НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»
Навчально-науковий інститут прикладного системного аналізу
Кафедра системного проектування

До захисту допущено:

Завідувач кафедри

_____Вадим МУХІН

«___»_____2023 р.

Магістерська дисертація

на здобуття ступеня магістра

за освітньо-науковою програмою

“Інтелектуальні сервіс-орієнтовані розподілені обчислювання”

зі спеціальності 122 "Комп'ютерні науки"

на тему: «Аналіз ризиків в задачах інформаційної безпеки»

Виконав (-ла):
студент (-ка) ІІ курсу, групи ДА-21мп
Северин Максим Сергійович _____

Керівник:
д.т.н., професор
Мухін Вадим Євгенійович _____

Консультант з Розробки стартап проекту:
д.т.н., професор
Мухін Вадим Євгенійович _____

Рецензент:

_____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент (-ка) _____

Київ – 2023

Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Навчально-науковий інститут прикладного системного аналізу

Кафедра системного проектування

Рівень вищої освіти – другий (магістерський)

Спеціальність – 122 "Комп'ютерні науки"

Освітньо-професійна програма – "Інтелектуальні сервіс-орієнтовані розподілені обчислювання"

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____Вадим МУХІН
_____2023 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Северину Максиму Сергійовичу

1. Тема дисертації «Аналіз ризиків в задачах інформаційної безпеки», науковий керівник дисертації Мухін Вадим Євгенійович, доктор технічних наук, професор, затверджені наказом по університету від __ листопада 2023 р. № ____

2. Термін подання студентом дисертації – 25 грудня 2023 р.

3. Об’єкт дослідження “Методи та засоби аналізу ризиків порушення захищеності в комп’ютерних системах”

4. Вихідні дані: Алгоритми машинного навчання для виявлення ризиків порушення захищеності комп’ютерної системи на основі існуючих датасетів.

5. Перелік завдань, які потрібно розробити

   1. Порівняльний аналіз методів та засобів оцінки ризиків порушення захищеності КС.
   2. Розробка моделі виявлення та прогнозування ризиків порушення захищеності КС.
   3. Розробка програмного забезпечення механізмів виявлення та прогнозування ризиків порушення захищеності КС.
   4. Експериментальні дослідження результатів роботи механізмів виявлення та прогнозування ризиків порушення захищеності КС.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу

1. Електронна презентація

7. Орієнтовний перелік публікацій

1. Аналіз ризиків в задачах інформаційної безпеки. / Северин М.С., Мухін В.Є.// Системні науки та інформатика: збірник доповідей II науково-практичної конференції з нагоди 125-річчя КПІ ім. Ігоря Сікорського «Системні науки та інформатика», 4–8 грудня 2023 року, Київ. – К., НН ІПСА КПІ ім. Ігоря Сікорського, 2023. – с. 347-352.

8. Консультанти розділів дисертації[1]*

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання – 25 червня 2023 р.

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1.	Огляд літератури за темою магістерської дисертації	1 вересня 2023	
2.	Аналіз (огляд) існуючих рішень в області інформаційної безпеки	14 вересня 2023	
3.	Порівняльний аналіз методів та засобів оцінки ризиків порушення захищеності КС	5 жовтня 2023	
4.	Розробка моделі виявлення та прогнозування ризиків порушення захищеності КС	26 жовтня 2023	
5.	Розробка програмного забезпечення механізмів виявлення та прогнозування ризиків порушення захищенності КС	16 листопада 2023	
6.	Експериментальні дослідження результатів роботи механізмів виявлення та прогнозування ризиків порушення захищенності КС	7 грудня 2023	
7.	Аналіз отриманих результатів	14 грудня 2023	
9.	Захист дипломної роботи	25 грудня 2023	

Студент Максим Сергійович СЕВЕРИН

Науковий керівник Вадим Євгенійович МУХІН

_____

РЕФЕРАТ

магістерської дисертації Северина Максима Сергійовича на тему

«Аналіз ризиків в задачах інформаційної безпеки»

Робота виконана на 103 сторінках, містить 34 ілюстрації, 24 таблиці, 1 додаток. При підготовці використовувалась література з 20 джерел.

Актуальність теми. У сучасному світі, де зростає використання технологій та залежність від комп'ютерних систем, проблема кібербезпеки стає надзвичайно важливою. Забезпечення надійності та захищеності інформаційних систем від кіберзагроз є ключовим завданням. Магістерська дипломна робота присвячена аналізу ризиків у сфері кібербезпеки, що визначає її високу актуальність. У ній пропонується нова модель виявлення та прогнозування кіберзагроз. Цей внесок допоможе вдосконаленню стратегій виявлення та запобігання ризикам безпеки, що є дуже актуальним у будь-якій IT області.

Мета та задачі дослідження. Метою даної магістерської дипломної роботи є розробка моделі та програмного забезпечення виявлення та прогнозування ризиків порушення захищеності комп’ютерної системи.

Об’єкт досліджень. Існуючі моделі оцінки ризиків порушення захищеності комп’ютерної системи, набори даних для їх навчання та тестування, зокрема NSL-KDD датасет.

Предмет досліджень. Створення моделі та програмного забезпечення виявлення та прогнозування ризиків порушення захищеності комп’ютеної системи. Використання існуючих даних для тренування та тестування моделі, тестування моделі на реальних даних.

Методи досліджень. Для розробки алгоритму, представленого у даній магістерській дипломній роботі було використано алгоритми машинного навчання, методи обробки даних.

Наукова новизна. Науковою новизною є розробка ансамблевої моделі (stacking), що ґрунтується на новій комбінації методів класифікації машинного навчання, їх параметрів, а також методів обробки даних. Цей підхід відрізняється від існуючих моделей поєднанням supervised алгоритмів в модель другого рівня для різностороннього аналізу даних. Причому кожна модель в стекі відповідає за виявлення конкретного класу загрози, що значно підсилює якість і надійність результатів.

Потенційні застосування та практична цінність результатів дипломної роботи:

1. Системи виявлення вторгнень
2. Системи боротьби зі спамом
3. Засоби антивірусного захисту
4. Мережеві сканери

Публікації

1. Аналіз ризиків в задачах інформаційної безпеки. / Северин М.С., Мухін В.Є.// Системні науки та інформатика: збірник доповідей II науково-практичної конференції з нагоди 125-річчя КПІ ім. Ігоря Сікорського «Системні науки та інформатика», 4–8 грудня 2023 року, Київ. – К., НН ІПСА КПІ ім. Ігоря Сікорського, 2023. – с. 347-352.

Ключові слова: Інформаційна безпека, методи оцінки ризиків, різновиди кібератак, моделі виявлення і прогнозування, аналіз проблем безпеки комп’ютерних систем, штучний інтелект, машинне навчання.

ABSTRACT

Severyn Maxim Sergiyovich master's thesis on the topic

«Risk analysis in information security tasks»

The work is completed on 103 pages, contains 34 illustrations, 24 tables and 1 appendix. The literature from 20 sources was used in the thesis preparation.

Relevance of the topic. In today's world, where the use of technology and dependence on computer systems is growing, the issue of cybersecurity is becoming extremely important. Ensuring the reliability and security of information systems against cyber threats is a key task. The master's thesis is devoted to the analysis of risks in the field of cybersecurity, which determines its high relevance. It proposes a new model for detecting and predicting cyber threats. This contribution will help to improve strategies for detecting and preventing security risks, which is very relevant in any IT field.

The purpose and objectives of the study. The purpose of this master's thesis is to develop a model and software for detecting and predicting the risks of a computer system security breach.

Object of research. Existing models for assessing the risks of compromising the security of a computer system, data sets for training and testing them, in particular, the NSL-KDD dataset.

Subject of research. Creating a model and software for detecting and predicting the risks of a computer system security breach. Using existing data to train and test the model, testing the model on real data.

Research methods. Machine learning algorithms and data processing methods were used to develop the algorithm presented in this master's thesis.

Scientific novelty. The scientific novelty is the development of an ensemble model (stacking) based on a new combination of machine learning classification methods, their parameters, and data processing methods. This approach differs from existing models by combining supervised algorithms into a second-level model for comprehensive data analysis. Moreover, each model in the stack is responsible for detecting a specific class of threat, which significantly enhances the quality and reliability of the results.

Potential applications and practical value of the results of the thesis:

1. Intrusion detection systems
2. Systems for combating spam

3. Antivirus protection tools
4. Network scanners

Publications.

Severyn M.S., Mukhin V.E. // System sciences and informatics: collection of reports of the II scientific and practical conference on the occasion of the 125th anniversary of Igor Sikorsky Kyiv Polytechnic Institute "System sciences and informatics", December 4-8, 2023, Kyiv - K., ER IASA KPI, 2023. - pp. 347-352.

Keywords: Information security, risk assessment methods, types of cyberattacks, detection and prediction models, analysis of computer system security problems, artificial intelligence, machine learning.

ЗМІСТ

ВСТУП

В рамках сучасного цифрового світу, коли суспільство все більше стає залежним від інформаційних технологій, тема кібербезпеки набуває надзвичайної актуальності. Зростання кількості технологічних рішень, їх використання в усіх сферах діяльності організацій, банків, медичних установ та промислових підприємств призводить до зростання ймовірності кібератак, витоку конфіденційної інформації та порушень цілісності систем. Ці загрози можуть призвести до серйозних наслідків, таких як фінансові втрати, втрата довіри споживачів та загроза громадській безпеці.

Оцінка ризиків в контексті завдань інформаційної безпеки стає критично важливою для забезпечення захисту інформаційних систем в умовах постійно зростаючого кількісного та технологічного розвитку кіберзагроз. Цей процес належить до ключових складових стратегії протидії потенційним ризикам і включає ряд кроків для повного охоплення і аналізу можливих небезпек.

Дослідження в галузі аналізу ризиків в інформаційній безпеці є важливим для розробки ефективних стратегій захисту інформаційних систем. Це сприяє прогнозуванню, попередженню та зменшенню проблем, пов'язаних із зловживанням інформацією, забезпечуючи стійкість та надійність їх інфраструктури.

Загальні напрями досліджень у цій галузі включають розробку нових методів оцінки ризиків, покращення інструментальних засобів для аналізу та моделювання, вивчення тенденцій у сфері кібербезпеки і розробку стратегій захисту. Розуміння викликів та впровадження ефективних заходів безпеки інформаційних систем є важливими компонентами сучасного світу, який все більше стає цифровим і залежним від інформаційних технологій.

В контексті даної магістерської дисертації центральною метою є розробка нового підходу для аналізу ризиків в задачах інформаційної безпеки. Основна ідея полягає в створенні моделі, побудованої на принципі ансамблевого навчання

декількох вхідних моделей машинного навчання, розроблених для аналізу ризиків безпеки на основі аномалій. Завдяки вдало підібраній комбінації алгоритмів ML досягається краща точність та стійкість у порівнянні з одиночними алгоритмами.

Таким чином, дана магістерська дисертація робить значний внесок у сферу інформаційної безпеки, надаючи новий підхід для розробки та впровадження моделі, спрямованої на ефективний комп'ютерної системи на предмет загроз. Результати цього дослідження можуть мати значущий вплив на вирішення різноманітних завдань кібербезпеки у різноманітних галузях.

Створена модель може бути застосована у роботі з існуючим програмним забезпеченням, таким як системи виявлення вторгнень, засоби антивірусного захисту, системи боротьби зі спамом, мережеві сканери для виявлення та прогнозування ризиків та може слугувати основою для розробки нових стратегій захисту.

1 ПОРІВНЯЛЬНИЙ АНАЛІЗ МЕТОДІВ І ЗАСОБІВ ОЦІНКИ РИЗИКІВ ПОРУШЕННЯ ЗАХИЩЕНОСТІ КС

1.1 Аналіз проблем інформаційної безпеки КС

Комп'ютерна система (КС) - це складна інтегрована система, що включає в себе різноманітні апаратні та програмні компоненти, призначені для обробки та управління інформацією.

Безпека комп'ютерної системи - це стан захищеності КС і мереж від зловмисних атак, що можуть призвести до таких наслідків, як несанкціоноване розкриття інформації, крадіжка або пошкодження обладнання, даних або програмного забезпечення, а також до порушення або некоректного спрямування послуг, які вони надають. Ця сфера має важливе значення через зростаючу залежність від комп'ютерних систем, Інтернету та стандартів бездротових мереж, таких як Bluetooth та Wi-Fi. Крім того, це пов'язано зі зростанням кількості розумних гаджетів, включаючи смартфони, телевізори та інші пристрої, що входять до Інтернету речей (IoT). Кібербезпека є одним з найважливіших викликів сучасного світу, що пов'язано як зі складністю інформаційних систем, так і з суспільством, яке вони підтримують. Безпека має особливо велике значення для засобів, що керують великомасштабними системами з далекосяжними фізичними наслідками, такими як розподіл електроенергії, управління та фінанси.

Загроза - це потенційний ризик використання вразливості для порушення безпеки та ймовірного пошкодження/порушення інформації/сервісу, який зберігається/пропонується в комп'ютерних системах або через канали зв'язку. Загроза комп'ютерній системі виникає тоді, коли порушується конфіденційність (запобігання доступу до неї несанкціонованих осіб), цілісність (неможливість модифікації без дозволу) або доступність (доступність на вимогу уповноважених осіб) інформації в системі. Таким чином, до загроз комп'ютерним системам загалом можна віднести будь-яку навмисну, ненавмисну або спричинену природним

шляхом дію, яка призводить до втрати/маніпуляції з даними чи фізичного знищення апаратного забезпечення.

Нефізичні загрози націлені на дані та програмне забезпечення комп'ютерних систем шляхом пошкодження даних або використання помилок у програмному забезпеченні. Успішне використання помилок призводить до атак на безпеку комп'ютерних систем. Таким чином, загроза - це потенційна небезпека, спричинена вразливістю системи, а атака - це спроба несанкціонованої або шкідливої дії. Логічні загрози є основною причиною інцидентів безпеки комп'ютерних систем. Знання цих загроз та їхніх характеристик допомагає ідентифікувати їх та завчасно розробити кроки для захисту систем.

1.2 Порівняльний аналіз методів захисту інформації в КС

1.2.1 Традиційні методи захисту

Ризики для інформації та даних можуть виникати з різних джерел, включаючи зловмисників, віруси, спам, атаки на мережу та багато інших. Існує багато способів забезпечення інформаційної безпеки, і важливо розуміти їхні переваги та недоліки. Серед методів захисту інформації можна виділити традиційні:

— Паролі – швидкий, розповсюджений і відносно простий спосіб захисту, який не вимагає великих витрат часу. Вони можуть бути індивідуально налаштовані для користувачів і не зберігаються у системах у відкритому вигляді, а хешуються, що унеможливлює доступ до їх оригінального вигляду. Проте, паролі вразливі до вгадування та атак перебору, і вони потребують регулярної зміни для підвищення безпеки.

— Антивірусні програми – ПЗ, що допомагає виявити та блокувати віруси та інші шкідливі програми. Вони регулярно оновлюються для впровадження нових шаблонів загроз. Проте, не завжди вони ефективні проти нових видів загроз, і можуть споживати ресурси комп'ютера, сповільнюючи його роботу.

— Методи аутентифікації - вони допомагають визначити особистість

користувача, а авторизація визначає, які дії він може виконувати після входу. Проте, втрата паролів чи ключів доступу може суттєво загрожувати безпеці системи, і керування доступом на великій кількості ресурсів може стати складним завданням.

— Резервне копіювання - надійний спосіб відновлення інформації в разі втрати чи аварії. Однак створення та зберігання резервних копій може потребувати додаткових ресурсів та інфраструктури. Крім того даний спосіб за своїм визначенням не може застосовуватися для передбачення загрози, а лише для зменшення її наслідків.

— Фізичний захист - включає обмеження доступу до серверів і комп'ютерів, що допомагає захистити обладнання від фізичного доступу. Проте, це може вимагати додаткових обладнання та інфраструктури для забезпечення безпеки.

1.2.2 Сучасні техніки захисту

Оскільки вищезгадані способи не включають сучасних підходів захисту, вони можуть застосовуватися, як правило, звичайними користувачами, які не вимагають серйозного підходу до захисту своїх даних. Однак, коли мова заходить про великі компанії, комп’ютерні мережі, бази даних – використання сучасних технік стає необхідністю. До найпопулярніших можна віднести:

— Брандмауер або фаєрвол (firewall) - це система безпеки, яка служить бар'єром між внутрішньою мережею комп'ютерів і зовнішніми мережами, такими як Інтернет. Основна мета firewall полягає в контролі трафіку, який проходить через неї, та у запобіганні несанкціонованому доступу до внутрішньої мережі. Вони можуть використовуватися на різних рівнях мережі, включаючи мережевий рівень (мережеві брандмауери) та рівень застосунків (програмні брандмауери). Firewall аналізує інформацію про джерело та призначення пакетів за їхніми IP-адресами та портами. Він може мати правила, які вказують, які комбінації IP-адрес та портів дозволені чи заборонені. Крім того, адміністратор може встановлювати правила,

які визначають, який трафік має бути дозволений, а який блокований. Це може включати блокування конкретних IP-адрес або дозвіл лише на певні типи послуг. Становий аналіз (Stateful Inspection) аналізує стан з'єднань, відстежуючи комунікації між двома системами. Він дозволяє визначати, чи є конкретний пакет частиною легітимного, вже існуючоого з'єднання, що робить його безпечним для передачі.

— Шифрування даних є критичним елементом для захисту конфіденційності інформації, забезпечення безпеки передачі даних та уникнення несанкціонованого доступу. Цей процес включає в себе перетворення зрозумілої інформації (тексту чи даних) у криптовану форму за допомогою алгоритму шифрування. Основні різновиди включають:

- симетричне шифрування – такий тип шифрування, при якому один і той самий ключ використовується як для шифрування даних, так і для їх дерозшифрування. Прикладом можуть бути алгоритми DES (Data Encryption Standard), AES (Advanced Encryption Standard), Triple DES (3DES), як покращена версія DES. Симетричне шифрування ефективно застосовуєтья для захисту великих обсягів даних, але важливо враховувати безпеку обміну ключами між сторонами для уникнення несанкціонованого доступу.
- асиметричне шифрування, при якому використовуються два ключі: публічний і приватний. Кожен користувач має пару ключів: один для шифрування (публічний ключ) і інший для розшифрування (приватний ключ). Публічний ключ може бути розповсюджений відкрито, але приватний ключ повинен залишатися суцільною та відомою лише власнику. Прикладом може слугувати RSA (Rivest-Shamir-Adleman), що є асиметричним криптографічним алгоритмом, який використовується для шифрування і цифрового підпису. Асиметричне шифрування широко використовується для вирішення проблем обміну ключами, які виникають при симетричному шифруванні.
- хешування - процес перетворення вхідних даних (текстового рядка або будь-якого іншого об'єкта) в унікальний, фіксований розмір хеш-значення за допомогою хеш-функції. Сучасне хешування також включає

методи Key Strengthening (повторення процесу хешування кілька разів (ітерації) для ускладнення атак, спрямованих на визначення паролю) та Salt (Додавання випадкового рядка до паролю перед хешуванням для ускладнення атак " brute-force" або використання раніше вичислених хеш-значень). Хешування відіграє ключову роль у підтриманні безпеки та цілісності інформації, зокрема при зберіганні паролів, перевірці цілісності даних та створенні цифрових підписів. Правильне використання цього методу сприяє захисту конфіденційності та автентичності в інформаційних системах.

