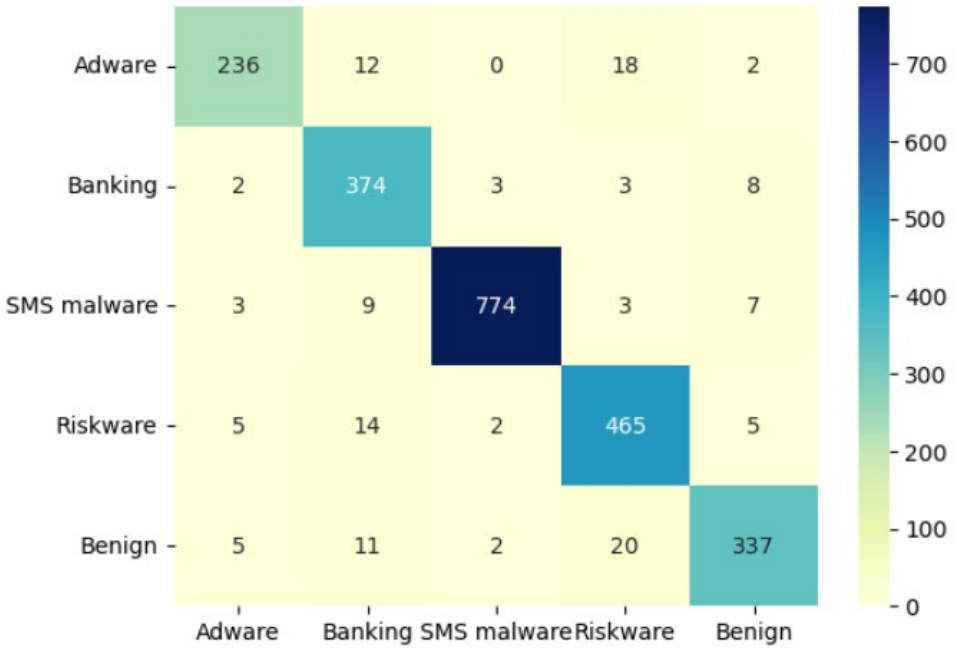

Рисунок 3.8 –Теплова карта для моделі випадкового лісу

Як бачимо, ця модель значно краще працює з виділення шкідливого ПЗ для всіх типів. На рисунку 3.9 наведено теплову карту для логістичної регресії, а на 3.10 – теплова карта для методу SVM. Як бачимо ці моделі теж значно краще працюють для визначення шкідливо ПЗ в порівнянні з моделлю Байєса

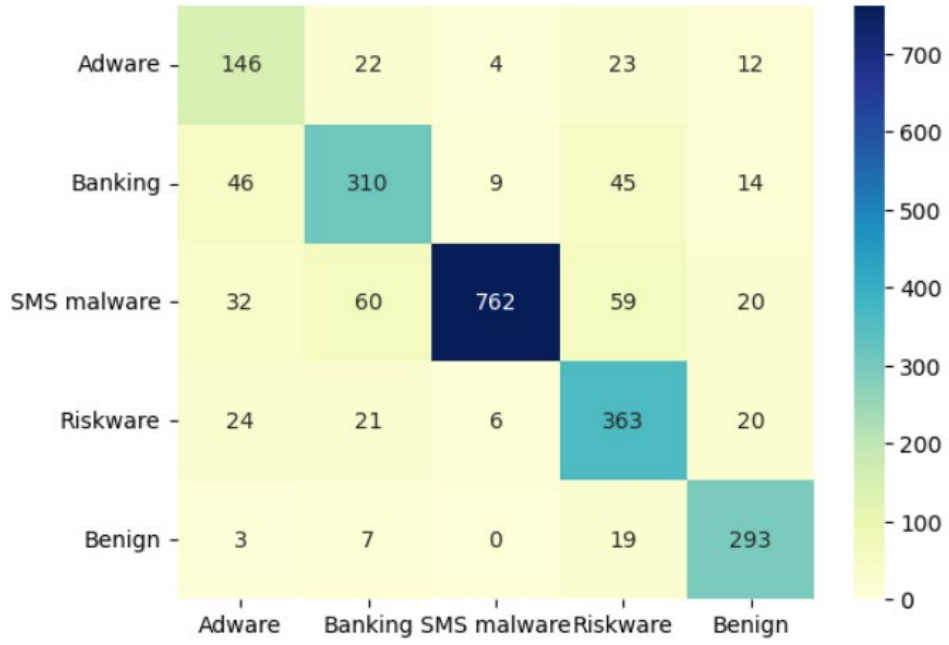

Рисунок 3.9 –Теплова карта для моделі логістичної регресії

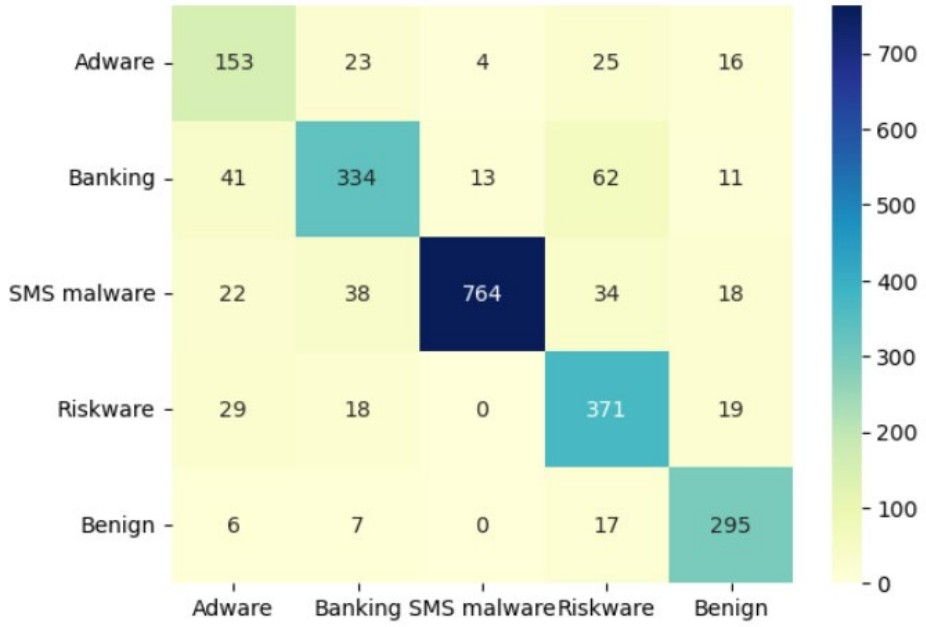

Рисунок 3.10 –Теплова карта для моделі методу опорних векторів (SVM)

Приведемо також теплову карту для методу kNN (рисунок 3.11)

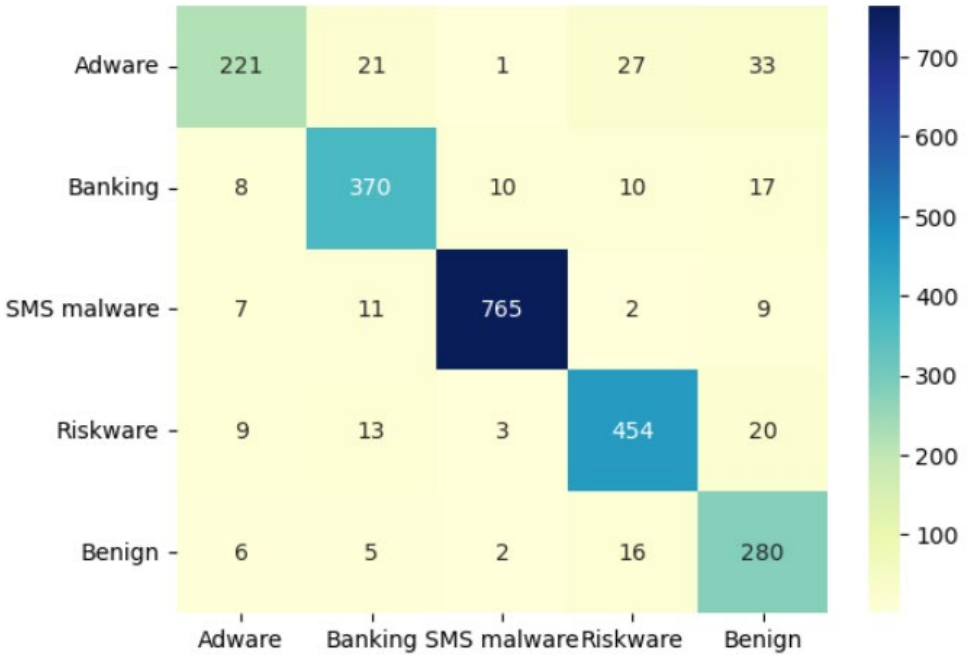

Рисунок 3.10 –Теплова карта для моделі методу k найближчих сусідів

Результати класифікації були узагальнені в таблиці 3.1 для різної кількості виділених ознак.

Таблиця 3.1 – Узагальнення результатів тестування для різної кількості виділених ознак

Кількість ознак	Моделі	Точність /Accuracy	Влучність/ Precision	Повнота/ Recall	F1
70	Модель Байєса	0.50	0.51	0.49	0.50
	Випадковий ліс	0.80	0.79	0.80	0.80
	Логістична регресія	0.72	0.72	0.72	0.72
	SVM	0.75	0.73	0.73	0.73
	kNN	0.81	0.80	0.80	0.80
120	Модель Байєса	0.59	0.69	0.59	0.55
	Випадковий ліс	0.94	0.94	0.94	0.94
	Логістична регресія	0.81	0.81	0.81	0.81
	SVM	0.83	0.83	0.83	0.83
	kNN	0.90	0.90	0.90	0.90
250	Модель Байєса	0.63	0.75	0.64	0.69
	Випадковий ліс	0.96	0.95	0.94	0.94
	Логістична регресія	0.84	0.84	0.84	0.84
	SVM	0.84	0.85	0.86	0.85
	kNN	0.91	0.91	0.91	0.91

Як бачимо модель Байєса не надто добре справилась з завданням виділення шкідливого програмного забезпечення. Як не дивно, доволі хорошу точність дав найпростіший метод k-найближчих сусідів. Модель випадкового лісу дала найкращий результат. За оптимальну кількість ознак в даній роботі обрано 120. Зменшення кількості ознак вдвічі порівняно з 250, не призвело до суттєвого падіння точності, проте збільшило продуктивність моделі в часі.

На рисунку 3.11 візуалізацію різних метрик точності для кількості ознак рівній 120.