— Системи управління інцидентами та відновлення після інциденту (SIEM) – це комплексні рішення в галузі інформаційної безпеки, які спрямовані на моніторинг, аналіз та реагування на події в інформаційних системах. Збираючи та агрегуючи журнальні дані з різноманітних джерел, SIEM нормалізує та корелює ці дані для виявлення можливих загроз. Використовуючи аналітику та правила, система виявляє підозрілі події та реагує на них через сповіщення або автоматичні заходи. SIEM допомагає управляти інцидентами, включаючи категоризацію та вирішення, а також забезпечує зберігання журнальних даних для аналізу та аудиту. Це ключовий елемент стратегії кібербезпеки для виявлення та протидії загрозам і забезпечення відновлення після інцидентів.

Вищезгадані засоби є важливими складовими інформаційної безпеки. Однак навіть застосування сучасних технік само по собі недостатнє, оскільки загрози для безпеки постійно зростають і розвиваються. Традиційні методи можуть бути ефективними для звичайних користувачів, продвинуті – для всіх сучасних видів небезпек. Однак навіть вони не здатні вичерпно вберегти від усіх потенційних проблем. Тому, для забезпечення надійного рівня інформаційного захисту, необхідно проводити аналіз ризиків. Він дозволяє ідентифікувати потенційно вразливі місця, оцінити ймовірність небезпеки та її вплив на систему, а також передбачити конкретні види загроз. Усе це небхідне для розробки стратегій захисту.

1.3 Класифікація ризиків інформаційної безпеки

1.3.1 Принципи класифікації

Класифікація загроз в інформаційній безпеці важлива з кількох ключових поглядів. По-перше, вона дозволяє систематизувати та впорядковувати різноманітні загрози, що спрощує розуміння їх характеру та потенційних наслідків. По-друге, класифікація допомагає у визначенні пріоритетів та розробці ефективних стратегій захисту, орієнтованих на конкретні види загроз. Цей підхід сприяє оптимальному використанню ресурсів для запобігання або виявлення інцидентів безпеки.

Крім того, класифікація створює загальноприйняту мову та рамки спілкування в галузі кібербезпеки, що сприяє обміну інформацією та розробці спільних стратегій в боротьбі з загрозами.

При класифікації загроз інформаційній безпеці будемо дотримуватися наступних принципів:

— Взаємовиключність - кожна загроза, віднесена до однієї категорії, виключає всі інші, оскільки категорії не перетинаються. Кожен зразок повинен відповідати щонайбільше одній категорії.

— Вичерпність - категорії в класифікації повинні включати всі можливості (всі зразки загроз).

— Однозначність - всі категорії повинні бути чіткими і точними, щоб класифікація була однозначною. Кожна категорія повинна супроводжуватися однозначними класифікаційними критеріями, що визначають, які зразки слід віднести до цієї категорії.

— Прийнятність - всі категорії є логічними, інтуїтивно зрозумілими і легко сприймаються більшістю.

— Корисність - класифікація може бути використана для отримання уявлення про сферу дослідження; може бути адаптована до різних потреб застосування.

Ці принципи можна використовувати для оцінки класифікацій загроз. Хороша класифікація загроз повинна підтримувати більшість представлених принципів.

Розглянемо найбільш поширені класифікації загроз інформаційній безпеці. Можна виділити два основних типи:

— Методи класифікації на основі технік кібератак.

— Методи класифікації, що базуються на наслідках загроз.

1.3.2 Класифікація за техніками атак

1) Модель трьох ортогональних вимірів. Лукас Руф та співавтори запропонували нову модель загроз для класифікації загроз безпеці з метою покращення розуміння загроз і полегшення уже існуючих моделей. Вона вирішує проблему складності шляхом введення тривимірної моделі, яка поділяє простір загроз на підпростори відповідно до трьох ортогональних вимірів, позначених мотивацією, локалізацією та агентом:

— Агент загрози - це діяч, який накладає загрозу на певний актив системи, який представлений трьома класами: людина, технології та форс-мажорні обставини.

— Мотивація загрози представляє причину створення загрози і поділяється на два класи: навмисна та випадкова загроза.

— Локалізація загрози відображає походження загрози - внутрішнє або зовнішнє.

2) Гібридна модель для класифікації загроз. Запропонувана Сандро та співавторами гібридна модель має назву C3, так як розглядає три основні критерії:

— Частота загрози безпеці - показує частоту виникнення загрози безпеці.

— Область дії загрози безпеці - відображає сферу, на яку впливає загроза, наприклад, фізична безпека, безпека персоналу, безпека зв'язку та даних, а також операційна безпека.

— Джерело загрози безпеці - наводить типи джерел загрози.

3) Піраміда класифікації загроз інформаційній безпеці (Information Security

Threats Classification Pyramid) - метод класифікації навмисних загроз безпеці в гібридній моделі. Вона класифікує навмисні загрози на основі трьох факторів:

— Попередні знання зловмисників про систему – показник того, на скільки багато зловмисник знає про систему з точки зору системного обладнання, програмного забезпечення, знань співробітників і користувачів.

— Критичність області - відображає критичність частин системи, на які може вплинути загроза.

— Збитки - відображає всі збитки, які можуть бути завдані системі або організації (конфіденційність, цілісність).

1.3.3 Класифікація за впливом загрози

1) Модель STRIDE. Компанія Microsoft розробила метод класифікації під назвою STRIDE, який застосовується на рівні мережі, хоста, та додатків. STRIDE дозволяє характеризувати відомі загрози відповідно до цілей і завдань атак (або мотивації зловмисника). Абревіатура STRIDE утворена з першої літери кожної з наступних категорій: підробка особистих даних (Spoofing), фальсифікація даних (Tampering), заперечення (Repudiation), розкриття інформації (Information Disclosure), відмова в обслуговуванні (Denial of Service) та підвищення привілеїв (Elevation of Privilege). Це підхід, заснований на цілях, де робиться спроба проникнути в свідомість зловмисника, оцінюючи загрози за ступенем важливості.

2) Модель ISO Стандарт ISO (ISO 7498-2) перелічує п'ять основних впливів загроз безпеці та послуг як еталонну модель: знищення інформації та/або інших ресурсів, спотворення або модифікація інформації, крадіжка, вилучення або втрата інформації та/або інших ресурсів, розголошення інформації та переривання надання послуг.

Оскільки у даній роботі розглядається саме питання вразливості комп’ютерної системи, буде доцільним розглянути конкретні види кібератак.

1.3.4 Типи кібератак

До основних типів кібератак можна віднести:

— Атаки для збору інформації - це методика отримання зловмисником цінних даних. Ця не є явною атакою, отже, вона повністю пасивна.

— DoS (Denial of Service) атака — це тип кібератаки, що спрямований на перешкоджання чи обмеження доступу користувачів до ресурсів або послуг, які надає комп'ютерна мережа чи система. Одним із типів DoS атак є переповнення мережевого каналу або ресурсів сервера, наприклад, шляхом відправки великої кількості запитів на обслуговування. Спільні методи DoS атак включають SYN flood (атака, побудована на основі великої кількості неповних TCP-з'єднань), UDP flood (атака на основі великої кількості UDP-запитів), а також атаки, спрямовані на використання вразливостей в програмному забезпеченні.

— Розподілені DoS-атаки (DDoS) - жертву атакують одночасно з великої кількості окремих систем. DDoS-атаки зазвичай здійснюються за допомогою бот-мереж. Бот-майстер - це зловмисник, який опосередковано атакує машину-жертву, застосовуючи при цьому армію ботів. DDoS-атаки відбуваються, коли велика кількість скомпрометованих систем діє синхронно та координується під контролем зловмисника, щоб повністю виснажити її ресурси та змусити відмовити в обслуговуванні справжнім користувачам.

— Атаки на основі IoT. Проблемою Інтернету речей є його слабка захищеність, оскільки ці пристрої часто ігноруються при встановленні патчів безпеки, що створює лазівки для зловмисників. Це дозволяє їм захопити цільові пристрої та проникнути в мережу. Атака на основі IoT - це будь-яка кібератака, яка включає використання жертвою IoT для проникнення в мережу шкідливого програмного забезпечення.

— Програми-вимагачі - це шкідливе програмне забезпечення, яке перешкоджає доступу до комп'ютера або файлів на ньому. Комп'ютери можуть бути заблоковані, а файли зашифровані. Від жертви вимагають викуп за зняття обмежень, і це повідомлення відображається в системі жертви.

— Шпигунське та рекламне програмне забезпечення - це програми, які мають спільну властивість збирати особисту інформацію користувачів без їхнього відома. Шпигунське ПЗ включає клавіатурні шпигуни, які реєструють усе, що набирається на клавіатурі, і створюють високий ризик викрадення особистих даних.

— Спам - це небажані масові повідомлення електронної пошти, які дратують користувача надмірною кількістю небажаних листів. Спам створює навантаження для постачальників послуг зв'язку, організацій і приватних осіб. Також він використовується в якості інструмента для фішингу.

— Атаки на веб-сайти - націлені на компоненти браузера, які можуть бути незахищеними, навіть якщо вони встановлені в браузері. Атаки з використанням SQL-ін'єкцій спрямовані на будь-який веб-сайт або веб-додаток, що використовує базу даних SQL, таку як MySQL, Oracle тощо, шляхом використання недоліків безпеки в програмному забезпеченні додатку.

Даний список не є вичерпним, однак він чітко дає зрозуміти, що необхідно постійно впроваджувати нові методи захисту.

1.4 Оцінка ризиків інформаційної безпеки

1.4.1 Програмне забезпечення для аналізу КС

Програмним забезпеченням для оцінки ризиків будемо вважати те, яке здатне давати користувачу достовірну інформацію про стан його системи в контексті її захищеності або виконувати дії, спрямовані на підвищення самого захисту.

Для оцінки ризиків можна використати програмне забезпечення сканування певного простору комп’ютерної системи. Прикладом такого ПЗ можуть бути активні сканери вразливостей - програми або інструменти, які активно взаємодіють з цільовою системою чи мережею. Вони використовують активну аналітику, тобто відправляють запити чи спроби атаки, щоб виявити слабкі місця в системі. Nessus, OpenVAS, Qualys, Rapid7Nexpose, Metasploit – приклади таких сканерів.

Report: Results

ID: 97cc63d0-65d7-45ee-8ca8-711df1baa7dd
Modified:
Created:
Owner: admin

1 - 100 of 312

Vulnerability	Severity	QoD	Host	Location	Actions
rexec Passwordless / Unencrypted Cleartext Login	10.0 (High)	75%	192.168.11.137	512/tcp	
Samba End Of Life Detection	10.0 (High)	75%	192.168.11.137	445/tcp	
Samba 'TALLOC_FREE()' Function Remote Code Execution Vulnerability	10.0 (High)	75%	192.168.11.137	445/tcp	
PHP Multiple Vulnerabilities - Aug08	10.0 (High)	75%	192.168.11.137	80/tcp	
PHP Version < 5.2.7 Multiple Vulnerabilities	10.0 (High)	75%	192.168.11.137	80/tcp	
PHP End Of Life Detection (Linux)	10.0 (High)	75%	192.168.11.137	80/tcp	
MySQL End Of Life Detection (Linux)	10.0 (High)	75%	192.168.11.137	3306/tcp	
PostgreSQL End Of Life Detection (Linux)	10.0 (High)	75%	192.168.11.137	5432/tcp	

Рисунок 1.1. Приклад згенерованого звіту OpenVAS

Мережеві сканери - це інструменти, призначені для збору трафіку та даних про мережу у пристроях і службах комп'ютерних мереж. Не всі вони можуть самостійно аналізувати систему на предмет загроз, однак здатні надати інформації про неї на різних рівнях, що в подальшому може допомогти в аналізі. Прикладами можуть слугувати сканери Nmap, Angry IP Scanner, WireShark, Zenmap, NetCat, Zenmap.

Nmap Output | Ports / Hosts | Topology | Host Details | Scans

Port	Protocol	State	Service	Version
22	tcp	open	ssh	OpenSSH 4.3 (protocol 2.0)
25	tcp	closed	smtp	
53	tcp	open	domain	
70	tcp	closed	gopher	
80	tcp	open	http	Apache httpd 2.2.3 ((CentOS))
113	tcp	closed	auth	

Рисунок 1.2. Приклад роботи Nmap

Для сканування додатків можна застосовувати Veracode - популярний комерційний сервіс для виявлення та аналізу вразливостей в програмному забезпеченні. Ця платформа розроблена для допомоги організаціям виявляти потенційні вразливості в їх програмному коді, робити тестування на проникнення. Вона може інтегруватися з іншими інструментами розробки програмного забезпечення, такими як системи контролю версій, що допомагає розробникам

виправляти вразливості на ранніх стадіях розробки. Застосовується як у хмарному режимі, так і локально на серверах організацій.

Для оцінки ризиків у просторі Інтернету речей (IoT) використовується сканер Armis. Ця платформа інтегрується з широким спектром рішень для кібербезпеки інших виробників, наприклад, Palo Alto, Checkpoint, Cisco ISE, платформами Aruba ClearPass або ForeScout. Також Armis підтримує інтеграцію з корпоративними системами SIEM, зокрема Splunk.

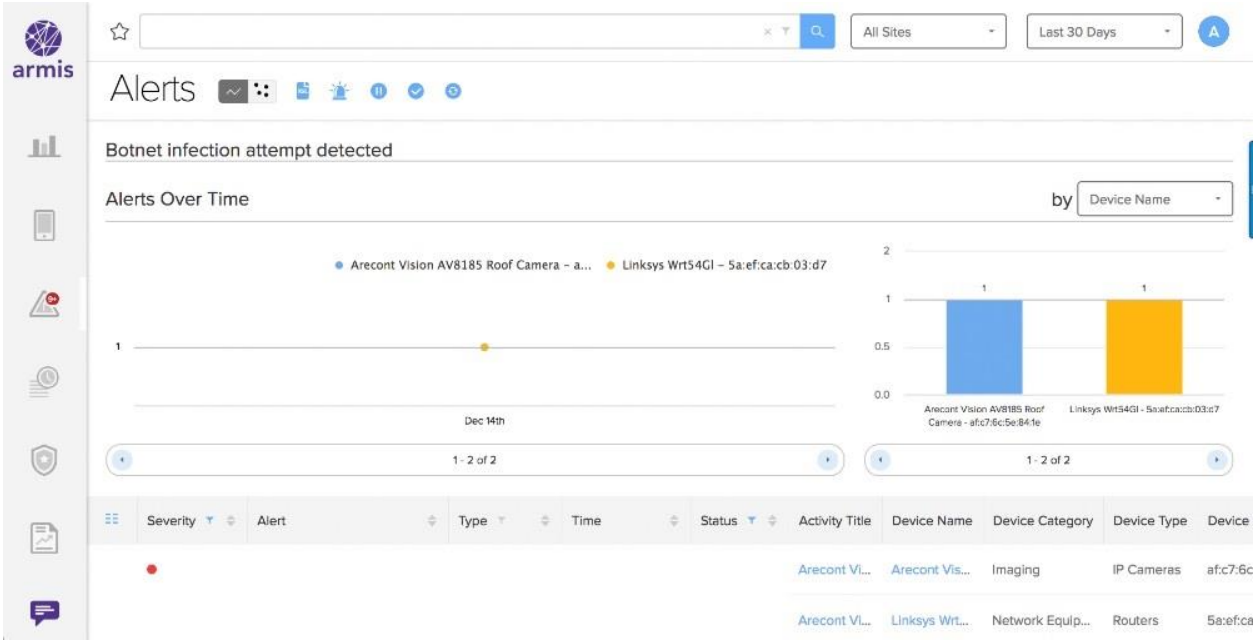

Рисунок 1.3. Вікно програми Armis

Багато сканерів побудовані на принципі алгоритмів евристичного сканування. Це означає, що під час сканування проводиться аналіз послідовності параметрів або команд у об'єкті, накопичується статистика та приймається рішення, такі як “вражений”, “ймовірно вражений”, “не вражений” і так для кожного об'єкта. Це значною мірою приклад ймовірнісного методу пошуку вразливостей, отже, його робота знаходиться під впливом законів теорії ймовірності. Як висновок, подібні методи далекі від стовідсоткового виявлення загроз.

1.4.2 Кількісна оцінка ризиків

Різноманітність можливих проблем приводить нас до питання цифрового опису, а саме кількісної оцінки можливих загроз. Потрібно також розуміти, що розробники захисного ПЗ вводять власні оцінки, а не дотримуються загальноприйнятих. Класичні оцінки зводяться до присвоєння ризику одного з 3 – 5 рівнів (відсутній, низький, середній, високий, критичний і подібні). Розглянемо найбільш прийнятний стандарт оцінювання – CVSS.

Загальна система оцінки вразливостей (Common Vulnerability Scoring System, CVSS) - це метод, який використовується для надання якісної оцінки серйозності. CVSS не є мірою ризику. CVSS складається з трьох метричних груп: Базові, часові та екологічні. Базові показники дають оцінку в діапазоні від 0 до 10, яка потім може бути змінена шляхом оцінки часових і екологічних показників. Оцінка CVSS також представлена у вигляді векторного рядка, стислого текстового представлення значень, що використовуються для отримання оцінки. Таким чином, CVSS добре підходить як стандартна система вимірювання для галузей, організацій та урядів, які потребують точних і послідовних оцінок серйозності вразливостей. Два найпоширеніші способи використання CVSS - це розрахунок серйозності вразливостей, виявлених в системах, і як фактор визначення пріоритетності заходів з усунення вразливостей. Національна база даних вразливостей (NVD) надає оцінки CVSS майже для всіх відомих вразливостей.

NVD підтримує обидва стандарти Common Vulnerability Scoring System (CVSS) v2.0 та v3.X. NVD надає "базові оцінки" CVSS, які представляють вроджені характеристики кожної вразливості. Наразі NVD не надає "часових оцінок" (метрики, які змінюються з часом через зовнішні по відношенню до вразливості події) або "оцінок середовища" (оцінок, налаштованих для відображення впливу вразливості на вашу організацію). Однак NVD надає калькулятор CVSS для CVSS v2 і v3, який дозволяє додавати дані часових і екологічних оцінок.