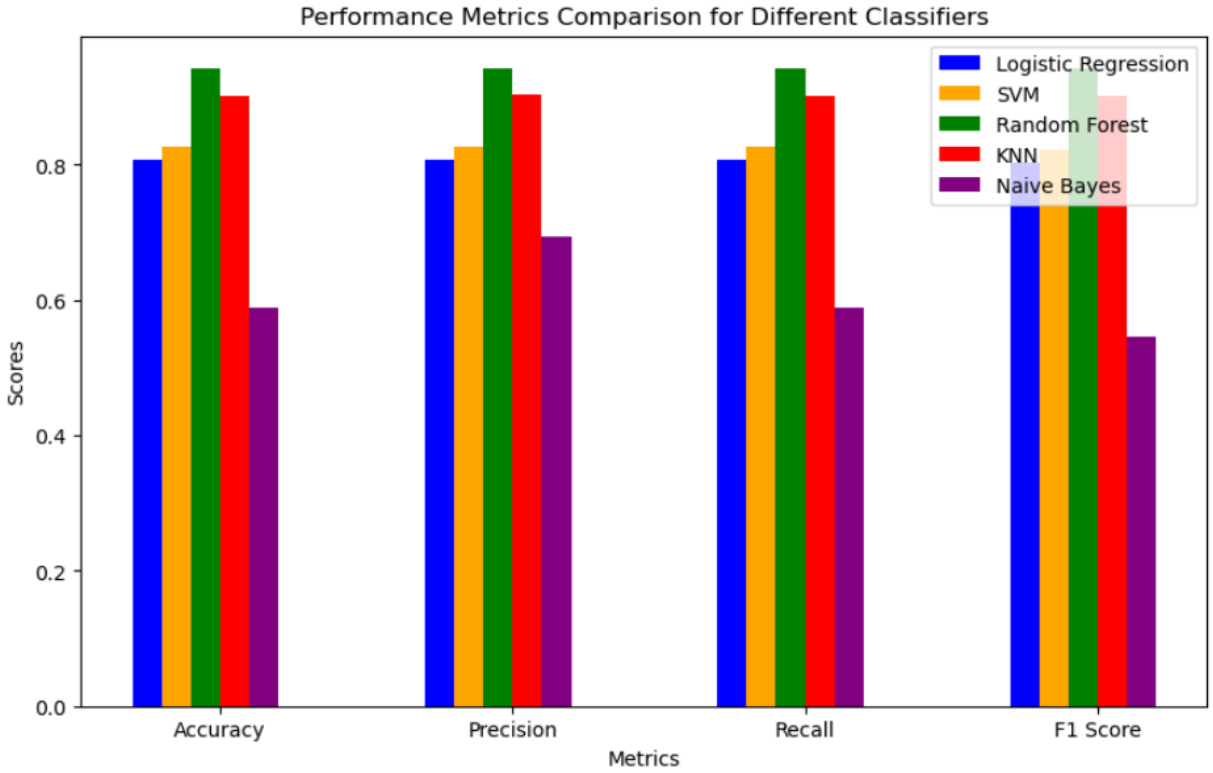

Рисунок 3.11 – Візуалізація метрик точності для різних моделей

З рисунку очевидно, що за всіма метриками метод випадкового лісу виявився найкращим, метод наївного Байєса найгіршим. Методи SVM та логістичної регресії виявилися приблизно одинаковими за всіма метриками. І несподівано метод k найближчих сусідів показав другий результат.

3.5 Висновок до 3 розділу

Отже в даному розділі обґрунтовано вибір програмного середовища, наведено характеристики обраного датасету, наведено основні функції для реалізації моделей, наведено результати експериментального дослідження щодо точності різних моделей в задачі виявлення шкідливо ПЗ та впливу кількості ознак на точність. Для подальших досліджень варто було б засобами бібліотеки Python провести підбір найкращих параметрів для кожної моделі, спробувати застосувати техніку ансамблювання для підвищення ефективності визначення шкідливого програмного забезпечення. Варто було б спробувати оцінити точність моделей для реальних об’єктів, а також застосувати крос-валідацію для підвищення достовірності результатів.

4 ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

У кваліфікаційній роботі магістра спроектовано систему для дослідження інтелектуальних методів для задач виявлення шкідливого програмного забезпечення на мобільній платформі Android.

Оскільки, проведення робіт з розробки та використання системи передбачає використання комп’ютерної техніки, зокрема ПК та периферійних пристроїв, то обов’язковим є дотримання вимог з охорони праці і техніки безпеки.

Для ефективної і безпечної роботи колективу працівників, в тому числі і фахівців з підвищення ефективності контролю доступу в приміщення, необхідно організувати безпечні умови праці. При цьому керівник організації несе безпосередню відповідальність за порушення нормативно-правових актів з охорони праці [26]. Окрім цього, на робочих місцях працівників необхідно забезпечити дотримання вимог, затверджених Наказом Мінсоцполітики від 14.02.2018 за № 207 «Про затвердження вимог щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями». Згідно вимог приміщення, де розміщені робочі місця операторів, крім приміщень, у яких розміщені робочі місця операторів великих ЕОМ загального призначення (сервер), мають бути оснащені системою автоматичної пожежної сигналізації відповідно до цих вимог:

- переліку однотипних за призначенням об'єктів, які підлягають обладнанню автоматичними установками пожежогасіння та пожежної сигналізації, затвердженого наказом Міністерства України з питань надзвичайних ситуацій та у справах захисту населення від наслідків Чорнобильської катастрофи від 22.08.2005 N 161, зареєстрованого в Міністерстві юстиції України 05.09.2005 за N 990/11270 (НАПБ Б.06.004-2005);
- Державних будівельних норм "Інженерне обладнання будинків і споруд. Пожежна автоматика будинків і споруд", затверджених наказом Держбуду

України від 28.10.98 N 247 (далі - ДБН В.2.5-56:2014, з димовими пожежними сповіщувачами та переносними вуглекислотними вогнегасниками.

В інших приміщеннях допускається встановлювати теплові пожежні сповіщувачі. Приміщення, де розміщені робочі місця операторів, мають бути оснащені вогнегасниками, кількість яких визначається згідно з вимогами ДСТУ 4297:2004 «Пожежна техніка. Технічне обслуговування вогнегасників». Загальні технічні вимоги і з урахуванням граничнодопустимих концентрацій вогнегасної рідини відповідно до вимог НАПБ А.01.001-2014. Приміщення, в яких розміщуються робочі місця операторів сервера загального призначення, обладнуються системою автоматичної пожежної сигналізації та засобами пожежогасіння відповідно до вимог ДБН В.2.5-56:2014, ДБН В.2.5-56:2010, НАПБ А.01.001-2014 і вимог нормативно-технічної та експлуатаційної документації виробника. Проходи до засобів пожежогасіння мають бути вільними.

Лінія електромережі для живлення комп’ютера та периферійних пристроїв повинні бути виконаними як окрема групова трипровідна мережа шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Не допускається використовувати нульовий робочий провідник як нульовий захисний провідник. Нульовий захисний провідник прокладається від стійки групового розподільного щита, розподільного пункту до розеток електроживлення. Не допускається підключати на щиті до одного контактного затискача нульовий робочий та нульовий захисний провідники.

Площа перерізу нульового робочого та нульового захисного провідника в груповій трипровідній мережі має бути не менше площі перерізу фазового провідника. Усі провідники мають відповідати номінальним параметрам мережі та навантаження, умовам навколишнього середовища, умовам розподілу провідників, температурному режиму та типам апаратури захисту, вимогам НПАОП 40.1-1.01-97.

У приміщенні, де одночасно експлуатуються понад п’ять комп’ютерів, на помітному, доступному місці встановлюється аварійний резервний вимикач,

який може повністю вимкнути електричне живлення приміщення, крім освітлення. Комп’ютери повинні підключатися до електромережі тільки за допомогою справних штепсельних з'єднань і електророзеток заводського виготовлення.

У штепсельних з'єднаннях та електророзетках, крім контактів фазового та нульового робочого провідників, мають бути спеціальні контакти для підключення нульового захисного провідника. Їхня конструкція має бути такою, щоб приєднання нульового захисного провідника відбувалося раніше, ніж приєднання фазового та нульового робочого провідників. Порядок роз’єднання при відключенні має бути зворотним. Не допускається підключати комп’ютери до звичайної двопровідної електромережі, в тому числі – з використанням перехідних пристроїв. Електромережі штепсельних з'єднань та електророзеток для живлення комп’ютерної техніки повинні бути виконаними за магістральною схемою, по 3-6 з'єднань або електророзеток в одному колі. Штепсельні з'єднання та електророзетки для напруги 12 В та 42 В за своєю конструкцією мають відрізнятися від штепсельних з'єднань для напруги 127 В та 220 В. Штепсельні з'єднання та електророзетки, розраховані на напругу 12 В та 42 В, мають візуально (за кольором) відрізнятися від кольору штепсельних з'єднань, розрахованих на напругу 127 В та 220 В.

При підвищенні ефективності контролю доступу в приміщення, де для забезпечення безпеки мешканців, співробітників і збереження майна використовуються ДС, важливим, з точки зору охорони праці, є забезпечення достатньої величини природного та штучного освітлення, які визначені у НПАОП 0.00-7.15-18. Організація робочого місця фахівця із дослідження методів та програмно-апаратних засобів оптимізаційних процесів на основі ГА повинна забезпечувати відповідність усіх елементів робочого місця та їх розташування ергономічним вимогам ДСТУ 8604:2015 «Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги». Відстань від екрана до ока фахівців, які працюють за комп’ютером визначається згідно з вимогами ДСанПіН 3.3.2.007-98.

Розміщення принтера або іншого пристрою введення-виведення інформації на робочому місці має забезпечувати добру видимість екрана комп'ютера, зручність ручного керування пристроєм введення-виведення інформації в зоні досяжності моторного поля згідно з вимогами ДСанПіН 3.3.2.007-98.

4.2 Безпека в надзвичайних ситуаціях

4.2.1 Структура системи БЖД

Поняття «життєдіяльність» стосується тільки людини. Людина живе і працює в безпосередньому зв'язку з навколишнім середовищем.

Життєдіяльність (ЖД) – це складна фізіологічна система, яка має назву «система ЖД».

Системою називають сукупність взаємозв'язаних елементів, функціонування яких спрямоване на досягнення певної загальної мети.

Система ЖД складається із взаємопов'язаних елементів: життя, діяльності людини, навколишнього середовища, – і має підтримувати комфортне та безпечне існування людини, забезпечити сталий розвиток людства.

Розглянемо характеристики елементів системи ЖД.

Життя – це форма існування матерії, яка характеризується обміном речовин, здатністю до розмноження і розвитку, вмінням пристосовуватись до навколишнього середовища.

Людина – вища форма розвитку живої матерії, і її існування – дуже складний процес, що не тільки підтримує її фізіологічний стан, але й задовольняє духовні потреби. Крім того, на життя людини суттєво впливають умови проживання та праці, медичний догляд і багато інших факторів, що виникають завдяки діяльності самих людей.