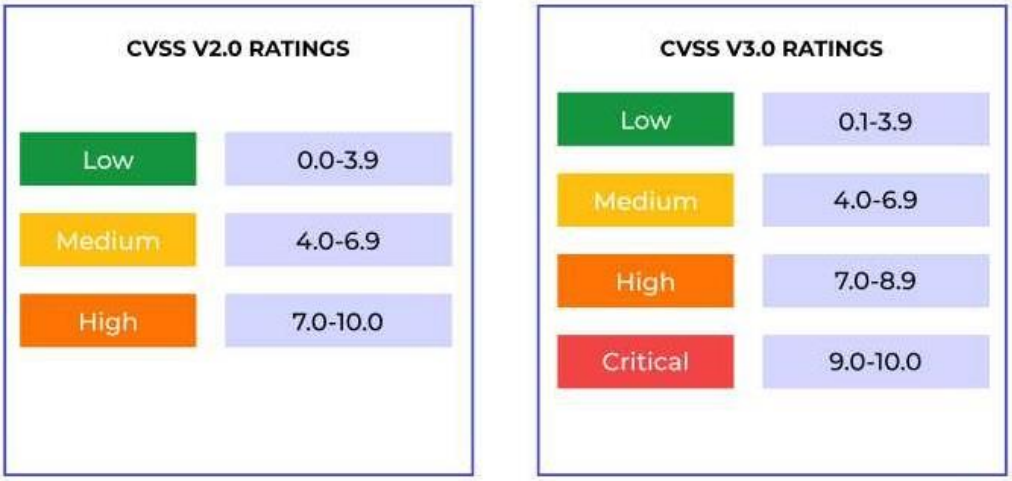

Рисунок 1.4. Оцінювання ризиків за шкалою CVSS [1]

CVSS оцінює уразливість з різних сторін:

— Якісна оцінка уразливості, яка не залежить від часу чи середовища, виражена базовими метриками:

- Access Vector (AV) показує, яким чином уразливість може бути використана.
- Access Complexity (AC) вказує, наскільки легко або складно використовувати дану уразливість.
- Authentication (Au) оцінює кількість аутентифікацій, які зловмисник повинен пройти, перш ніж використати уразливість.

— Вплив уразливості на комп'ютерну систему (Impact metrics):

- Confidentiality (C) описує вплив на конфіденційність даних, які обробляються системою.
- Integrity (I) описує вплив на цілісність даних комп'ютерної системи.
- Availability (A) описує вплив уразливості на доступність комп'ютерної системи. Наприклад, атаки, які впливають на пропускну спроможність мережі або використовують ресурси процесора, можуть впливати на доступність системи.

— Часові метрики (Temporal metrics), які враховують реакцію виробника уразливого продукту, і змінюються від моменту виявлення уразливості до моменту її виправлення:

- Exploitability (E) показує поточний стан методів використання уразливості, включаючи автоматизовані методи.
- Remediation Level (RL) є коригувальним числом, яке дозволяє покращувати тимчасову оцінку з часом, коли виправлення для уразливості стають доступними.
- Report confidence (RC) дозволяє виміряти рівень впевненості в існуванні уразливості та достовірності її технічних даних.

— Метрики уразливості, які враховують конкретні вимоги безпеки системи, в якій працює уразливий продукт (Environmental metrics):

- Collateral Damage Potential (CDP) оцінює потенційні збитки для компанії від даної уразливості, такі як можливі фізичні пошкодження обладнання тощо.
- Target Distribution (TD) оцінює частку уразливих систем в комп'ютерній мережі.
- Impact Subscore Modifier включає коригувальні числа для конфіденційності (CR), цілісності (IR) та доступності (AR), які дозволяють виправити Impact metrics та остаточну оцінку згідно з конкретними вимогами безпеки конкретного середовища.

1.5 Постановка задачі

Враховуючи велику і постійно зростаючу різноманітність ризиків кібербезпеки, стає необхідним розробити модель ПЗ, яка буде мати можливість виявляти та прогнозувати загрози в різних інформаційних середовищах. До вимог включимо наступні пункти:

— Адаптивність до обсягу даних – здатність ефективно працювати з великим обсягом даних, підтримка функціональності для обробки зростаючого потоку інформації.

— Точність — забезпечення високого рівня виявлення та прогнозування загроз.

— Швидкодія — здатність працювати за проводити аналіз за оптимальні проміжки часу, доки загроза не стала критичною.

— Можливість застосування в різних системах — модель можна використовувати у взаємодії з будь-яким відкритим для цього існуючим ПЗ.

Висновки до розділу 1

В цьому розділі було проведено порівняльний аналіз традиційних методів захисту інформації комп'ютерних систем. Під час розгляду загальних проблем інформаційної безпеки було визначено, що комп'ютерні системи потребують більш надійного захисту від потенційних загроз та ризиків. Виявлено, що у зв’язку з розвитком засобів ураження КС, з часом такі методи починають набувати більше недоліків ніж переваг.

Класифікація ризиків безпеки КС дозволила систематизувати потенційні загрози і їх характер, що є важливим етапом для подальшого аналізу та захисту інформації. Аналіз ризиків інформаційної безпеки виявився складним процесом, який передбачає врахування великої кількості факторів, таких як ймовірність впливу та характер загроз, який не завжди легко оцінити чисельно.

Важливою висновком є те, що традиційні засоби захисту інформації в комп'ютерних системах недостатні для ефективного виявлення ризиків. Враховуючи постійний розвиток технологій та зростання кількості загроз, необхідно використовувати сучасні моделі виявлення та прогнозування і модифіковувати їх залежно від потреб.

2 РОЗРОБКА МОДЕЛІ ВИЯВЛЕННЯ ТА ПРОГНОЗУВАННЯ РИЗИКІВ ПОРУШЕННЯ ЗАХИЩЕНОСТІ КС

2.1 Структури існуючих моделей виявлення та прогнозування ризиків порушення захищеності КС

Зловмисники разом з їх ПЗ розвиваються з кожним днем. Вони знають все про основні програми виявлення та знаходять способи їх обійти. Вони створюють нові шкідливі програми або нові версії вже застарілих, щоб провести успішну атаку, наприклад на підприємство, або його певний компонент. Рівень вразливості ніколи не опиститься до нуля, незалежно від того, велика це організація чи мала. Саме тому важливим кроком є вибір моделі для виявлення вразливостей. Розглянемо основні типи моделей для виявлення і прогнозування ризиків.

2.1.1 Виявлення на основі сигнатур

Сигнатури в контексті кібербезпеки є унікальними характеристиками або підписами, які ідентифікують конкретні об'єкти, такі як віруси, шкідливі програми або атаки. У випадку виявлення на основі сигнатур, це стосується характеристик відомих шкідливих програм.

Цей метод використовує базу даних сигнатур, що представляє собою унікальні характеристики відомих шкідливих програм. Кожен шкідливий файл має свою сигнатуру, яку можна ідентифікувати, порівнюючи з попередньо відомими. Процес виявлення можна розглядати наступним чином:

— Створення сигнатур: аналітики вивчають властивості відомих шкідливих програм і формують сигнатури, які можуть ідентифікувати ці програми. Це може бути хеш-сума файлу, конкретні байтові послідовності або інші характеристики.

— База даних сигнатур: створюється база даних, де зберігаються сигнатури відомих шкідливих програм. Ця база постійно оновлюється для включення нових

загроз.

— Сканування файлів: коли антивірус або система виявлення загроз отримує новий файл, вона порівнює його сигнатуру з базою даних. Якщо виявляється співпадіння, файл визнається як шкідливий.

— Дії при виявленні: якщо сигнатура визнана, система може призначити файлу категорію шкідливого, видалити його або взяти інші заходи для запобігання використанню.

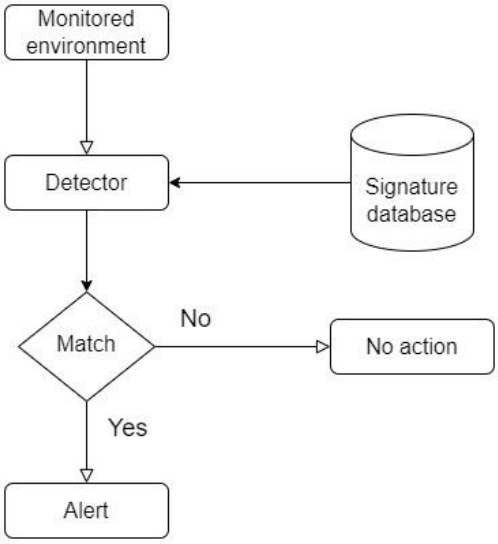

Рисунок 2.1. Загальна блок-схема методу на основі сигнатур

Механізми виявлення можуть використовувати хешування даних, при цьому вираховується хеш-сума файлу, що є унікальним ідентифікатором для визначення сигнатури. Також можливий варіант і через використання регулярних виразів. Тоді відбувається пошук в текстових частинах файлів на основі регулярних виразів, які збігаються з певними характеристиками шкідливих програм.

Потрібно враховувати, що база даних сигнатур повинна постійно оновлюватися, оскільки нові шкідливі програми з'являються неперервно. Крім того, недоліком даного методу можуть бути часті false-positive помилки: Іноді файл, який не є шкідливим, може мати схожість сигнатур і бути помилково визнаним як загроза.

Шкідливі програми можуть намагатися уникнути виявлення шляхом зміни своєї сигнатури або використання криптоантидетекторів. Виявлення на основі

сигнатур є ефективним для відомих загроз, але неефективним щодо нових або змінених шкідливих програм, що вимагає поєднання з іншими методами виявлення.

2.1.2 Виявлення на основі аналізу поведінки

Це метод, який фокусується на виявленні аномальної або підозрілої діяльності в системі, замість використання фіксованих сигнатур для ідентифікації конкретних загроз.

Програми такого типу аналізують дії користувачів та системи, реєструючи, які операції вони виконують, як вони взаємодіють з файлами та ресурсами, іншими користувачами тощо. Це може включати в себе створення "профілів" користувачів, що враховують їхні типові звички, рівень доступу та зв'язки, а також збір та аналіз інформації про стан та дії кінцевих точок (комп'ютерів, пристроїв) для виявлення аномальних паттернів.

Даний метод часто використовують при аналізі мережевого трафіку для виявлення незвичайних з'єднань, великої активності або інших аномалій. Також можливий розгляд інформації про метадані (наприклад, дати та часи дій, місцезнаходження, обсяг даних), щоб виявити непередбачувані або неприродні зміни.

Складність роботи включає в себе розрізнення між "нормальною" та "аномальною" поведінкою, а також визначення оптимальних порогів для сповіщень про підозрілі події. Для ефективного функціонування систем аналізу поведінки необхідно регулярно оновлювати базові дані і підбирати нові алгоритми розпізнавання аномалій. Більше того, встановлення параметрів “нормальної” поведінки може змінюватися зі зміною системи, чи ПЗ, яке досліджується, а зловмисники можуть адаптуватися під них.

2.1.3 Виявлення на основі машинного навчання (ML)

ML використовує великі структуровані дані для ідентифікації загроз та надання наочності аналізу безпеки. Наприклад, EDR (Endpoint Detection and Response), побудований на машинному навчанні, надає контроль над кінцевими точками, але для повноцінного аналізу потрібно поєднання з іншими програмами, такими як SIEM. Важливо, щоб модель надавала зрозумілі результати для подальшого розслідування аналітиками.

Очевидно, що всі методи мають свої переваги та обмеження. Виявлення на основі сигнатур ефективне проти відомих загроз, аналіз поведінки виявляє аномальні активності, але тільки за чітко визначеними правилами.

Машинне навчання здатне виявляти нові, раніше не розпізнані загрози на основі моделей, натренованих на великих об’ємах даних, що забезпечує їх глибокий аналіз. Отже, для оптимального захисту, модель виявлення і прогнозування варто побудувати на основі ML. Оптимальний захист може включати комбінацію цих методів для максимальної ефективності.

2.2 Проектування модифікованої моделі аналізу ризиків порушення захищеності КС

2.2.1 Принцип побудови моделей ML

Machine Learning (ML) - це різновид штучного інтелекту, який спеціалізується на розробці моделей, здатних навчатися на основі раніше зібраних даних і підвищувати свою точність без явного програмування. У ML алгоритми можуть обробляти великі обсяги даних, щоб виявляти релевантні закономірності, які використовуються під час навчання. Після початкового навчання, коли модель набуває знань з наявних даних, вона може робити прогнози для нових вхідних даних.

Загалом, кожен процес машинного навчання можна розділити на п'ять основних етапів:

1) Збір даних. Включає в себе збір та обробку даних, що використовуються для навчання моделі. Це може включати в себе очищення, перетворення та підготовку даних для подальшого використання.

2) Вибір моделі. У машинному навчанні існує багато підходів, спрямованих на пошук найбільш вдалого рішення для конкретної задачі. Залежно від характеру проблем безпеки, застосовуються різні моделі ML, як показано на рисунку 2.2.

3) Навчання моделі. На цьому етапі модель навчається на вхідних даних і виявляє закономірності та шаблони у даних. Цей етап включає в себе вибір відповідних налаштувань для параметрів моделі.

4) Валідація моделі. Після навчання моделі її ефективність оцінюється за допомогою валідаційних даних, які не використовувалися під час навчання. Це допомагає визначити, наскільки добре модель здатна робити прогнози.

5) Впровадження рішення. Якщо модель успішно випробована і пройшла всі необхідні етапи валідації та тестування, вона може бути впроваджена в реальному середовищі для роботи з реальними даними та здійснення прогнозів або прийняття рішень.

2.2.2 Ідея моделі ансамблевого навчання

Кожна конкретна модель ML не може бути абсолютно точною у передбаченні, більш того висока точність досягається лише при ідеально підібраних даних та методах. Це відбувається, бо кожен окремий алгоритм має свої недоліки, через що і застосовуються такі техніки, як fine-tuning (процес дотреновування моделі на конкретних даних для підбору кращих параметрів та підвищення точності). Fine-tuning відноситься до категорії методів transfer-learning (переносу навчання), що в цілому може стати складним процесом, адже залежить і від моделі, і від даних.

Тому, для використання більшої продуктивності алгоритмів ML використаємо метод ансамблевого навчання (Ensemble Learning). Цей підхід полягає в комбінуванні прогнозів кількох моделей для досягнення кращої загальної ефективності, ніж може бути досягнута окремими моделями (рис.2.2). Основна ідея заключається в тому, що об'єднання декількох слабких або базових моделей може призвести до потужної та більш точної прогностичної системи. Стійкість метода закладена у правилах об’єднання: завдяки налаштуванням можна гарантовано отримати вищу точність, ніж та, яка є максимальною серед усіх моделей.

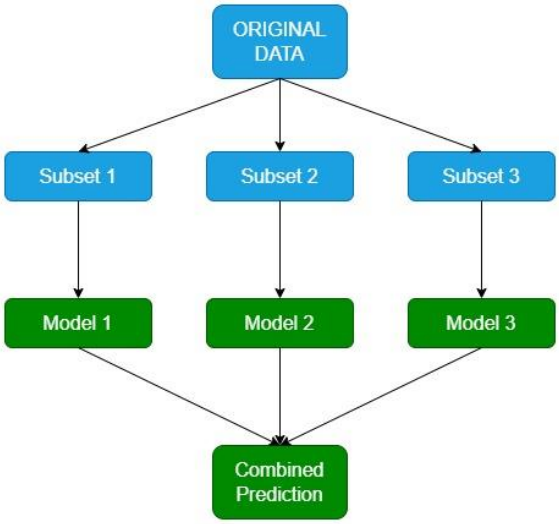

Рисунок 2.2. Загальна блок-схема ансамблевого навчання

До різновидів Ensemble learning включають:

— Voting (голосування) - кожна модель участі (базова модель) голосує за свій прогноз, і результат обчислюється на основі голосів. При Majority Voting класифікаційний результат обирається на основі класу, який отримав більше всього голосів. У Weighted Voting кожен голос може мати вагу в залежності від довіри, яка призначається відповіднії моделі. Наприклад, якщо одна модель вважається більш відомою або точною, її голос може мати більший ваговий коефіцієнт. При використанні Soft Voting замість простого голосування за клас, використовуються ймовірності, і результат обчислюється як середнє або зважене середнє ймовірностей.

— Bagging (багатомодельність) - використовується багато незалежних

моделей, які обчислюють прогнози. Потім вони комбінуються для зменшення перенавчання та покращення стійкості.

— Boosting (підсилення) – тренування моделей відбувається послідовно, при цьому кожна наступна модель намагається скоригувати помилки попередніх.

— Stacking (накопичення) – крім базових моделей (що лежать на першому рівні) використовується ще одна (мета-модель або мета-класифікатор), яка навчається на основі прогнозів базових моделей. Тобто, вихідні дані для мета-моделі - це прогнози базових моделей.

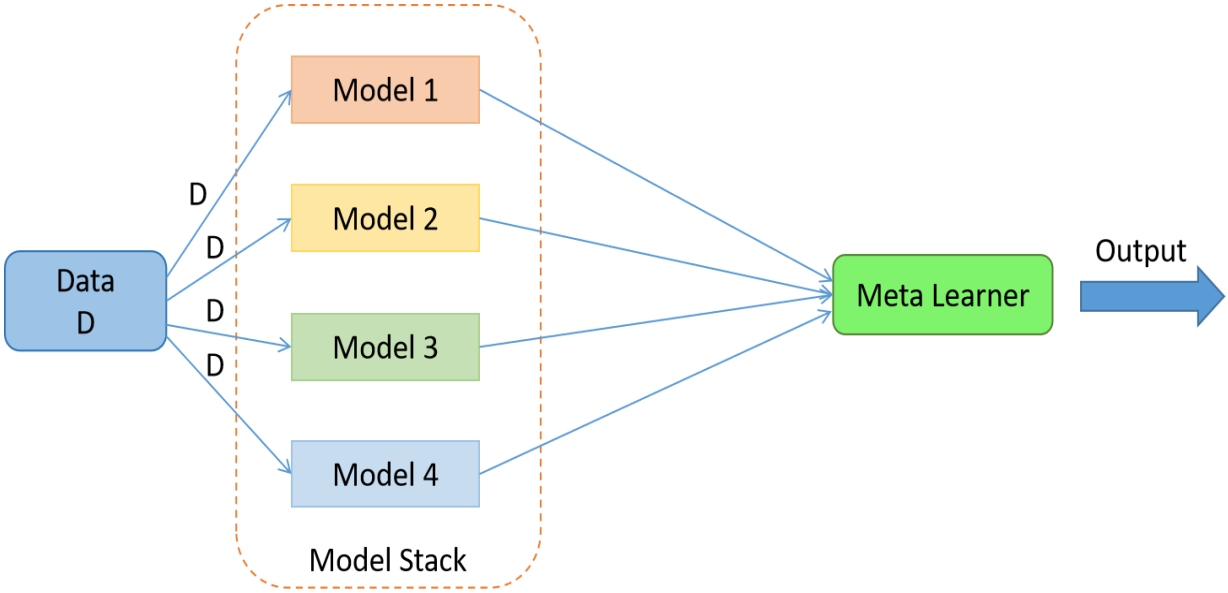

Рисунок 2.3. Принцип роботи Stacking моделей [2]

Навіть врахувавши певні часові витрати на навчання та деяку ймовірність перенавчанння, stacking моделі все одно показують кращий результат за рахунок своєї адаптивності. Завдяки можливості вибору мета-моделі та її окремого налаштування, якщо деякі моделі працюють краще на певних частинах даних, stacking може ефективно комбінувати їхні сильні сторони.

2.2.3 Формування стеку моделей для ансамблевого навчання

Для того, щоб побудувати дану модель, небхідно визначити набір базових моделей, їх параметри, а також мета-модель. У цей стек ми визначимо моделі, які

найкраще демонструють себе у тому числі в контексті аналізу ризиків безпеки.