Діяльність – це специфічна форма ставлення людей до навколишнього середовища та одне до одного, яка має задовольняти потреби та інтереси людини. Це соціальна категорія, нерозривно зв'язана із суспільством. Тільки завдяки діяльності людини створено всі блага, які має людство.

Основні види діяльності такі:

- виробнича;
- наукова;
- мистецька;
- освітня.

Однією із специфічних форм діяльності людини є праця – перша й основна умова існування людини (людства).

Праця – цілеспрямована діяльність людини, у процесі якої вона впливає на природу і використовує її з метою виробництва матеріальних та інших благ, необхідних для задоволення своїх потреб.

Потреби – це необхідність для людини того, що забезпечує її існування і самозабезпечення (фізіологічне, матеріальне, соціальне, духовне та ін.).

Навколишнє середовище (довкілля) або середовище існування – це все, що оточує людину впродовж її життя. Навколишнє середовище, у свою чергу, поділяють на такі види:

- природне середовище;
- штучне середовище.

Природне середовище (біосфера) – це частина Землі і простору навколо неї, де зосереджено все живе. Біосфера включає:

- атмосферу (газоподібна частина);
- гідросферу (рідка водна частина);
- літосферу (тверда частина).

На ЖД людей найбільше впливає частина біосфери від поверхні Землі вглиб на 15–20 км і до висоти 20–22 км, де починається озоновий шар. Природне середовище є джерелом природних ресурсів для існування людини: повітря, води, деревини, корисних копалин, ґрунту та ін.

Штучне середовище – це складова довкілля, створена людством за тривалий час його існування. Штучне середовище умовно можна поділити на два види:

- виробниче середовище;
- побутове середовище.

Виробничим називають середовище, в якому людина реалізує свою трудову діяльність (підприємства, установи, навчальні заклади тощо).

Побутовим є середовище, де люди мешкають або проводять вільний час. Воно охоплює сукупність житлових будинків, комунально-побутових об’єктів, місця відпочинку та ін.

Організм людини може нормально функціонувати тільки тоді, коли умови (параметри) зовнішнього середовища відповідають оптимальним. Якщо умови середовища змінюються, стають несприятливими, то на протидію їм організм людини включає спеціальні механізми, які зберігають постійність параметрів внутрішнього середовища (всередині організму) чи змінюють їх у межах допустимого.

Можливість функціонування організму в середовищі, параметри якого постійно змінюються, забезпечується завдяки механізму, який називають адаптацією.

Адаптація (лат. adapto – пристосування) – динамічний процес пристосування організму до мінливих умов зовнішнього середовища, який спостерігається в будь-якому виді діяльності щоразу, коли виникають значні зміни в системі «людина – середовище». Адаптація може бути фізіологічною, психологічною, соціальною.

Отже, для функціонування системи ЖД середовище має обов’язково відповідати природним параметрам. Відхилення можливі в межах допустимого, коли організм людини здатний адаптуватися, захистити себе, підтримувати існування. Усе, що існує за цими межами, становить загрозу життю, тому виникає потреба захисту ЖД людей. Отже, безпека – важлива складова системи ЖД.

Розглядаючи систему ЖД як взаємодію людей з навколишнім середовищем, слід зауважити, що вона завжди підпорядкована певним принципам, правилам, умовам життя, природним умовам, традиціям тощо.

Система ЖД має такі характерні ознаки:

- її функціонування підпорядковане об’єктивним законам природи;
- це динамічна система, яка розвивається, удосконалюється, пристосовується до змін умов існування;

- тяжіє до сталого розвитку, вживаючи заходів захисту від упливу негативних факторів.

Основні принципи забезпечення ЖД такі:

- своєчасність, достатність, якість забезпечення людей необхідними для життя засобами високої якості і заходами в потрібний час у належній кількості;
- безпека ЖД (захист ЖД від упливу негативних факторів, що виникають унаслідок як природних явищ, так і діяльності людей).

Рівень реалізації цих принципів значною мірою залежить від способів забезпечення ЖД. Виходячи із сказаного, можна визначити такі головні способи забезпечення ЖД:

1. Організація ефективної трудової діяльності людей в суспільстві з максимальним залученням усіх ресурсів (створення робочих місць, упровадження високопродуктивного виробництва і технологій, нормування праці тощо).

2. Організація та удосконалення освіти і підготовка кадрів, розвиток науки відповідно до вимог часу.

3. Розвиток сфери послуг (комунальних, транспортних, торговельних, побутових і т. ін.).

4. Розширення мережі культурних, спортивних, розважальних установ.

5. Проведення заходів щодо збереження здоров’я людей (диспансеризація, оздоровлення, кваліфіковане медичне обслуговування і лікування, санітарно-епідеміологічний стан).

6. Розроблення законодавчих і нормативно-правових актів із забезпечення прав, свобод і захисту людей і суспільства в цілому.

Залежно від того, якою мірою реалізуються принципи та способи забезпечення ЖД, визначається рівень життя людей окремих країн і загальний розвиток людства.

4.2.2 Елементи теорії, що відповідають моделі безпеки життєдіяльності

Модель у широкому розумінні – це предмет, явище, система (опис, схема, знак, графік, план, макет та ін.), які за певних умов відіграють роль замінника або представника будь-якого іншого предмета, явища чи системи.

З точки зору науки модель – це матеріальна чи уявна система, що відображає чи імітує принципи внутрішньої організації, функціонування, певні властивості чи характеристики об'єкта дослідження, безпосереднє вивчення якого неможливе. Модель може замінити цей об'єкт у пізнавальному процесі з метою отримання нових знань про нього. Таким чином, відношення «модель—оригінал» не природне, а зумовлене процесом пізнання, і питання про їх співвідношення, ступінь їх подібності, адекватності – одне з найважливіших і найскладніших у процесі використання моделей у науковому пізнанні.

Сам процес моделювання – це непрямий, опосередкований метод наукового дослідження об'єктів пізнання на їх моделях, коли з певних причин безпосереднє їх вивчення неможливе.

Моделі в дисципліні «Безпека життєдіяльності» можна систематизувати за об'єктом зв'язків. Усі моделі можна умовно поділити на дві множини залежно від обсягу зв'язків, які вони демонструють.

Перша множина об'єднує моделі, що характеризуються структурою зв'язків.

Друга множина об'єднує моделі парних зв'язків. Певна умовність щодо цієї множини пов'язана з тим, що запровадження глибокого аналізу дозволяє уявити механізми реалізації цих зв'язків діючих великих систем.

Для характеристики довкілля на глобальному, державному і регіональному рівні використовують поняття структури зв'язків (на світовому рівні – навіть загальної). Відповідно до визначеної послідовності рівнів (за територією, від світового до регіонального) зменшується кількість таких зв'язків – з одного боку, а з іншого – збільшується рівень їх деталізації.

Під державним рівнем у цьому випадку розуміють сукупність діючих галузей виробництва як джерел забруднення і географічні чинники території, що одержує це забруднення. Відповідно до двох визначених рівнів подано моделі, що формують уявлення про стан світового довкілля і держави (на прикладі

сільськогосподарської галузі). На регіональному рівні модель, що формує стан довкілля, може бути представлена у вигляді взаємодій комплексу діючих (діючого) підприємств із середовищем виробництва.

Для визначення умов роботи підприємства найбільшу увагу для застосування привертають моделі, що відображають зв'язки:

- «регіональний природно-виробничий комплекс – середовище виробництва»;
- «виробниче підприємство – довкілля»;
- «виробниче середовище виробничого підприємства (середовище робочого місця) – людина».

Здобуття найбільш деталізованої інформації за взаємодії можливе на рівні парних (взаємодій) у вигляді: забруднювач середовища (джерелом є підприємство) – елемент довкілля. Таким чином, необхідно розробити відповідні моделі парної взаємодії.

До таких моделей (як зразок) належать:

- модель розповсюдження елемента забруднення в середовищі (елементи довкілля – атмосфера, гідросфера, літосфера);
- моделі обігу елемента забруднення в елементах довкілля;
- моделі обігу елементів середовища;
- моделі взаємних впливів на елементи довкілля;
- моделі взаємодій екологічних компонентів і організації екосистем;
- моделі впливів небезпечних і шкідливих чинників;
- моделі ієрархії екосистем та ін.

У рамках пари «виробниче середовище – людина» певний зміст взаємодій реалізується на базі спрощення уявлення «виробниче середовище» і представлення його як «технологічний процес, обладнання, види господарських робіт тощо».

В період виконання «технологічного процесу...» виникають небезпеки. Це може бути ініційовано як з боку «технологічного процесу, обладнання, видів господарських робіт», так і з боку – «людини». Виходячи з цього, у схемі розгляду нещасного випадку необхідно йти двома шляхами відносно:

- технологічного процесу, обладнання, видів господарських робіт та ін.;
- «людини» як джерела небезпек.

Розвиток подій вивчають за допомогою ступеневих логіко-імітаційних моделей. Характер ступеневої суті моделі визначає перехід від події до події. Події і переходи за змістом формуються трьома складовими: 1) технологічний процес, його операції й елементи; 2) конструкція обладнання; 3) стан охорони праці при їх взаємодії.

За наявності небезпечних обставин під час виконання будь-яких робіт людина сприяє, усвідомлює, приймає і реалізує відповідні рішення в послідовності.

Обидві моделі в межах поєднання свого змісту дають змогу усвідомити комплексний розвиток подій, причини аварій та ін., сприяють створенню безпечних умов праці і запобіганню травматизму.

4.3 Висновки до 4 розділу

Таким чином, у результаті аналізу вимог щодо охорони праці користувачів комп’ютерів, визначено особливості організації робочих місць, вимог з електробезпеки, природного та штучного освітлення для ефективної і безпечної роботи.

Також розглянуто питання структури системи БЖД, елементів теорії, що відповідають моделі безпеки життєдіяльності.

.

ВИСНОВКИ

Під час виконання кваліфікаційної роботи освітнього рівня «Магістр» було досліджено використання класифікаційних моделей машинного навчання для задач виявлення шкідливо програмного забезпечення на мобільних пристроях з ОС Android. Для дослідження використано найновіший набір даних CICMalDroid 2020.