У машинному навчанні алгоритми багатокласової класифікації (Multi-Class Classification) спрямовані на вирішення завдань класифікації екземплярів в один із трьох або більше вихідних класів. Під час вибору моделі використовуються популярні алгоритми класифікації для проведення якісної оцінки ризиків в галузі кібербезпеки. Такі моделі, базуючись на актуальних контекстних даних, можуть здійснювати точні та якісні прогнози оцінки ризиків та надавати можливість подальшого моніторингу інфраструктури організації, забезпечуючи безперервну оцінку на основі вхідних даних.

Саме такі моделі повинні бути використані для аналізу аномалій: так, отримуючи дані з комп'ютерної системи, модель зможе швидко виявити загрозу, класифікуючи її.

1) Дерева рішень. Дерево рішень (Decision Tree) - це алгоритм навчання з учителем (Supervised Learning), який використовується для класифікації, і він є частиною запропонованого рішення. Основна ідея полягає в тому, щоб розглянути особливості вхідного набору даних і класифікувати їх на основі певного параметра, відомого як інформаційний приріст. На першому етапі алгоритм ітерується над кожним стовпчиком ознак у вхідному наборі даних D, який містить історичні дані організації, і обчислює інформаційний приріст. Мета полягає в пошуку стовпця ознак, який має найбільший інформаційний приріст, і він стає коренем дерева рішень. Далі алгоритм розбиває набір даних на вузлі рішення і виконує той самий пошук для піднаборів даних. Таким чином, створюється деревоподібна структура, де кожен вузол представляє стовпець ознак, а листя вказує на вихідний клас.

Крім того, алгоритм дерева рішень є простим у використанні та зрозумілим методом класифікації. Він може навчатися на вхідних даних без значної попередньої обробки. В порівнянні з іншими алгоритмами класифікації, які використовуються в цьому підході, дерево рішень вимагає менше зусиль для підготовки даних і не потребує кроку нормалізації. Таким чином, модель, створена за допомогою дерева рішень, легко зрозуміла як технічним, так і нетехнічним користувачам. На рисунку 2.4 наведено візуальне представлення алгоритму дерева

рішень, навченого на згенерованому наборі даних. Для зроблення прогнозу за допомогою DT, новий зразок і проходить по дереву на основі кожного значення ознаки, а отримане значення листка вказує на вихідний клас.

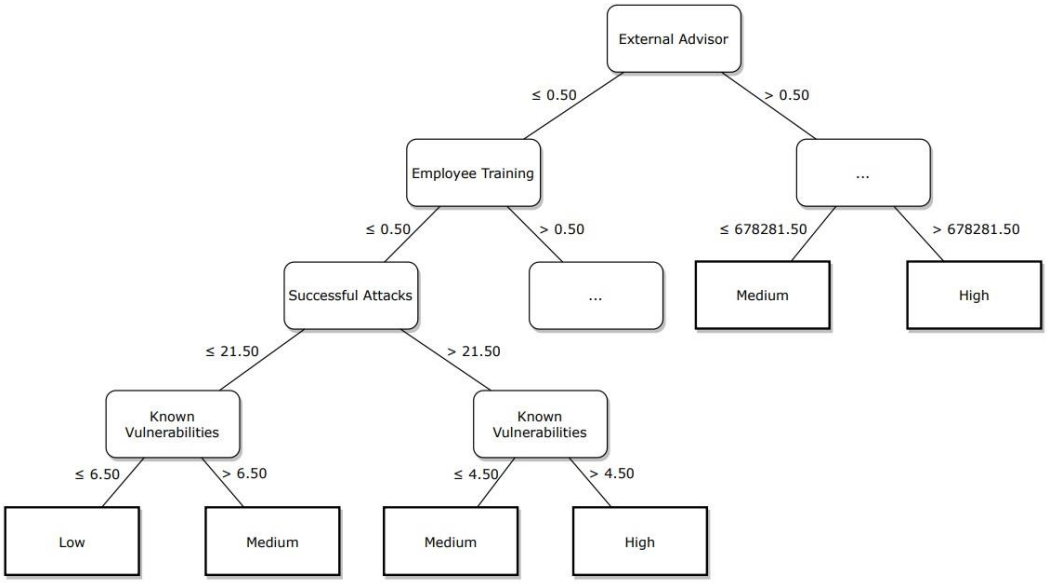

Рисунок 2.4. Візуалізація дерева рішень [3]

Випадковий ліс (Random Forest, RF) — це ансамбль алгоритмів, який використовує кілька дерев рішень для досягнення кращої ефективності в порівнянні з окремими деревами рішень.

Основна ідея випадкового лісу полягає в тому, щоб створити багато дерев рішень під час тренування і дозволити їм "голосувати" для вирішення задачі класифікації чи регресії. Процес формування кожного дерева рішень у випадковому лісі відрізняється від традиційного дерева рішень. Так при кожному розгляді розбиття вузла випадковим чином вибираються лише певна підмножина ознак для врахування, що допомагає зменшити кореляцію між деревами та робить модель менш вразливою до перенавчання.

Крім того, кожне дерево будується на випадковій підмножині тренувальних даних, яка створюється шляхом випадкового вибору прикладів з поверненням. Це робить кожне дерево унікальним та забезпечує більше різноманітності в ансамблі. При прийнятті рішення для конкретного прикладу, RF обчислює прогнози всіх дерев і вибирає результат, який отримав найбільше голосів.

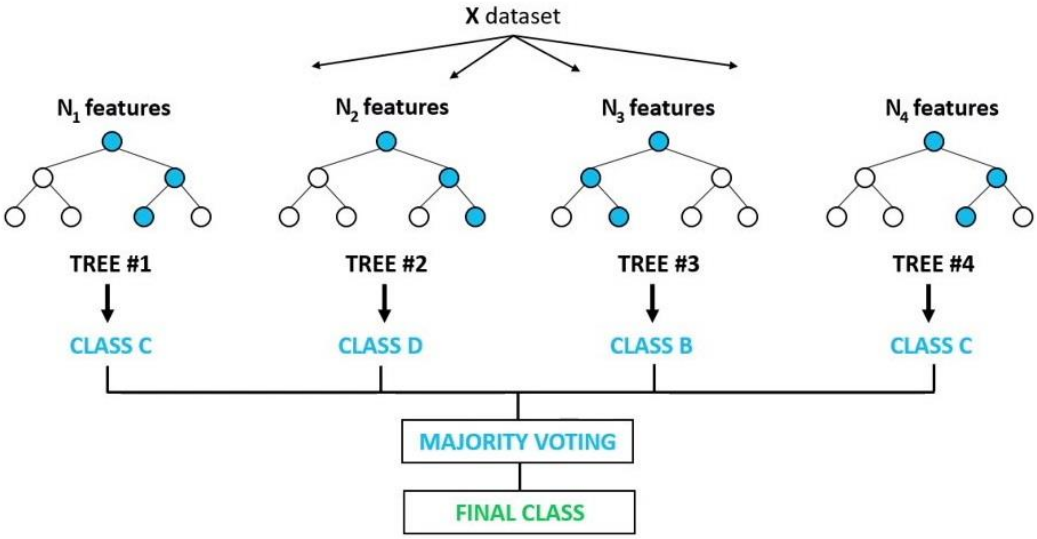

Рисунок 2.5. Візуалізація роботи Random Forest [4]

Ці особливості дозволяють випадковому лісу бути стійким до перенавчання та добре пристосовуватися до різних типів даних, забезпечуючи при цьому високу точність прогнозів.

2) K-Nearest neighbors - К-найближчих сусідів (KNN) - це ще один алгоритм навчання з учителем (Supervised Learning), який використовується для класифікації. Конкретно, KNN зазвичай називають класифікатором на основі екземплярів, оскільки основна ідея цього методу полягає в запам'ятовуванні вхідного набору даних для майбутніх прогнозів.

Алгоритм KNN вимагає три вхідні параметри: набір даних D, який містить інформацію, обрану кількість сусідів k та вибірку x, яку потрібно класифікувати. Далі алгоритм обчислює відстань між x та кожним записом у D. Ці обчислені відстані сортуються в порядку зростання, і вибираються k найближчих зразків, відомих як сусіди. Нарешті, передбачуваний клас для x (Classx) базується на схожості з сусідами, що означає, що x призначається після голосування більшістю голосів класів серед сусідів.

Іншими словами, KNN обчислює ймовірність належності вибірки x до певного класу на основі спостережень сусідів. В порівнянні з деревом рішень (DT), KNN вимагає більшої попередньої обробки даних.

З іншого боку, фаза навчання є швидшою, і нові навчальні дані можуть бути

додані без необхідності повної перетренування моделі. Рисунок 2.6 надає візуальне представлення класифікації KNN, де k дорівнює семи, а x - це новий зразок для класифікації. У цьому прикладі враховано лише два виміри. Після визначення k найближчих сусідів до x, з рисунку 2.6 видно, що прогнозований клас для x - "Низький", оскільки більшість сусідів належать до класу "Низький".

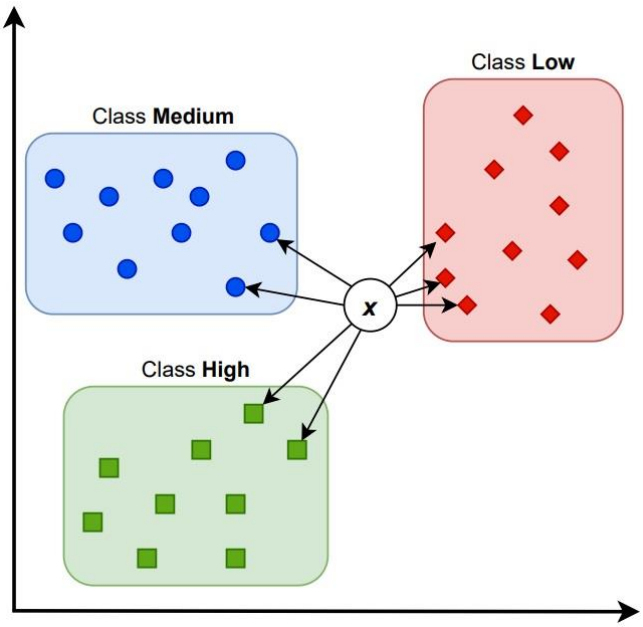

Рисунок 2.6. Візуалізація KNN алгоритма (k = 7) [3]

3) Gradient Boosting - це ефективний алгоритм навчання, що об'єднує кілька слабких моделей у сильні, у якому кожна нова модель навчається мінімізувати функцію втрат, таку як середньоквадратична помилка або перехресна ентропія попередньої моделі, використовуючи градієнтний спуск. На кожній ітерації обчислюється градієнт функції втрат по відношенню до прогнозів поточного ансамблю, після чого навчає нову слабку модель мінімізувати цей градієнт. Прогнози нової моделі додаються в ансамбль, а процес повторюється до тих пір, поки не буде досягнуто критерію зупинки.

Ваги навчальних екземплярів у цьому алгоритмі не змінюються, натомість кожен предиктор навчається, використовуючи залишкові помилки попередника як мітки. Є техніка, що називається Gradient Boosted Trees (Дерева з градієнтним підсиленням), базовим навчальним елементом якої є CART (Дерева класифікації та

регресії). На наведеній нижче блок-схемі показано, як навчаються дерева з градієнтним підсиленням для задач класифікації.

Варіант градієнтного бустинга, який буде використовуватись в даній моделі, навизається CatBoost (категоріальний бустинг) працює на основі градієнтного бустингу над рішеннями дерев прийняття рішень. Однією з його ключових особливостей є автоматична обробка категоріальних ознак. Він може працювати з категоріальними даними без конвертації їх в числові формати. Внутрішній механізм використовує ефективні методи кодування категорій для обробки цих ознак. CatBoost також вміє ефективно працювати з відсутніми значеннями, що є важливою рисою в реальних наборах даних, де можуть бути пропущені дані. А за рахунок введення додаткових обмежень і ваг для дерев, які зменшують їх вплив на модель CatBoost включає регуляризацію, яка допомагає уникнути перенавчання.

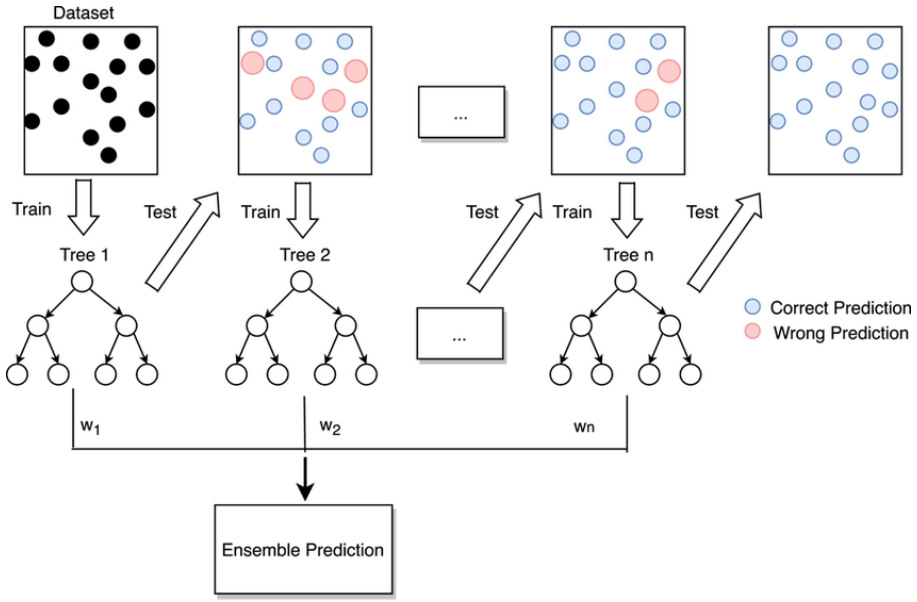

Рисунок 2.7. Діграма роботи градієнтного бустинга [3]

4) XGBoost - це оптимізована розподілена бібліотека градієнтного бустингу, призначена для ефективного та масштабованого навчання моделей машинного навчання. Це метод ансамблевого навчання, який об'єднує прогнози декількох слабких моделей для отримання сильнішого прогнозу. XGBoost розшифровується як "Extreme Gradient Boosting", він став одним з найпопулярніших і широко використовуваних алгоритмів машинного навчання

завдяки своїй здатності обробляти великі набори даних і досягати найсучаснішої продуктивності в багатьох завданнях машинного навчання, таких як класифікація і регресія.

Однією з ключових особливостей XGBoost є ефективна обробка пропущених значень, що дозволяє йому обробляти реальні дані з пропущеними значеннями без необхідності значної попередньої обробки. Крім того, XGBoost має вбудовану підтримку паралельної обробки, що дозволяє навчати моделі на великих наборах даних за розумний проміжок часу.

Для традиційного алгоритму radient Boosting Decision Tree (GBDT) використовується лише інформація про похідні першого порядку. Крім того, через залежність між слабкими учнями, GBDT важко навчати дані паралельно. XGBoost бере розклад Тейлора функції втрат до другого порядку і додає член регуляризації для пошуку оптимального рішення, який використовується для збалансування спаду цільової функції і складності моделі, щоб уникнути надмірного припасування. Модель XGBoost має такий вигляд:

$$y_i = a_0 + \sum_{k=1}^{K} (f(x_i)), f_k \in F \tag{1}$$

де K - кількість дерев рішень, f_k(xi) - функція входу в k-му дереві рішень, y_i - прогнозоване значення, а F - множина всіх можливих CART. Цільова функція XGBoost складається з двох частин: помилка навчання X та регуляризація θ.

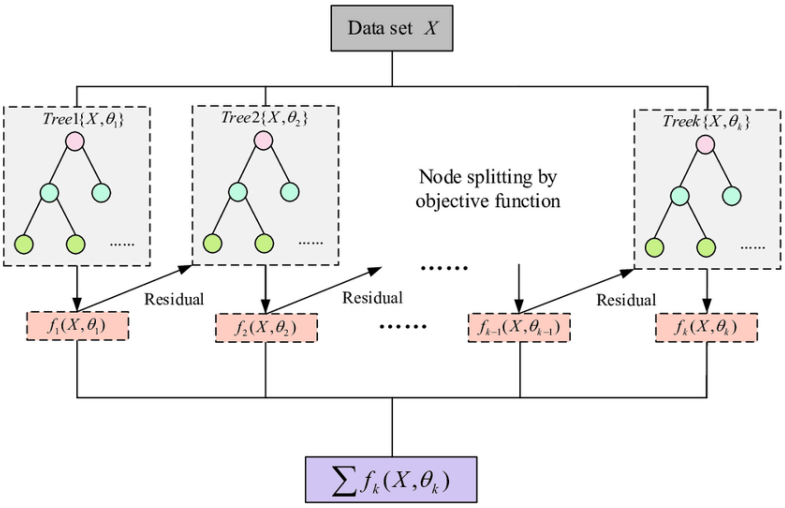

Рисунок 2.8. Діграма роботи XGBoost [3]

Отже, ми сформували стек із чотирьох ефективних моделей класифікації. Кожна з них увійшла до нього через свої особливі переваги. Random Forest стійкий до перенавчання та має високу ефективність на великих наборах даних. Модель k-Nearest Neighbors є простою та легко інтерпретованою, що робить її відмінним вибором для розуміння та аналізу результатів. Також вона пристосовується до форми даних, адже не передбачає жорстких припущень про розподіл даних. XGBoost відзначається високою точністю та швидкодією завдяки використанню градієнтного бустингу та оптимізованим структурам даних. Модель ефективно працює з різноманітними типами ознак та добре підходить для наборів даних із змішаними ознаками. CatBoost виділяється автоматичною обробкою категоріальних ознак, що полегшує підготовку даних та може покращити результати в задачах з категоріальними змінними. Алгоритм володіє вбудованим механізмом регуляризації, що забезпечує стійкість до перенавчання.

2.2.4 Формування метамоделі ансамблевого навчання

Метамодель в ансамблевому навчанні типу stacking є моделлю верхнього рівня, яка використовує передбачення базових моделей (моделей нижчого рівня) як вхідні дані для прийняття окремого рішення. Основна ідея полягає в тому, щоб використовувати передбачення базових моделей для створення нового прогнозу, який враховує внутрішні взаємодії та закономірності між ними.

Процес стекінгу можна розділити на декілька етапів:

1) Навчання базових моделей:
   - На початку виконується навчання базових моделей на наборі даних.
   - Кожна базова модель робить свої власні прогнози для вхідних даних.
2) Формування набору даних для метамоделі:
   - Прогнози базових моделей стають вхідними ознаками для метамоделі.
   - Окремо зберігається вектор цільової змінної (вихідні дані).
3) Навчання метамоделі:

- Метамодель навчається на новоутвореному наборі даних, використовуючи прогнози базових моделей та вихідні дані.
- В результаті навчання метамодель уявляє та враховує внутрішні залежності між передбаченнями базових моделей.

4) Прогнозування за допомогою стекінгу:
   - Після завершення навчання метамодель може використовуватися для здійснення прогнозів для нових вхідних даних.
   - Метамодель враховує внутрішні взаємодії та коригує передбачення базових моделей для досягнення кращої загальної точності.

Переваги стекінгу полягають у його здатності ефективно комбінувати сильні сторони різних моделей, покращуючи таким чином результати. Також, враховуючи внутрішні взаємодії, метамодель може виявити патерни, які можуть бути непоміченими окремими базовими моделями.

У якості метамоделі будемо використовувати логістичну регресію. Це метод для вирішення задач класифікації. Вона застосовує логістичну функцію для перетворення лінійної комбінації ознак у ймовірність віднесення до певного класу. Ймовірність потім порівнюється з певним пороговим значенням, і об'єкт класифікується в один із двох або більше класів (бінарна та мульти- класифікації).

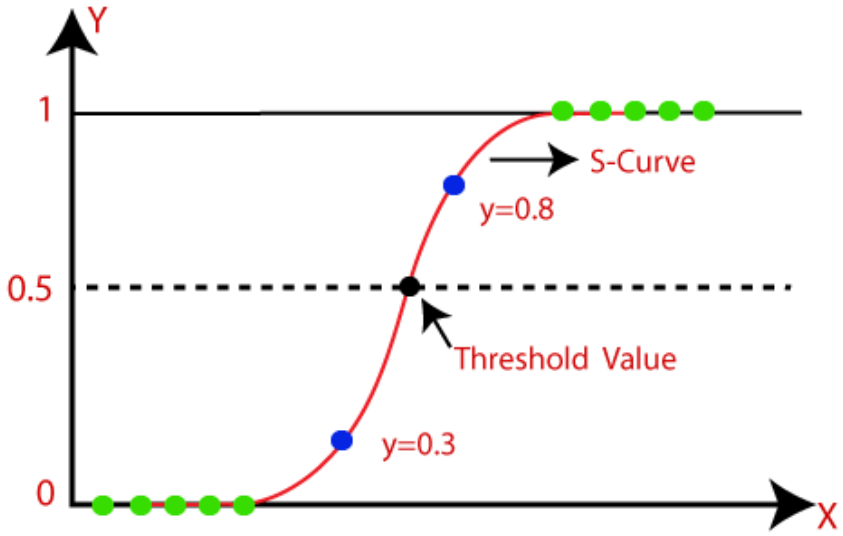

Рисунок 2.9. Візуалізація логістичної регресії [3]

У мультикласовій логістичній регресії (Softmax регресії) ми маємо кілька

класів, і ймовірність того, що зразок належить кожному з класів, обчислюється окремо. Основна формула для розрахунку ймовірностей у мультикласовій логістичній регресії виглядає наступним чином:

$$P(Y = k) = \frac{e^{b_{0k}+b_{1k}X_1+\cdots+b_{nk}X_n}}{\sum_{i=1}^{K} e^{b_{0i}+b_{1i}X_1+\cdots+b_{ni}X_n}} \quad (2)$$

де:

- $P(Y = k)$ - ймовірність того, що зразок належить класу k,
- K - загальна кількість класів,
- $b_{0k}, b_{1k}, \dots, b_{nk}$ - коефіцієнти для класу k,
- $X_1, \dots, X_n$ - ознаки зразка.

Мультикласова лінійна регресія є сильним кандидатом на роль мета-моделі, оскільки вона є добре вивченим, ефективним та широко використовуваним алгоритмом. Вона визначає ймовірність належності до класу як лінійну комбінацію вхідних ознак. Це дозволяє легко інтегрувати прогнози базових моделей, які вже мають складніші структури. Логістична регресія забезпечує інтерпретованість параметрів, що може бути важливим для аналізу вкладу кожної базової моделі та її впливу на метамодель, особливо якщо базові моделі різноманітні та різнорідні.

На рисунку 2.10 відображено принцип того, як моделі у стеку взаємодіють для прийняття рішення. На вхід подається набір визначених даних $\{x_1, \dots x_n\}$. Задача полягає у тому, щоб визначити, чи належить заданий об’єкт до категорії небезпечних. Для цього незалежно використовуються алгоритми RF, KNN, XGB, CB. При цьому вони можуть мати різні параметри для обробки даних (наприклад, застосовувати oversampling чи ні). Крім того, кожен алгоритм необов’язково має обробляти всі дані чи класифікувати усі можливі загрози. Завдяки тонкому налаштуванню, кожну модель можна навчити виявляти ризики лише певної категорії, що дозволить підвищити точність і зменшити час навчання.

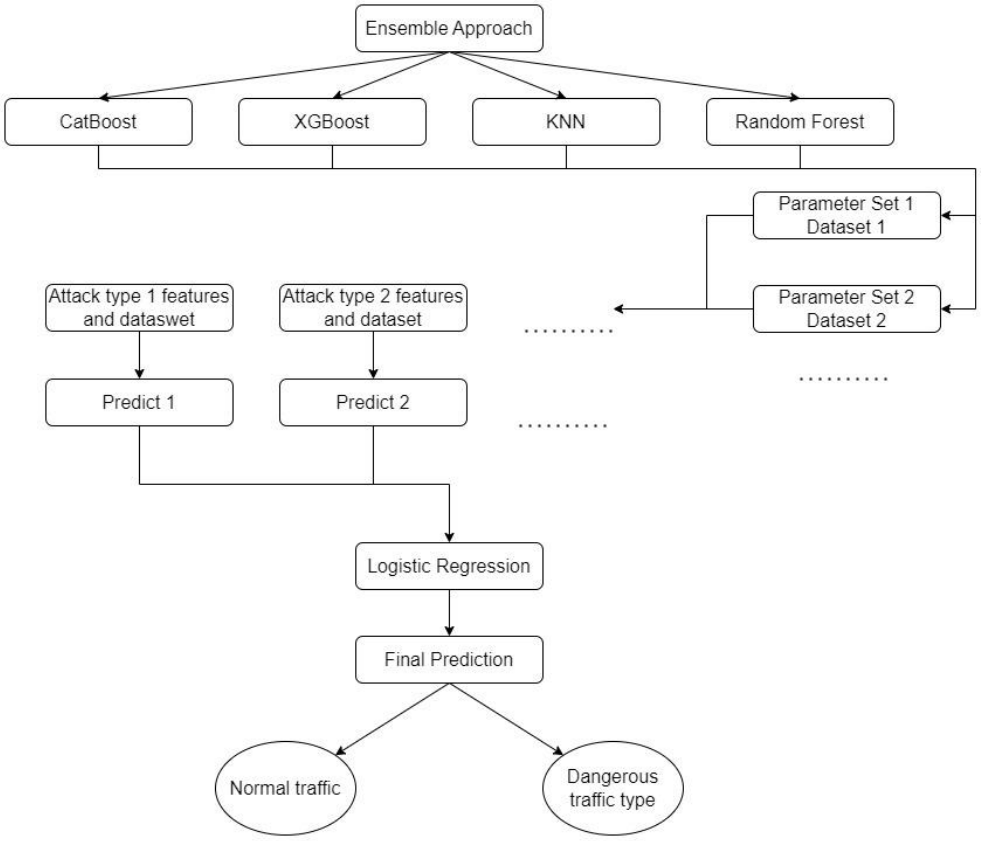

Рисунок 2.10. Модифікована модель аналізу ризиків

2.3 Оцінка параметрів розробленої моделі прогнозування ризиків

2.3.1 Теоретичний підхід оцінювання параметрів

В Ensemble Stacking оцінка параметрів метамоделі включає в себе налаштування параметрів, які зв'язані із зважуванням та комбінуванням прогнозів базових моделей. Ці параметри визначаються шляхом оптимізації функції втрат або функції правдоподібності на основі тренувальних даних.

Основна ідея в тому, що метамодель вивчає, як краще комбінувати прогнози базових моделей, з урахуванням їхньої різноманітності та різнорідності. Це може включати в себе вагові коефіцієнти для кожної базової моделі або навіть вивчення оптимального способу комбінування їхніх прогнозів.

Оцінка параметрів логістичної регресії у метамоделі стекінгу зазвичай виконується шляхом максимізації функції правдоподібності. Нехай у нас є K моделей у стекі (у вашому випадку, $K = 4$ для RF, KNN, CB, XGB). Позначимо ймовірність належності зразка до класу k для моделі i як P_{ik}, а вектор параметрів логістичної регресії для класу k як β_k.

Тоді функція правдоподібності для одного зразка має вигляд:

$$L(\beta) = \prod_{i=1}^{K} \prod_{k=1}^{C} P_{ik}^{y_{ik}} \tag{3}$$

де:

- C - кількість класів,
- y_{ik} - індикатор того, чи належить зразок i до класу k.

Функція правдоподібності максимізується шляхом знаходження оптимальних параметрів β. У зв'язку з тим, що максимізація добре здійснюється за допомогою логарифму, введемо логарифмічну функцію правдоподібності:

$$l(\beta) = \sum_{i=1}^{K} \sum_{k=1}^{C} y_{ik} \log(P_{ik}) \tag{4}$$

Тепер, для запобігання перенавчанню, можемо додати регуляризаційний член (зазвичай L2-регуляризацію) до цієї функції. Остаточний вигляд функції втрат для логістичної регресії виглядає наступним чином:

$$J(\beta) = -l(\beta) + \frac{\lambda}{2} \sum_{k=1}^{C} \|\beta_k\|^2 \tag{5}$$

де:

- λ - параметр регуляризації.

Оптимізацію цієї функції можна виконати за допомогою різноманітних методів, таких як градієнтний спуск чи алгоритми оптимізації другого порядку.

Отже, при використанні логістичної регресії як метамоделі у стекінгу, ми шукаємо набір параметрів β, які максимізують ймовірність належності зразка до

класу, використовуючи прогнози з K базових моделей.

2.3.2 Метрики оцінювання алгоритмів

Метрики – це характеристики, які визначають, наскільки ефективною є модель в розв'язанні конкретної задачі. Зазвичай вони використовуються для оцінки якості класифікації. Розглянемо кожну з метрик і їх зв'язок із зазначеними параметрами.

1) Accuracy (точність) - це відношення правильно класифікованих зразків до загальної кількості зразків.

$$Accuracy = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{6}$$

де TP - true positive, TN - true negative, FP - false positive, FN - false negative. Це усі можливі результати класифікації.

2) F1Score – це середнє гармонічне між precision і recall.

$$F1 = \frac{2 \cdot \text{ precision recall}}{\text{precision+recall}} \tag{7}$$

3) Precision (влучність) - це відношення правильно класифікованих позитивних зразків (TP) до всіх позитивних зразків (TP + FP).

$$Precision = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{8}$$

4) Recall (повнота) - це відношення правильно класифікованих позитивних зразків (TP) до всіх актуальних позитивних зразків (TP + FN).

$$Recall = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{9}$$

5) Характеристика "ROC AUC" (Receiver Operating Characteristic Area Under the Curve) визначається як площа під кривою характеристики роботи приймача (Receiver Operating Characteristic, ROC). Ця крива відображає залежність чутливості (true positive rate) від специфічності (1 - false positive rate) для різних значень порогу в класифікації.

$$ROC\ AUC\ = \int TPR\ d\ FPR \tag{10}$$

ROC AUC вимірюється в діапазоні від 0 до 1, значення 1 вказує на ідеальну модель, яка має 100% чутливість і 100% специфічність, тоді як значення 0.5 вказує на модель, яка працює як випадковий вибір.

Висновки до розділу 2

В данному розділі було зпроектовано модифіковану модель для виявлення і прогнозування ризиків безпеки комп'ютерних систем на основі методу Ensemble Stacking. Для цього були використані чотири базові моделі: Random Forest (rf), k-Nearest Neighbors (knn), CatBoost (cb) та XGBoost (xgb), які були об'єднані мета-моделлю.

Мета-модель у вигляді логістичної регресії була вибрана для об'єднання прогнозів базових моделей та покращення їхньої загальної ефективності. Цей підхід дозволяє використовувати переваги кожного окремого алгоритма та компенсувати їхні недоліки, формуючи більш точний та стійкий прогноз.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ МЕХАНІЗМІВ ВИЯВЛЕННЯ ТА ПРОГНОЗУВАННЯ РИЗИКІВ ПОРУШЕННЯ ЗАХИЩЕНОСТІ КС

3.1 Розробка програмного забезпечення механізмів виявлення ризиків

На даному етапі можна приступити до розробки ПЗ виявлення загроз. Для цього важливо розуміти механізми, які потрібно реалізувати. Виявлення загрози полягає у здатності програми розпізнавати серед даних або команд ті, які можуть завдати шкоди системі. Прогнозування – здатність описати знайдену загрозу, тобто виявити, який це тип атаки, звідки здійснений, які у нього характеристики.

Механізми виявлення мають працювати в межах комп'ютерної системи (КС). Однак, комп'ютерна система – широке поняття, воно включає усе можливе апаратне і програмне забезпечення конкретного пристрою. Розроблюване ПЗ має покрити значну частину КС, щоб можна було говорити про високий рівень захисту. Тому якщо навчати модель на відповідних даних, її можна застосувати і для аналізу повідомлень на спам, і для виявлення атак на веб-сайти, і навіть для перевірки коду програм на предмет загроз.

Найкращим варіантом для дослідження КС на предмет ризиків безпеки є система виявлення інцидентів (Intrusion Detection System) - це складна програма або пристрій, призначений для моніторингу та аналізу активності в інформаційній системі чи мережі з метою виявлення можливих інцидентів безпеки (рис.3.1).

Такі системи аналізують мережевий трафік (Network IDS), порівнюють активність з нормою (ADS, Active Detectory Security) та збирають результати аналізу подій в системі (EDS, Endpoint Detection System). Збирати дані можна як з усієї мережі, так і з окремого вузла. Host-Based IDS працює на рівні застосунків (рівень 7 моделі OSI), враховуючи конкретні аспекти програм та служб на хості. Network-Based IDS робить аналіз мережевого трафіку, що проходить через

мережеві пристрої, такі як маршрутизатори чи комутатори. Працює на рівні мережевого трафіку (рівень 3 OSI) або на рівні мережевого з'єднання (рівень 2).

Звідси можна зробити висновок, що таке ПЗ може застосовувати усі моделі виявлення небезпек, згадані у другому розділі.

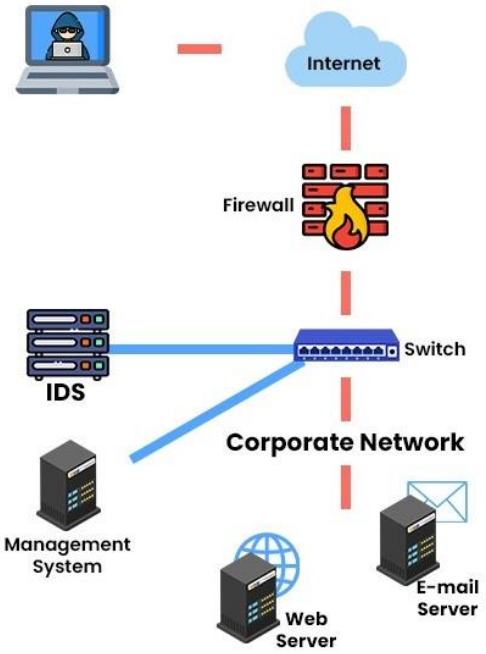

Рисунок 3.1. Мережа з IDS [9]

Для побудови аналогічної програми на основі спроектованої моделі скористаємось мовою програмування Python та її бібліотеками. Дані для аналізу ми будемо отримувати з pcap-файлів (Packet Capture). Це формат зберігання захопленого мережевого трафіку.

1) Спочатку захопимо трафік з мережі для аналізу. Для цього використаємо бібліотеку scapy.all, а саме функцію sniff() та її параметрами:

- prn – дозволяє задати функцію з правилами захоплення трафіку
- timeout – час сканування
- lfilter=lambda x: x.value('x1') – фільтрація пакетів

2) Зчитаємо pcap файл у датасет, для цього:

- функцією rdpcap зчитаємо дані у змінну
- проініціалізуємо списки для кожного поля, про яке вдалося витягнути

інформацію

- переводимо дані в числовий формат для обробки

3) За допомогою бібліотеки pickle, а саме pickle.load() завантажуємо розроблену модель.

4) Передаємо датасет у функцію predict() для аналізу трафіка.

5) За допомогою бібліотеки tkinter створюємо простий GUI (Graphic User Interface), де відображаємо дані за допомогою табличного віджета Treeview()

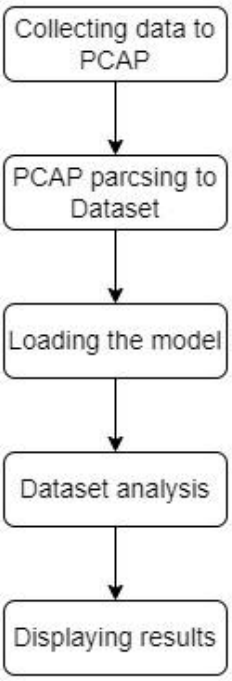

Рисунок 3.2. Блок-схема роботи програми

Отримуємо програму з простим і зрозумілим інтерфейсом (рис.3.3). В цілому для аналізу трафіка можна задати будь-які параметри. У нашому випадку можна відфільтрувати конкретні протоколи для сканування, задати час та максимально потрібну кількість пакетів.

Про правильність роботи говорить те, що весь проаналізований трафік був визначений як нормальний (рис.3.4) і поідомлення про виявлену небезпеку та її тип не надходило.