Було досягнуто виконання наступних завдань:

- Проведено огляд проблеми виявлення шкідливого ПЗ, типів та реальних кейсів шкідливого ПЗ для мобільних додатків.
- Проаналізувано сучасні підходи до виявлення шкідливого ПЗ за допомогою машинного навчання.
- Проаналізовано доступні набори даних для навчання моделей виявлення шкідливого програмного забезпечення (ПЗ) та обрано один набір.
- Досліджено ефективність різних алгоритмів машинного навчання для задачі виявлення шкідливого програмного ПЗ
- Проведено експериментальну оцінку вибраних моделей машинного навчання на тестових даних та проведено їх порівняльний аналіз за ключовими метриками, такими як точність, повнота, F1-міра.
- Надано рекомендації щодо впровадження та вдосконалення запропонованого рішення.
- Розглянуто окремі питання з охорони праці і безпеки в надзвичайних ситуаціях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Mobile Operating System Market Share Worldwide. [Електронний ресурс]. URL: https://gs.statcounter.com/os-market-share/mobile/worldwide.
2. Share of malware attacks targeting mobile devices worldwide from 2021 to 2023 [Електронний ресурс] URL: https://www.statista.com/ statistics/1449739/malware-targeting-mobile-devices/
3. Tymoshchuk, D., & Yatskiv, V. (2024). Slowloris ddos detection and prevention in real-time. Collection of scientific papers «ΛΌГОΣ», (August 16, 2024; Oxford, UK), 171-176.
4. First Palm Virus Found. [Електронний ресурс]. URL: https://www.computerworld.com.au/article/78795/first_palm_virus_found/.
5. ZAGORODNA, N., STADNYK, M., LYPA, B., GAVRYLOV, M., & KOZAK, R. (2022). Network Attack Detection Using Machine Learning Methods. Challenges to national defence in contemporary geopolitical situation, 2022(1), 55-61.
6. Qamar, Attia & Karim, Ahmad & Chang, Victor. (2019). Mobile malware attacks: Review, taxonomy & future directions. Future Generation Computer Systems. 97. 10.1016/j.future.2019.03.007.
7. Lypa, B., Horyn, I., Zagorodna, N., Tymoshchuk, D., Lechachenko T., (2024). Comparison of feature extraction tools for network traffic data. CEUR Workshop Proceedings, 3896, pp. 1-11.
8. Kouliaridis, Vasileios & Barmpatsalou, Konstantia & Kambourakis, Georgios & Chen, Shuhong. (2020). A Survey on Mobile Malware Detection Techniques. IEICE Transactions on Information and Systems. E103-D. 204-211. 10.1587/transinf.2019INI0003.
9. Tymoshchuk, D., Yasniy, O., Mytnyk, M., Zagorodna, N., Tymoshchuk, V., (2024). Detection and classification of DDoS flooding attacks by machine learning methods. CEUR Workshop Proceedings, 3842, pp. 184 - 195.
10. William Enck, Machigar Ongtang, and Patrick McDaniel. 2009. On lightweight mobile phone application certification. In Proceedings of the 16th ACM

conference on Computer and communications security (CCS '09). Association for Computing Machinery, New York, NY, USA, 235–245. https://doi.org/10.1145/1653662.1653691

11. ТИМОЩУК, Д., ЯЦКІВ, В., ТИМОЩУК, В., & ЯЦКІВ, Н. (2024). INTERACTIVE CYBERSECURITY TRAINING SYSTEM BASED ON SIMULATION ENVIRONMENTS. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (4), 215-220.
12. Dimitrios Papamartzivanos, Dimitrios Damopoulos, and Georgios Kambourakis. 2014. A cloud-based architecture to crowdsource mobile app privacy leaks. In Proceedings of the 18th Panhellenic Conference on Informatics (PCI '14). Association for Computing Machinery, New York, NY, USA, 1–6. https://doi.org/10.1145/2645791.2645799
13. D.-J. Wu, C.-H. Mao, T.-E. Wei, H.-M. Lee, and K.-P. Wu, “Droid-Mat: Android Malware Detection through Manifest and API Calls Tracing,” AsiaJCIS, pp.62–69, 2012.
14. ТИМОЩУК, Д., & ЯЦКІВ, В. (2024). USING HYPERVISORS TO CREATE A CYBER POLYGON. MEASURING AND COMPUTING DEVICES IN TECHNOLOGICAL PROCESSES, (3), 52-56.
15. Yang, Z., et al. Appintent: Analyzing sensitive data transmission in android for privacy leakage detection. in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. 2013. ACM
16. Grace, M., et al. Riskranker: scalable and accurate zero-day android malware detection. In Proceedings of the 10th international conference on Mobile systems, applications, and services. 2012. ACM
17. Tymoshchuk, V., Mykhailovskyi, O., Dolinskyi, A., Orlovska, A., & Tymoshchuk, D. (2024). OPTIMISING IPS RULES FOR EFFECTIVE DETECTION OF MULTI-VECTOR DDOS ATTACKS. Матеріали конференцій МЦНД, (22.11. 2024; Біла Церква, Україна), 295-300.
18. Samra, A.A.A., K. Yim, and O.A. Ghanem. Analysis of clustering technique in android malware detection. in Innovative Mobile and Internet Services in

Ubiquitous Computing (IMIS), 2013 Seventh International Conference on. 2013. IEEE.