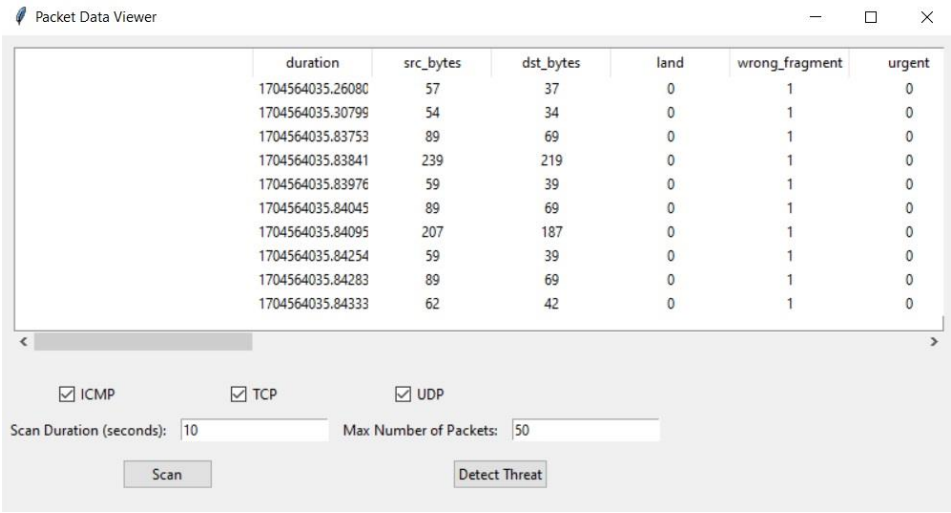

Рисунок 3.3. Інтерфейс програми

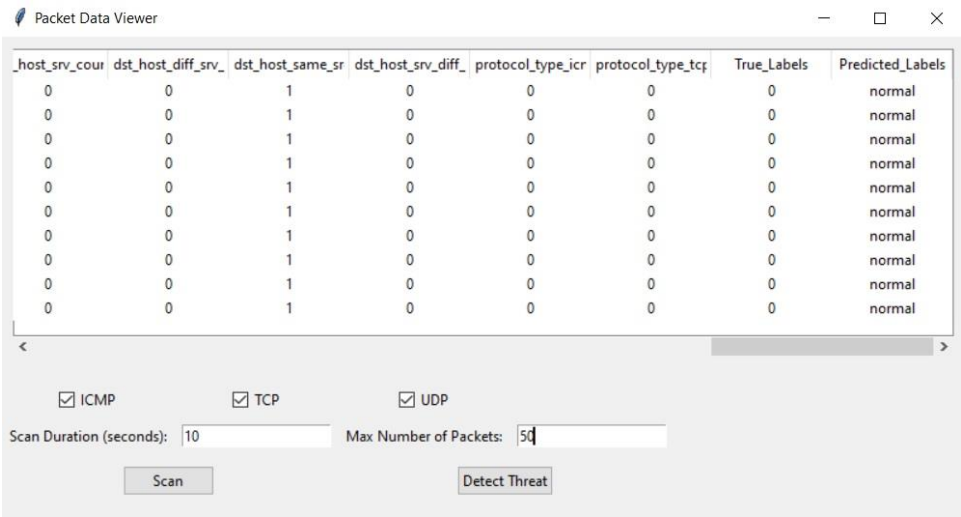

Рисунок 3.4. Сканування

3.2 UML-діаграми розробленого програмного забезпечення

Для кращого сприйняття та в цілому для ведення документації розробимо UML-діаграми розробленого ПЗ. Unified Modeling Language - це стандарт для моделювання, що використовується у сфері розробки програмного забезпечення. Вони дозволяють розробникам та інженерам систем створювати графічне відображення різних аспектів системи, щоб полегшити розуміння, взаємодію і співпрацю між різними учасниками проекту.

Діаграма послідовності (Sequence Diagram) використовується для

відображення послідовності взаємодії між об'єктами або компонентами системи в часі. Вона графічно відображає, як об'єкти спілкуються між собою та в якому порядку вони виконують свої дії. У даному випадку серед об’єктів у нас є користувач, інтерфейс, компонент сканування і модель ML.

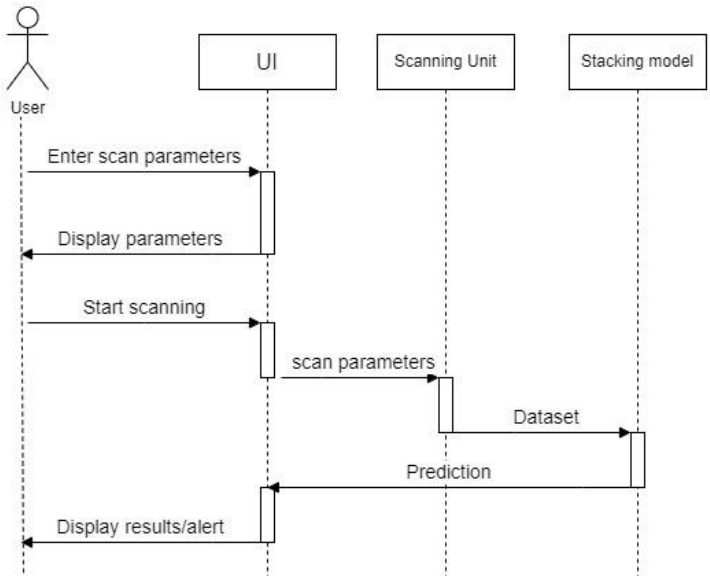

Рисунок 3.5. Sequence Diagram

Діаграма використання (Use Case Diagram) використовується для моделювання функціональності системи з точки зору її користувачів. Діаграма використання дозволяє визначити основні функції системи та способи взаємодії користувачів з нею.

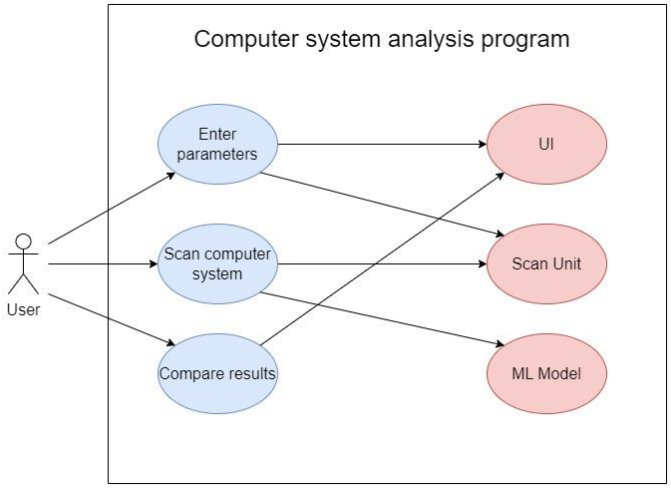

Рисунок 3.6. Use Case Diagram

Діаграма компонентів (Component Diagram) використовується для візуалізації структури системи на рівні компонентів та їх взаємодії. Самі компоненти представляють фізичні або логічні частини системи, які мають певні функції чи відповідальності. Компоненти можуть бути програмними модулями, бібліотеками, файлами конфігурації тощо.

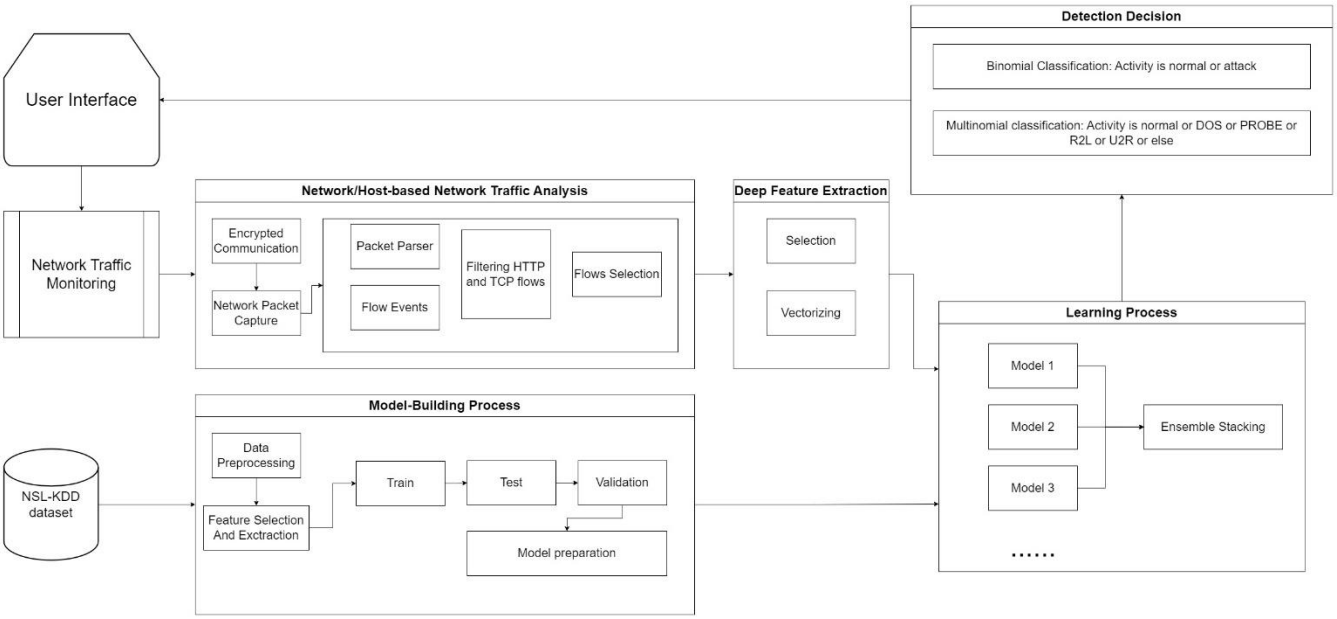

Рисунок 3.7. Component Diagram

Розглянемо компоненти розробленого ПЗ. В інтерфейсі користувача можна запускати сканування мережі, вибирати деякі його параметри, а також отримувати результати після завершення. Блок Network Traffic Monitoring відповідає за збір даних.

— Захоплення пакетів мережі (Network Packet Capture) включає в себе запис пакетів мережі для подальшого аналізу. Завдяки цьому є можливість детально вивчати взаємодії в мережі, аналізувати дані, які передаються між пристроями.

— Аналізатор пакетів (Packet Parser) використовується для розбору (аналізу) записаних пакетів мережі. Парсери можуть виділяти різноманітні дані, такі як адреси, заголовки пакетів, протоколи тощо.

— Події потоків (Flow Events) - вказує на виникнення подій або змін в потоках мережевого трафіку.

— Фільтрація HTTP і TCP потоків (Filtering HTTP and TCP Flows) - означає

вибіркову обробку потоків, які пов'язані з протоколами HTTP і TCP. Це допомагає виділяти певний тип трафіку для подальшого аналізу.

— Вибір потоків (Flow Selection) полягає у можливості вибору конкретних мережевих потоків для подальшого аналізу. Це може бути корисним для зосередження на конкретних подіях чи пристроях.

— Deep Feature Extraction – відповідає за вибір даних, отриманих скануванням.

— Model-building process – відповідає за тренування моделей машинного навчання, яке відбувається шляхом використання NSL-KDD датасета, що є потужним джерелом інформації щодо інцидентів в мережі.

— Learning process – компонент, що визначає процедуру вивчення обраними ML моделями даних, отриманих на основі згенерованого трафіка. Ensemble Stacking — розроблена модель ансамблевого навчання на основі 4 алгоритмів.

— Detection Decision – виконує дві функції: виявляє аномальну активність та визначає характер атаки (прогнозування). Для цього застосовується біноміальна та поліноміальна класифікації. Результат користувач отримує в інтерфейсі програми.

3.3 Інтеграція розроблених механізмів в існуючі програмні засоби

Отже, ми побудували ПЗ для аналізу ризиків КС на основі технологій бібліотеки scapy.all. Однак, такий підхід має недоліки. Дана бібліотека спеціалізована на веб-скрапінгу. Якщо ж потрібно вивчати заголовки пакетів або робити аналіз мережевого трафіку, Scrapy може бути занадто важким і неефективним інструментом. Тому він використовується переважно для обробки веб-сторінок і не надає такого рівня контролю над мережевим трафіком, як у випадку з tcpdump чи Wireshark. Застосуємо розроблені механізми разом з існуючими програмними засобами, щоб зробити аналіз більш досконалим.

1) В першу чергу розглянемо Wireshark. Це потужний інструмент для аналізу

мережевого трафіку, який надає широкі можливості для вивчення та зрозуміння деталей комунікації в мережі. Дозволяє захоплювати та аналізувати пакети мережевого трафіку в режимі реального часу або на основі раніше захопленого файлу. Забезпечує докладні відомості про кожен пакет, включаючи заголовки протоколів, дані та іншу інформацію. В той же час scrapy орієнтований на веб-скрапінг та взаємодію з веб-сайтами. Зазвичай не надає детальної інформації про мережевий трафік на рівні пакетів, оскільки визначений для витягання інформації зі структурованих джерел. Wireshark підтримує широкий спектр мережевих протоколів і може автоматично визначати та аналізувати їхні заголовки. Це охоплює TCP, UDP, IP, HTTP, SSL, DNS, FTP, і багато інших. Scrapy в свою чергу спеціалізований на протоколах HTTP і HTML, які використовуються для взаємодії з веб-сайтами.

Стосовно фільтрації та пошуку, Wireshark надає потужні засоби фільтрації та пошуку, що дозволяє швидко знаходити та аналізувати пакети за різними критеріями, такими як IP-адреса, порт, протокол тощо. Scrapy зазвичай працює на рівні вищого рівня абстракції, і функції фільтрації зазвичай обмежені структурою веб-сайту чи декількома властивостями. Ну і головна перевага - Wireshark може працювати в режимах захоплення пакетів на локальній машині або в режимі захоплення на мережевому інтерфейсі.

Для захоплення пакетів необхідно вибрати мережу та налаштувати фільтри (рис.3.8), після чого запустити сканування (Рис.3.9). Далі сканування можна зупинити, коли це потрібно, а пакети зберегти у pcap файл, після чого можна буде використати розроблену програму, щоб проаналізувати дані.

Рисунок 3.8. Вибір мережі для захоплення пакетів

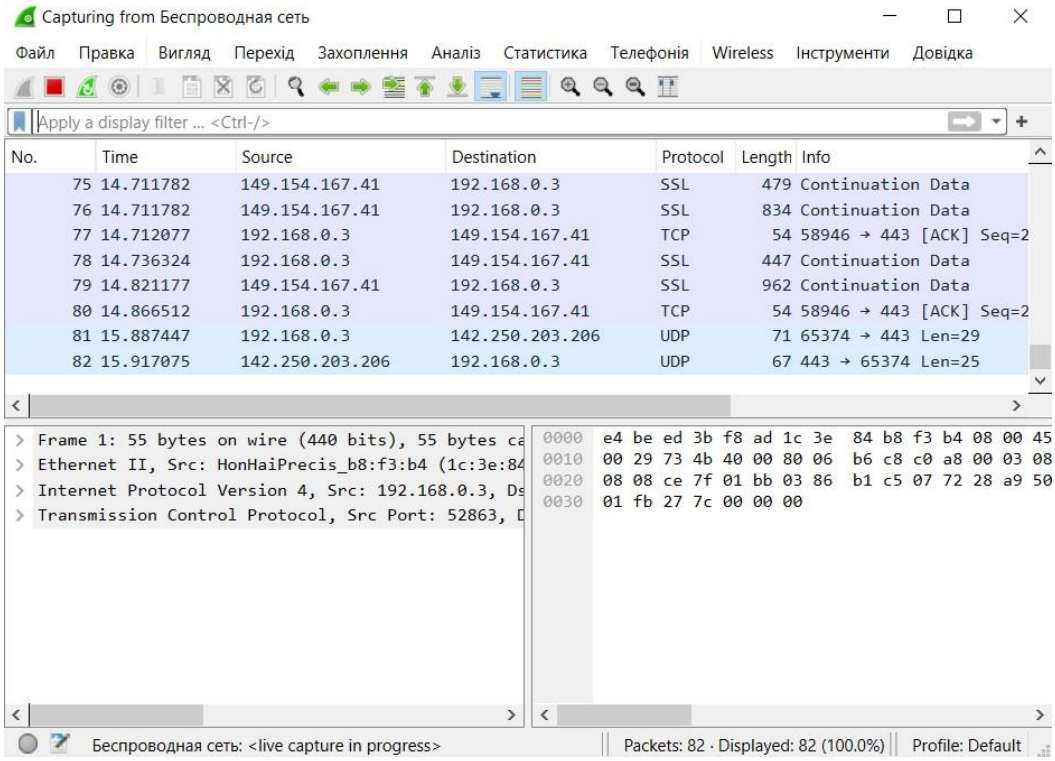

Рисунок 3.9. Процес сканування

Для того, щоб зробити цей процес автоматичним, ми можемо застосувати принцип роботи Wireshark у розробленому ПЗ. Для цього використаємо модуль subprocess. Модуль subprocess в Python надає можливість запускати нові процеси, підключатися до їхнього стандартного вводу/виводу/помилок, та отримувати статус їхньої роботи. Він є частиною стандартної бібліотеки Python і використовується для взаємодії з зовнішніми процесами.

Можна таким чином створити функцію capture_packets і передати туди параметри file_path, duration, capture_filter. У ній прописати команду виклику command = ['tshark', '-a', f'duration:{duration}', '-w', file_path, capture_filter]. Після цього викликати subprocess.run(command). Коли запустимо дану функцію, програма запустить Wireshark для збору даних, які можна буде обробити автоматично.

Таким чином ми значно покращили якість виявлення загроз у комп’ютерній системі, просто використовуючи уже існуючий аналізатор Wireshark.

2) Другим програмним продуктом, який ми протестуємо, буде tcpdump. Це утиліта командного рядка для аналізу та захоплення мережевого трафіку. Вона

дозволяє вам перехоплювати пакети, що проходять через мережевий інтерфейс пристрою в мережі.

Tcpdump працює на наступних рівнях, визначених моделлю OSI (Open Systems Interconnection):

- Link Layer (рівень зв'язку): На цьому рівні tcpdump може захоплювати та аналізувати пакети протоколів, таких як Ethernet, Wi-Fi, а також інші, які працюють на рівні зв'язку.
- Network Layer (рівень мережі): tcpdump може аналізувати пакети протоколів мережевого рівня, таких як IP (Internet Protocol). Він може фільтрувати пакети за IP-адресами, протоколами та іншими параметрами.
- Transport Layer (рівень транспортування): Пакети, які працюють на рівні транспортного протоколу, такі як TCP (Transmission Control Protocol) та UDP (User Datagram Protocol), також можуть бути аналізовані tcpdump.

Tcpdump працює з командного рядка, що дозволяє аналізувати та захоплювати трафік без використання графічного інтерфейсу. Приклад запису команди: command = ['tcpdump', '-i', interface, '-w', output_file, '-c', str(packet_count)], де

- interface = 'eth0' - інтерфейс для сканування
- output_file = 'captured_packets.pcap' – файл для збереження
- packet_count = 100 – кількість пакетів

Таким чином, найбільшими перевагами tcpdump є його функціональність і доступність.

3) NetworkMiner – ще один інструмент аналізу мережі (рис.3.10). Він здатен захоплювати передані файли та мультимедійні потоки. Крім уже згаданих функцій, він може витягувати метадані з файлів та зображень, знайдених у мережевому трафіку, що надає відомості про характер переданого вмісту. Одним із його переваг є ідентифікація операційних систем пристроїв в мережі, що допомагає профілюванню та розумінню ландшафту мережі. Крім того, інструмент надає вид таймлайну подій мережі, полегшуючи візуалізацію послідовності подій.

Виклик цього мережевого аналізатора на мові Python аналогічний до попередніх, слід лише пам’ятати, що NetworkMiner працює лише в середовищі

Windows.

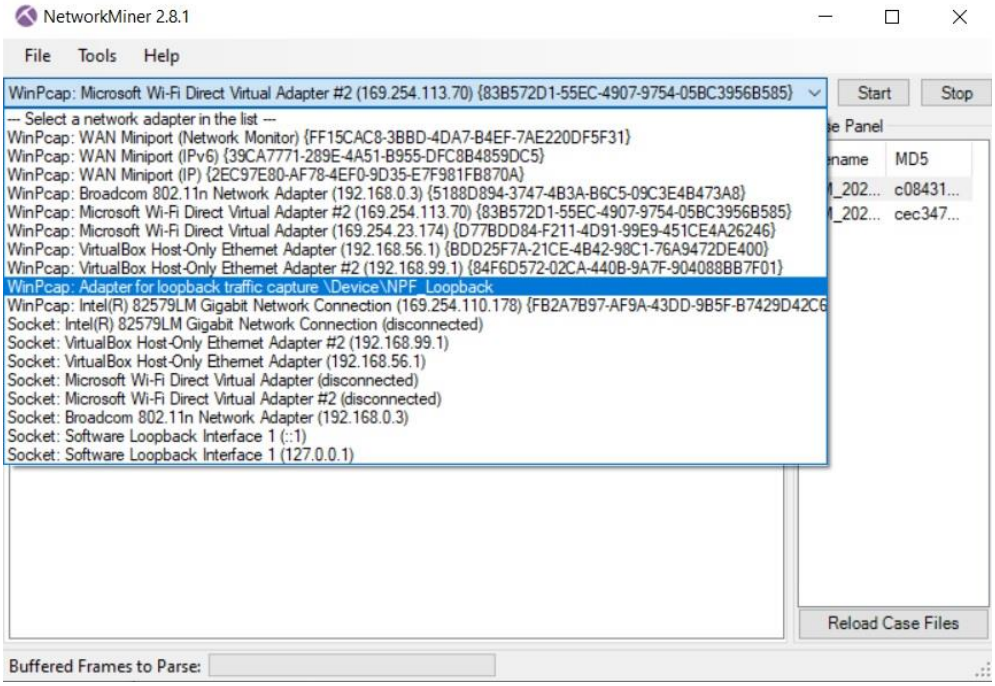

Рисунок 3.10. Вибір мереж для сканування з NetworkMiner

Висновки до розділу 3

В данному розділі було розроблено програмне забезпечення механізмів виявлення і прогнозування ризиків безпеки КС на основі аналізу даних системи. Розроблена програма має простий інтерфейс користувача, якого достатньо для аналізу та виявлення. Крім того, у розділі описано принципи роботи і складові даного ПЗ за допомогою UML-діаграм, що дозволяє спростити розуміння роботи використаних технологій.

Розроблена у другому розділі модель, вже натренована на відповідному датасеті, була успішно використана в роботі. Під час аналізу КС загроз виявлено не було, що відповідає дійсності. В останньому пункті було продемонстровано можливість використання розроблених механізмів у взаємодії з різними мережевими аналізаторами. Кожен з них дає підвищує якість аналізу і має свої переваги, порівняно із технологією scrapy, використаною спочатку.

4 ЕКСПЕРИМЕНТАЛЬНІ ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ РОБОТИ МЕХАНІЗМІВ ВИЯВЛЕННЯ ТА ПРОГНОЗУВАННЯ РИЗИКІВ ПОРУШЕННЯ ЗАХИЩЕНОСТІ КС

4.1 Постановка експериментів та підготовка вхідних даних

Оскільки у даній роботі розглядається аналіз ризиків системи комп’ютера на основі даних різного роду трафіка, то підготовка моделей має відбуватися на відповідних даних. Тож для тренування використаємо NSL-KDD датасет.

NSL-KDD (National Science Laboratory - Knowledge Discovery and Data Mining) є датасетом, що використовується розробниками для створення і тестування систем виявлення інцидетів у системах (IDS). Він є розширеною версією відомого датасету KDD Cup 99, який використовувався для конкурсу KDD Cup 1999 щодо виявлення вторгнень. KDD Cup - це щорічний конкурс, який організовується спільнотою знань та видобутку даних з метою сприяння розвитку алгоритмів та методів в галузі обробки даних та виявлення знань. Створення NSL-KDD було обумовлене недоліками та вкрай великою нерівністю даних в KDD Cup 99.

Спочатку визначимо усі можливі атрибути даних, які можна знайти в датасеті:

- Duration: Тривалість з'єднання в секундах.
- Protocol_Type: Тип протоколу (tcp, icmp, udp).
- Service: Тип мережі (private, ftp_data, eco_i, telnet, http, smtp, ftp, ldap, pop_3, courier, discard, ecr_i, imap4, domain_u, mtp, інші).
- Flag: Статус прапорця (REJ, SF, RSTO, S0, RSTR, SH, S3, S2, S1, RSTOS0, OTH).
- Src_Bytes: Кількість байтів, відправлених від джерела до призначення.
- Dst_Bytes: Кількість байтів, відправлених до призначення від джерела.
- Land: Якщо з'єднання на той же хост (1), інакше 0.
- Wrong_Fragment: Кількість неправильних фрагментів (0, 1, 3).
- Urgent: Кількість термінових пакетів (0, 1, 2, 3).
- Hot: Кількість "гарячих" індикаторів (0, 1, 2, 3, 4, 5, 6, 7, 10, 11, 15, 18, 19,

22, 30, 101).

- Num_Failed_Logins: Кількість невдалих спроб входу (0, 1, 2, 3, 4).
- Logged_In: Якщо ввійшов (1), інакше 0.
- Num_Compromised: Кількість компрометованих умов (0-796).
- Root_Shell: Якщо отримано кореневий shell (1), інакше 0.
- Su_Attempted: Якщо "su root" звертається (1), інакше 0.
- Num_Root: Кількість звернень до кореневого доступу (0-878).
- Num_File_Creations: Кількість створених файлів (0-100).
- Num_Shells: Кількість shell-промптів (0, 1, 2, 5).
- Num_Access_Files: Кількість операцій з доступом до файлів (0-4).
- Num_Outbound_Cmds: Кількість вихідних команд.
- Is_Host_Login: Якщо вхід гарячий (1), інакше 0.
- Is_Guest_Login: Якщо вхід гостинний (1), інакше 0.
- Count: Кількість сполучень з тим самим хостом за останні 2 секунди.
- Srv_Count: Кількість сполучень з тією ж самою службою за останні 2 секунди.
- Serror_Rate: Процент сполучень із синхронною помилкою.
- Srv_Serror_Rate: Процент сполучень із синхронною помилкою на службі.
- Rerror_Rate: Процент сполучень із відхиленою помилкою.
- Srv_Rerror_Rate: Процент сполучень із відхиленою помилкою на службі.
- Same_Srv_Rate: Процент сполучень з тією ж самою службою.
- Diff_Srv_Rate: Процент сполучень з різними службами.
- Srv_Diff_Host_Rate: Процент сполучень з різними хостами для тієї ж самої служби.
- Dst_Host_Count: Кількість сполучень з тим самим призначенням хоста.
- Dst_Host_Srv_Count: Кількість сполучень з тим самим призначенням хоста та службою.
- Dst_Host_Same_Srv_Rate: Процент сполучень з тим самим призначенням хоста та службою.

- Dst_Host_Diff_Srv_Rate: Процент сполучень з різними службами на тому ж самому хості.
- Dst_Host_Same_Src_Port_Rate: Процент сполучень на тому ж самому хості з тим самим портом джерела.
- Dst_Host_Srv_Diff_Host_Rate: Процент сполучень на тому ж хості та службі з різними хостами.
- Dst_Host_Serror_Rate: Процент сполучень на тому ж хості з помилкою S0.
- Dst_Host_Srv_Serror_Rate: Процент сполучень на тому ж хості та службі з помилкою S0.
- Dst_Host_Rerror_Rate: Процент сполучень на тому ж хості з відхиленою помилкою.
- Dst_Host_Srv_Rerror_Rate: Процент сполучень на тому ж хості та службі з відхиленою помилкою.
- Attack: Тип атаки (apache2, back, buffer_overflow, ftp_write, guess_passwd, httptunnel, imap, ipsweep, land, loadmodule, mailbomb, mscan, multihop, named, neptune, nmap, normal, perl, phf, pod, portsweep, processtable, ps, rootkit, saint, satan, sendmail, smurf, snmpgetattack, snmpguess, sqlattack, teardrop, udpstorm, warezmaster, worm, xlock, xsnoop, xterm).
- Level: Рівень небезпеки атаки (0-21).

	duration	src_bytes	dst_bytes	land	wrong_fragment	urgent	hot
count	125973.00000	1.259730e+05	1.259730e+05	125973.000000	125973.000000	125973.000000	125973.000000
mean	287.14465	4.556674e+04	1.977911e+04	0.000198	0.022687	0.000111	0.204409
std	2604.51531	5.870331e+06	4.021269e+06	0.014086	0.253530	0.014366	2.149968
min	0.00000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000
25%	0.00000	0.000000e+00	0.000000e+00	0.000000	0.000000	0.000000	0.000000
50%	0.00000	4.400000e+01	0.000000e+00	0.000000	0.000000	0.000000	0.000000
75%	0.00000	2.760000e+02	5.160000e+02	0.000000	0.000000	0.000000	0.000000
max	42908.00000	1.379964e+09	1.309937e+09	1.000000	3.000000	3.000000	77.000000

Рисунок 4.1. Коротка статистика по даним

Датасет складається з кількох файлів формату arff та txt, що містять дані щодо трафіку з мітками чи без. Відобразимо статистичну складову даних на екрані

(рис.4.1). На цьому етапі можна зробити висновок, що більшість даних є числовими і лише деякі атрибути категоріальні (і можуть бути приведені до числа). Крім того, тренувальна і тестова версія датасета мають однакові змінні, що спрощує роботу з ними.

Приберемо пусті значення та значення, що повторюються. Оцінимо, як збалансовані дані (рис.4.2). Отже, можливо всього 23 різних видів атаки. Однак, у багатьох атак подібний принцип роботи, і для спрощення ми можемо віднести їх до одного класу.

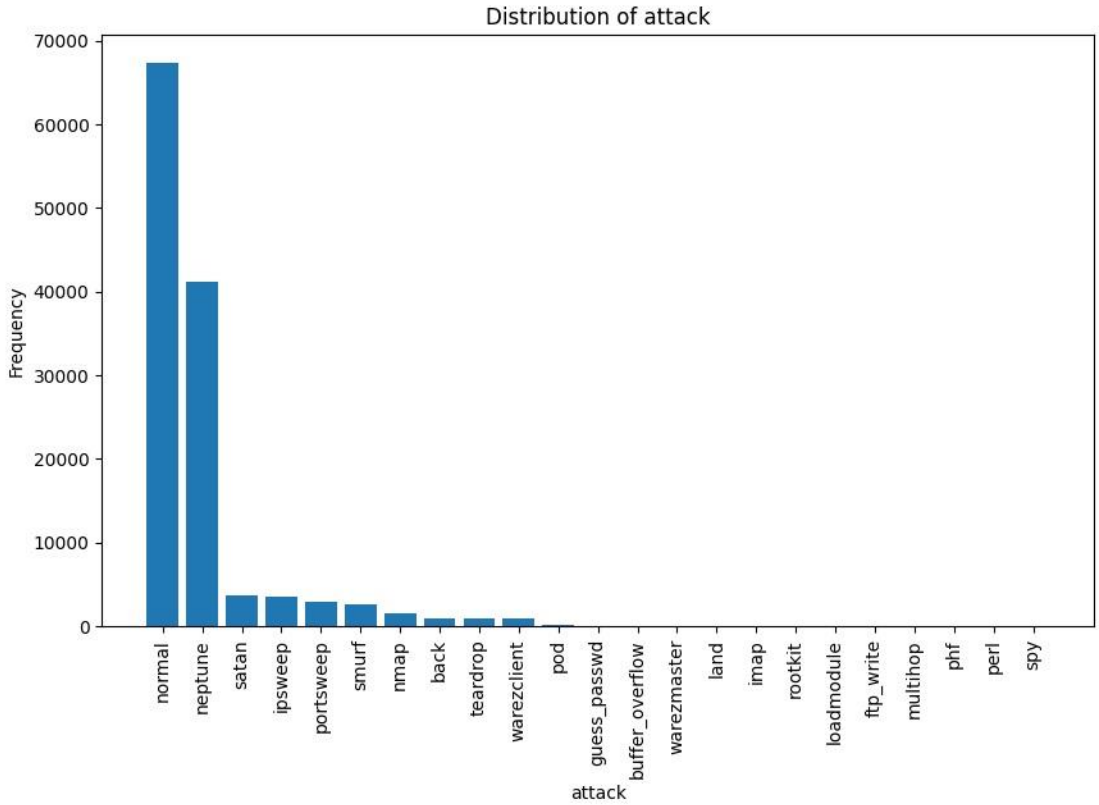

Рисунок 4.2. Гістограма розподілу даних

Таким чином NSL-KDD містить п’ять основних класів трафіку (табл. 4.1):

— Normal - легітимний мережевий трафік, який вважається безпечним.

— DoS (Denial of Service) - атаки, спрямовані на перевантаження ресурсів цілі, забираючи їх у легітимних користувачів.

— Probe - атаки, що виконують сканування мережі з метою виявлення слабких місць для проведення подальших атак.

— U2R (User to Root) - атаки, спрямовані на те, щоб отримати права адміністратора системи з прав користувача.

— R2L (Remote to Local) - атаки, що спрямовані на отримання доступу до системи через мережу, якщо ви вже є користувачем у цій системі.

Таблиця 4.1 Спрощена класифікація загроз

Тип	Підтип	Частота	Відсоток
normal	-	67343	53.458281
Dos	neptune, back, land, pod, smurf, teardrop, mailbomb, apache2, processtable, udpstorm, worm	45927	36.457812
Probe	ipsweep, nmap, portsweep, satan, mscan, saint	11656	9.252776
R2L	ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient, warezmaster, sendmail, named, snmpgetattack, snmpguess, xlock, xsnoop, httptunnel	995	0.789852
U2R	buffer_overflow, loadmodule, perl, rootkit, ps, sqlattack, xterm	52	0.041279

Оскільки маємо велику кількість даних та атрибутів, доцільно провести кореляцію даних. Виберемо числові та категоріальні ознаки з навчального набору даних (df_train). Потім обчислимо кореляційну матрицю для цих ознак, і відобразимо її у вигляді теплової карти (heatmap).

Основна ідея тут полягає в тому, щоб вивести графічне представлення кореляції між числовими та категоріальними ознаками. Кожен елемент матриці представляє кореляцію між двома ознаками, де кольори теплової карти вказують на силу та напрямок кореляції (наприклад, сильна позитивна кореляція позначається яскравим кольором, а сильна негативна - темним).

Також, за допомогою параметру mask функції sns.heatmap, що має значення True для верхнього трикутника матриці, приховаємо дубльовані значення (оскільки кореляція між ознакою і собою ж завжди дорівнює 1). Це зробить графік більш читабельним (рис 4.3).

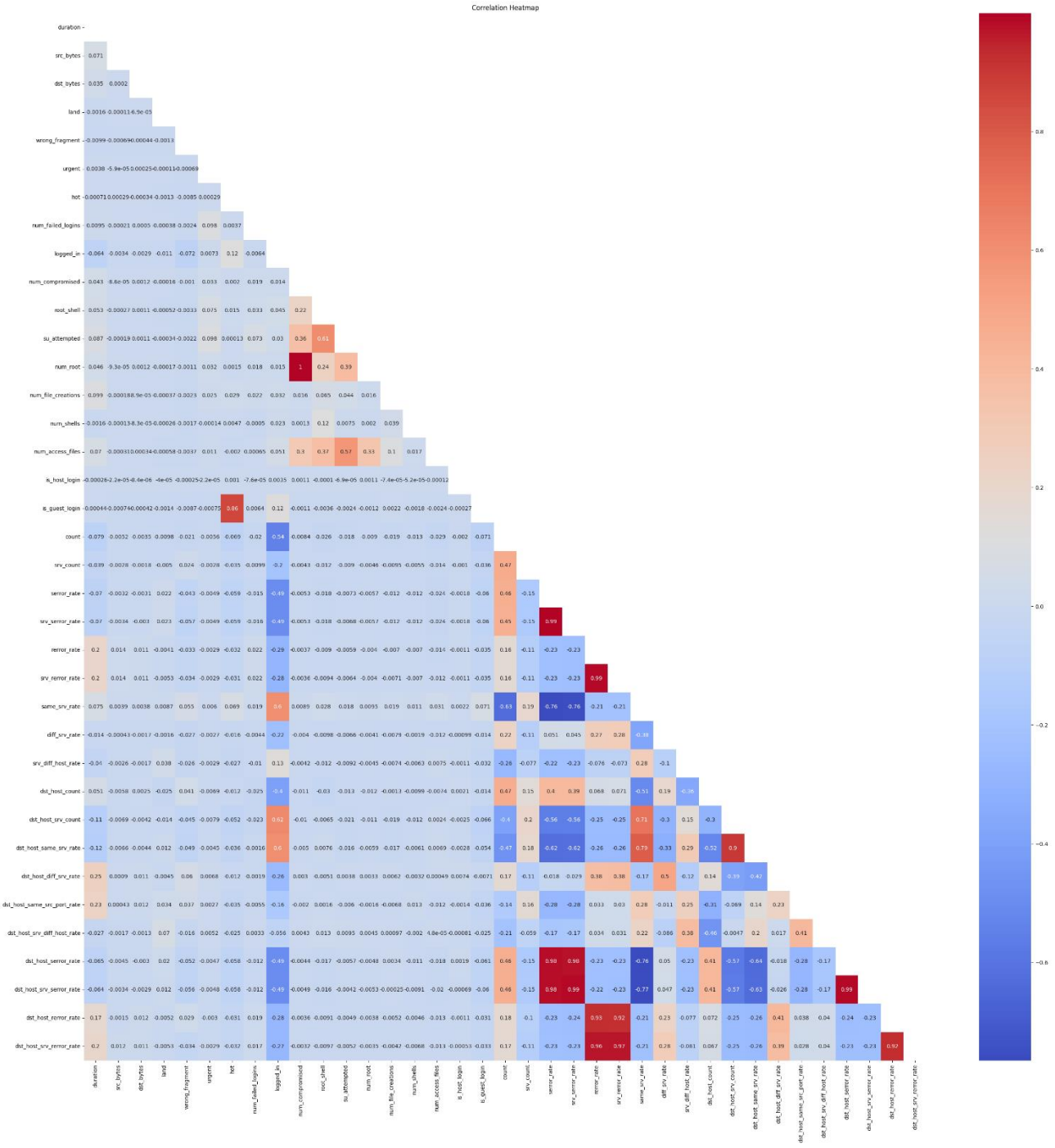

Рисунок 4.3. Кореляційна матриця даних

Висновки щодо кореляції даних:

- num_outbund_cmds не має кореляції з іншими ознаками, оскільки всі його значення дорівнюють 0. Це свідчить про те, що ця ознака не надає значущої інформації для аналізу.
- Пари значень (srv_rerror_rate, rerror_rate), (root, compromised) та деякі інші майже ідеально корелюють позитивно, вказуючи на схожу дисперсію. Зайва

інформація може впливати на продуктивність моделі.

- Змінна "level" сильно незбалансована, оскільки 89.58% рівнів є вищими або рівними 18. Тому вона також підлягає вилученню, оскільки більшість рівнів є високими.

Далі для кожного атрибута, що залишився побудуємо діаграми розсіювання (scatter plot). Це графік, на якому окремі точки представлені значеннями двох змінних. Кожна точка на графіку відображає конкретне спостереження та його значення на двох змінних. Це дозволяє візуально оцінити, чи існує яка-небудь взаємодія або залежність між цими змінними.

Основні елементи діаграми розсіювання:

- Вісі X та Y - відображають значення двох змінних.
- Точки - кожна точка представляє конкретне спостереження.
- Підписи точок - можуть бути мітки, що вказують ідентифікатор або значення спостереження.