19. Egele, M., et al. An empirical study of cryptographic misuse in android applications. in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security. 2013. ACM.
20. Tymoshchuk, V., Vantsa, V., Karnaukhov, A., Orlovska, A., & Tymoshchuk, D. (2024). COMPARATIVE ANALYSIS OF INTRUSION DETECTION APPROACHES BASED ON SIGNATURES AND ANOMALIES. Матеріали конференцій МЦНД, (29.11. 2024; Житомир, Україна), 328-332.
21. da Costa, V.G., et al. Detecting mobile botnets through machine learning and system calls analysis.in Communications (ICC), 2017 IEEE International Conference on. 2017. IEEE.
22. Tymoshchuk, V., Vorona, M., Dolinskyi, A., Shymanska, V., & Tymoshchuk, D. (2024). SECURITY ONION PLATFORM AS A TOOL FOR DETECTING AND ANALYSING CYBER THREATS. Collection of scientific papers «ΛΌГОΣ», (December 13, 2024; Zurich, Switzerland), 232-237.
23. Zaman, M., et al. Malware detection in Android by network traffic analysis. in Networking Systems and Security (NSysS), 2015 International Conference on. 2015. IEEE.
24. Python vs R for Data Science [Електронний ресурс]. URL: https://www.stratascratch.com/blog/python-vs-r-for-data-science/
25. Detection of Potential Malware in Android Devices [Електронний ресурс]. URL: https://github.com/hasanccr92/CSE422-AI/blob/main/07_04.pdf
26. ДСН 3.3.6.042-99. Санітарні норми мікроклімату виробничих приміщень. [Електронний ресурс]. URL: https://zakon.rada.gov.ua/rada/show/va042282-99#Text
27. Методичний посібник для здобувачів освітнього ступеня «магістр» всіх спеціальностей денної та заочної (дистанційної) форм навчання «БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ» / В.С. Стручок – Тернопіль: ФОП Паляниця В.А., – 156 с. Отримано з https://elartu.tntu.edu.ua/handle/lib/39196.

Додаток А Публікація

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ
імені ІВАНА ПУЛЮЯ
НАУКОВЕ ТОВАРИСТВО ім. ШЕВЧЕНКА

М А Т Е Р І А Л И

XII НАУКОВО-ТЕХНІЧНОЇ КОНФЕРЕНЦІЇ

«ІНФОРМАЦІЙНІ МОДЕЛІ, СИСТЕМИ ТА ТЕХНОЛОГІЇ»

18-19 грудня 2024 року

ТЕРНОПІЛЬ
2024

УДК 004.056

Павло Луговський, студент гр.СБм-51, Ілля Фалендиш, студент гр. СБ-32

Тернопільський національний технічний університет імені Івана Пулюя

ВИБІР ІНФОРМАТИВНИХ ОЗНАК ДЛЯ ЗАДАЧІ ВИЯВЛЕННЯ ШКІДЛИВОГО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ В ANDROID

Pavlo Lugovskyi, student of gr. СБм-51, Illia Falendysh, student of gr.СБ-32

SELECTION OF INFORMATIVE FEATURES FOR THE TASK OF ANDROID MALWARE DETECTION

Шкідливе програмне забезпечення для Android - це будь-яке шкідливе програмне забезпечення або код, призначений для завдання шкоди мобільному пристрою користувача Android, наприклад, віруси, трояни, програми-вимагачі, рекламне, шпигунське та інше. Таке шкідливе програмне забезпечення може бути завантажене або встановлене з різних джерел. Після проникнення шкідливого програмного забезпечення на мобільний телефон користувача Android, воно може виконати різні шкідливі дії, такі як крадіжка та продаж конфіденційної інформації, шпигунство за активністю на телефоні (наприклад, текстовими повідомленнями, дзвінками тощо) або вимагання викупу.

Загалом методи виявлення шкідливого програмного забезпечення діляться на три великі групи: статичні, динамічні та гібридні [1]. Як показує дослідження [2], методи машинного навчання виявились дуже перспективними для виявлення шкідливого програмного забезпечення для Android, оскільки вони можуть розпізнавати складні шаблони даних і навчатися на великих масивах даних. Алгоритми машинного навчання ідеально підходять для виявлення шкідливого програмного забезпечення для Android.

Вибір інформативних і незалежних ознак є вирішальним елементом ефективних алгоритмів розпізнавання образів, класифікації та регресії, а ознаки, як правило, є числовими. У нашому дослідженні з виявлення шкідливого програмного забезпечення для Android за допомогою динамічного методу ML на рівні API (Application Programming Interface), інтерфейс прикладного програмування було обрано як ознаку для навчання моделей машинного навчання, які будуть використані для класифікації того чи іншого додатка Android як шкідливого програмного забезпечення чи ні. Наприклад, якщо під час роботи мобільного додатку було викликано один API, то для цього API буде зазначено значення один для ознаки, що стосується цього додатку, або ж буде надано значення нуль. Специфікація API - це детальний опис, який допомагає розробнику використовувати такий інтерфейс для виконання певних вимог або функцій.

Інженерія ознак полягає у визначенні ознак для кращого представлення цільової задачі, яку розв’язує машинне навчання, і подальшого підвищення його точності. Зазвичай для цього використовуються знання з певної галузі або автоматизовані методи для вилучення, вибору або побудови правильних ознак. Для цього сценарію використання для виявлення шкідливого програмного забезпечення на Android ключовим фактором досягнення високої точності є вибір ознак, який, по суті, є вибором API в підході.

Література:

1. Ashawa, Moses & Morris, Sarah. (2019). Analysis of Android Malware Detection Techniques: A Systematic Review. International Journal of Cyber-Security and Digital Forensics.No 8. P. 177-187. 10.17781/P002605.
2. Chowdhury, Naseef & Haque, Ahshanul & Soliman, Hamdy & Hossen, Mohammad Sahinur & Ahmed, Imtiaz & Fatima, Tanjim. (2023). Android Malware Detection using Machine learning: A Review. 10.36227/techrxiv.22580881.v1.