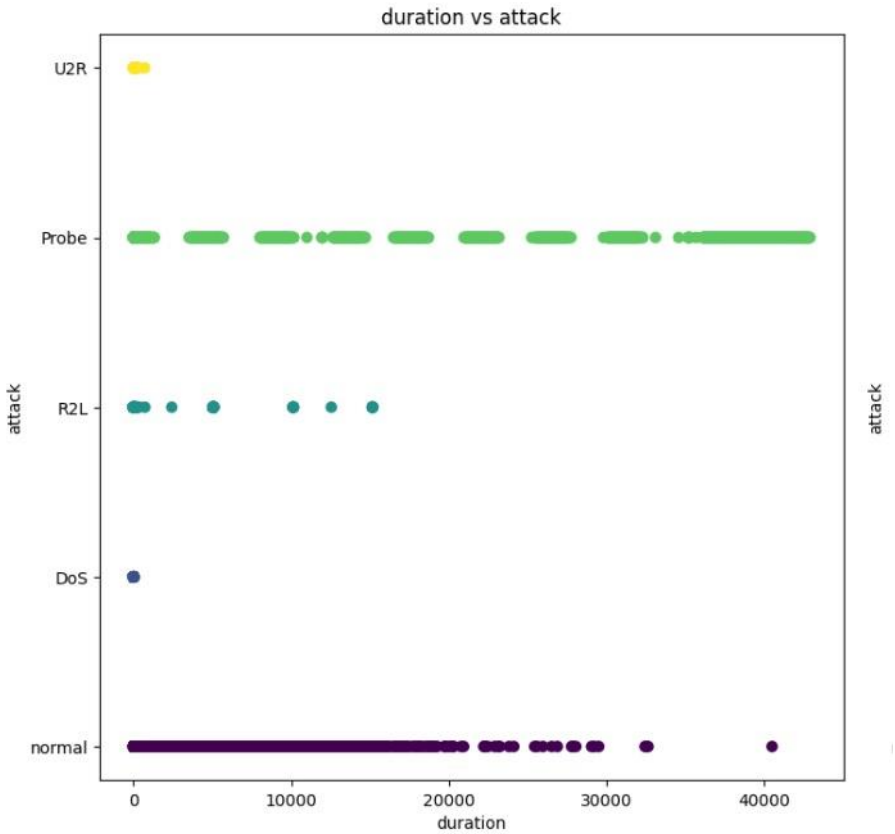

Рисунок 4.4. Діаграма розсіювання для Duration

Діаграми розсіювання дуже корисні для виявлення кореляцій, визначення взаємозв'язків та виявлення аномалій у великих наборах даних. Вони дозволяють швидко виявляти закономірності та взаємозв'язки між змінними, що може бути корисним при аналізі даних та виборі відповідних моделей.

Враховуючи особливості всіх даних, які ми отримали, виконаємо ще декілька операцій, спрямованих на підготовку та обробку даних для моделювання:

- Категоріальні змінні, такі як 'protocol_type', 'service', та 'flag', перетворимо за допомогою One-Hot Encoding (OHE). Це дозволяє представити категоріальні дані у вигляді числових фіч, що полегшує їх використання в моделях машинного навчання.
- Використаємо RobustScaler для масштабування числових змінних. Цей метод є стійким до викидів та інших аномалій у даних.
- Приведення до бінарного вигляду (Бінаризація): Колонки 'land' та 'logged_in' конвертуються в тип float, а стовпці 'class' в модифікованих наборах даних 'df_train2' та 'df_test2' перетворюються на бінарну форму. 'class' визначає, чи атака є "нормальною" чи "аномальною".
- Застосовується аналіз головних компонентів для зменшення розмірності даних. PCA (Principal Component Analysis) — це метод зменшення розмірності даних, що використовується для видалення зайвих або корельованих функцій (ознак) та концентрації варіації вздовж нових осей, які є лінійними комбінаціями оригінальних функцій. Основна мета - зменшити кількість вимірів (або атрибутів) даних, при цьому зберігаючи якнайбільше інформації.

Тепер побудуємо гістограми розподілу атак для кожного унікального значення певного ознаки (feature). На рисунку 4.5 зображено розподіл різних видів атак по мережевим протоколам. Ця та інші гістограми показують нерівномірність розподілу основних типів атак, що може зменшити точність при класифікації. Не дивлячись на відносно рівний розподіл нормального і аномального трафіка в цілому (53.46% до 46.54%), майже всі аномальні атаки є Dos-атаками,а R2L, U2R покривають дуже низький відсоток (рис.4.6).

Для того щоб позбутися такої проблеми, застосуємо метод Oversampling. Це

техніка в області обробки даних, яка використовується для балансування класів в задачах класифікації, де кількість екземплярів одного класу суттєво менша за кількість екземплярів іншого класу. Основна ідея полягає в тому, щоб створити додаткові приклади менш представлених класів шляхом копіювання, модифікації або генерації нових екземплярів цих класів.

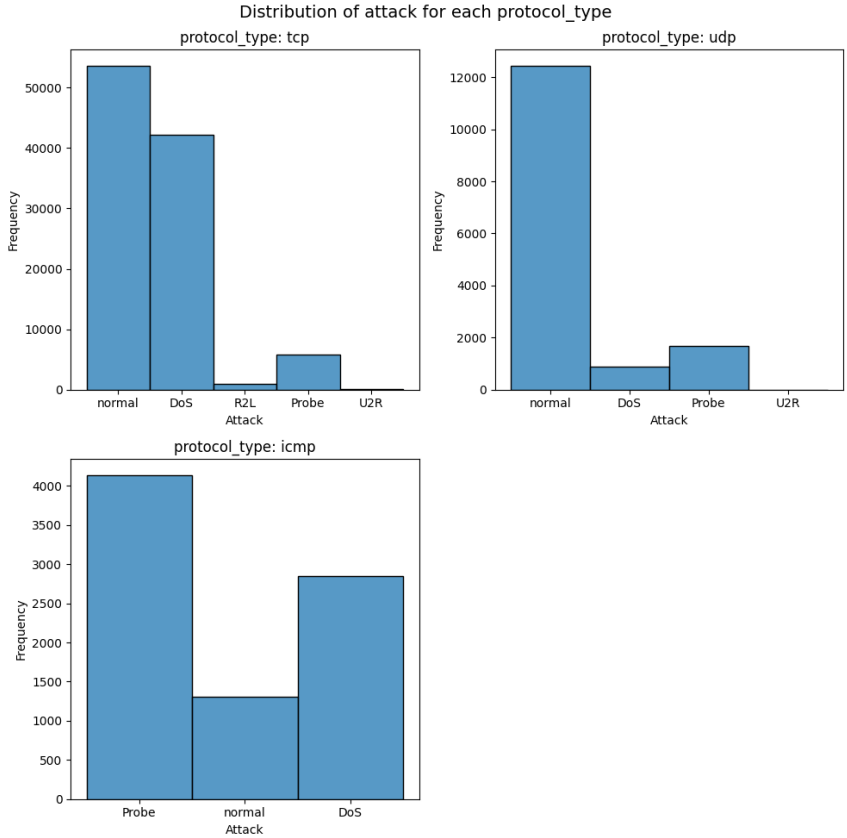

Рисунок 4.5. Розподіл видів атак по протоколам

На відміну від Random Oversampling, стратегія якого включає в себе випадкове копіювання екземплярів менш представлених класів для рівномірного збалансування, використаємо метод SMOTE. Synthetic Oversampling (Synthetic Minority Over-sampling Technique) використовується для створення штучних прикладів менш представлених класів. Він вибирає існуючі приклади більш слабкого класу та генерує нові шляхом лінійної інтерполяції між кількома найближчими сусідами цих прикладів.

Останнім кроком для підготовки даних буде вибір головних ознак на основі алгоритма ANOVA F-test. Analysis of Variance застосовується, щоб порівнювати

середні значення двох або більше груп та визначати, чи є статистично значущі різниці між ними. У контексті вибору ознак ANOVA F-test використовується для визначення того, як кожна конкретна ознака впливає на цільову змінну.

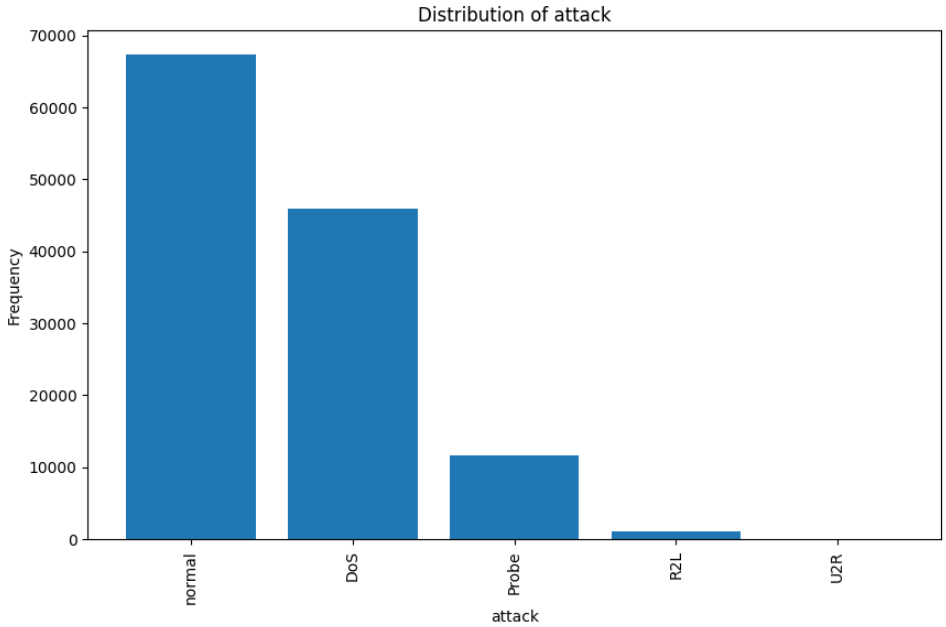

Рисунок 4.6. Розподіл видів атак в датасеті

- SelectPercentile - це метод вибору ознак від sklearn, який дозволяє вибрати ознаки на основі їхньої важливості.
- score_func = f_classif - це функція важливості, в даному випадку, ANOVA F-test, яка вимірює статистичну значущість впливу ознак на цільову змінну.
- percentile = p - вказує, який відсоток найважливіших ознак ви хочете залишити.
- Після виклику методу fit_transform здійснюється вибір ознак. Обрані ознаки можна отримати за допомогою selector.get_support(indices=True).

Завдяки функції виділення ознак, можемо сформувати чотири основних датасети. У кожен із цих датасетів покладемо дані про нормальний трафік та дані про один вид атаки. Таким чином матимемо датасети: normal/Dos, normal/Probe, normal/R2L, normal/U2R. Далі розділимо дані для тренування та тестування і використаємо 4 моделі, що входять до стеку ансамблевої моделі, спроектованої у другому розділі (Random Forest, CatBoost, K-nearest neighbors, XGBoost). Кожну

модель будемо тренувати як на даних з використанням oversampling, так і без нього. Oversampling ефективний метод, однак може призводити до перенавчання. Використавши обидва варіанти, ми зменшимо цей вплив.

Отже, кожна з чотирьох моделей буде 8 разів виконувати задачу бінарної класифікації – відділяти нормальний трафік від аномального.

Такий підхід набагато важливіший та ефективніший, бо не змушує базові моделі відрізняти різні види атак один від одного, а дає можливість зосередитись на виявленнні аномалій серед легітимного трафіку. Потім мета-модель, яка отримує результати усіх інших моделей, на основі висновків кожної з них робить остаточний прогноз і відносить пакет до однієї з категорій загроз, таким чином виявляючи і прогнозуючи небезпеку.

4.2 Порівняльний аналіз результатів експериментів

Для порівняння результатів використаємо кожну з побудованих моделей для класифікації і визначимо якості моделей для різних відсоткових значень ознак.

Цикл обчислення метрик для різних відсотків ознак виглядає так:

- Створюється порожній DataFrame results, де будуть зберігатися результати обчислень для кожного відсотку ознак.
- Використовується цикл для ітерації від 5% до 100% включно і обчислення метрик для кожного відсотка ознак.
- Для кожного відсотка обирається відповідна кількість ознак за допомогою feature_selector.
- Кожна з моделей навчається на обраних ознаках, і прогнозується цільова змінна для тестового набору.
- Обчислюються різні метрики, такі як Accuracy, Precision, Recall, F1 Score та ROC AUC.
- Результати для кожного відсотка ознак додаються до DataFrame results.
- Функція plot_metrics створює графік (рис.4.7), на якому відображаються

різні метрики для кожного відсотка ознак.

Інтерпритуючи побудовані графіки важливо розуміти, що нам необхідно максимізувати точність – усі застосовані метрики. Вони відображаються ламаними лініями, побудованими по точках на графіках

Accuracy, Precision, Recall, F1 Score, ROC AUC - всі вони відображаються на кожному графіку різними кольорами, допомагаючи визначити оптимальний відсоток ознак в залежності від конкретної метрики. Графік може допомогти визначити, який відсоток ознак призводить до найкращої якості моделі в залежності від вибраної метрики.

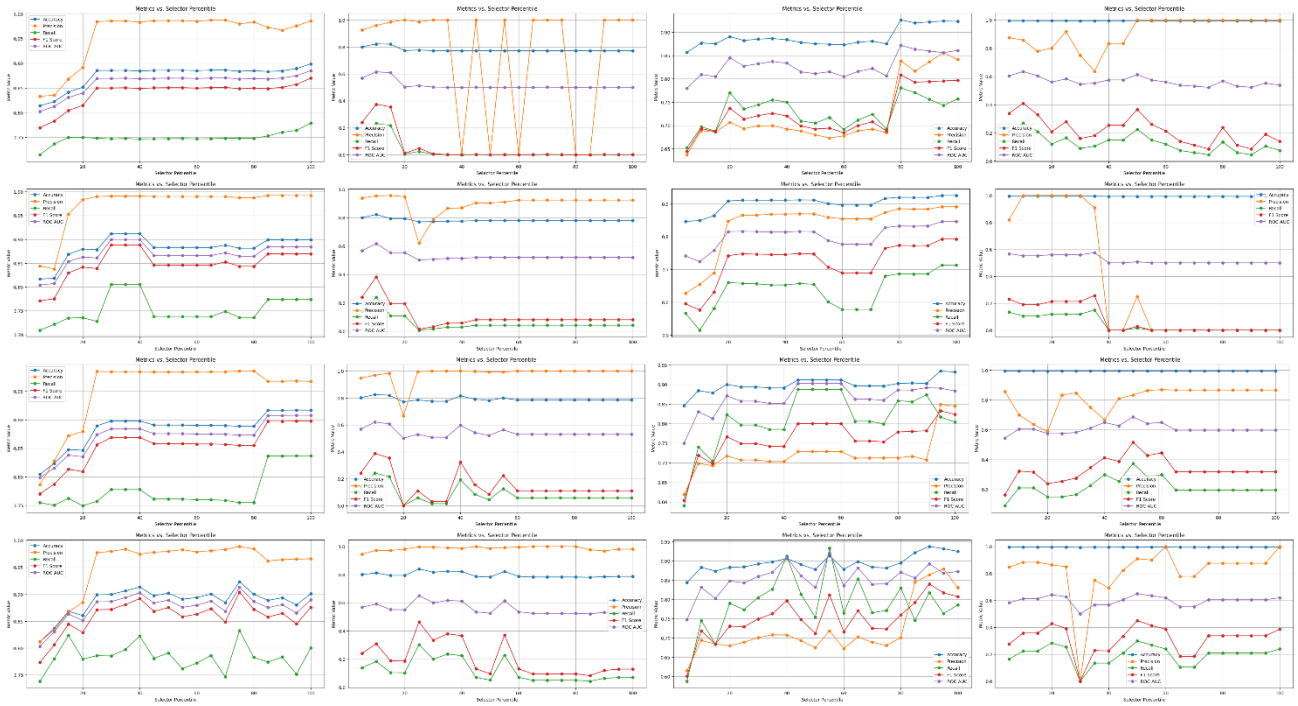

Рисунок 4.7. Графіки кривих вивчення за об’ємом даних

Зверху-вниз відображені графіки по алгоритмам (RF, KNN, XGB, CB) – в кожному рядку свій алгоритм. Типи атак (Dos, Probe, R2L, U2R) – зліва-направо, для кожного типу атак свій стовпчик. Звідси можна зробити висновок, для якої саме кількості даних кожна модель буде працювати найкраще, згідно кожної метрики. Наприклад, легко бачити, що Accuracy при визначенні Dos-атак не перевищує 93%, а рідкісний вид атак U2R всі алгоритми визначають досить точно. Якщо не враховувати його, можна сказати, що і на інших атаках моделі працюють нормально, але вони все-таки далекі від 100%.

Рисунок 4.8. Гістограма порівняння метрик моделей

На рисунку 4.8 представлені гістограми порівняння метрик. За допомогою них просто зробити висновок, яка метрика дає більші/менші результати для кожного

виду атак і що з цього випливає. Наприклад, низьке значення Recall для R2L, U2R говорить про велику кількість False Negative спрацювань (хибно-від'ємний результат або помилка другого роду), що є дуже небезпечним, адже це означає, що реальна загроза була класифікована як легітимний трафік.

Таблиця 4.2. Порівняння метрик моделей

Model Name	Accuracy	Precision	Recall	F1 Score	ROC AUC
CB	0.924040	0.939938	0.562795	0.664436	0.776429
CB(OS)	0.903795	0.893876	0.574966	0.635880	0.774585
KNN	0.911959	0.770001	0.436540	0.518163	0.714293
KNN(OS)	0.886677	0.898647	0.456239	0.505011	0.716689
RF	0.907123	0.945728	0.496146	0.599741	0.741998
RF(OS)	0.886677	0.898647	0.456239	0.505011	0.716689
XGB	0.908342	0.916555	0.534321	0.631487	0.761510
XGB(OS)	0.897860	0.902349	0.529081	0.585303	0.757474
Average	0.903309	0.895718	0.505791	0.580629	0.744958

Зведемо всі дані у таблицю, щоб отримати висновки. Кожне значення в таблиці – це середнє значення серед максимальних, показаних алгоритмом для різних атак. Як бачимо, Oversampling частіше зменшував точність, хоч його застосування і вважалося очевидним. Також середнє значення точності вийшло 90%, що в цілому непогано для моделей класифікації, але недостатньо для ефективної боротьби з ризиками. Інші метрики за принципом своєї роботи є більш чутливими і видають ще менші результати.

Тому оцінимо модель ансамблевого навчання, побудовану на основі розглянутих вище алгоритмів. Метод ансамблювання stacking передбачає тривале навчання моделі другого рівня на базі моделей першого рівня. У нашому випадку було досліджено 32 різні моделі: 4 принципово різних (RF, KNN, XGB, CB) навчались на двох видах наборів даних (з та без oversampling) для виявлення 4 видів загроз (Dos, Probe, R2L, U2R).

Для кожного прикладу в навчальному наборі базові моделі виробляють прогнози. Ці прогнози стають мета-даними. Мета-дані використовуються для

навчання топової моделі (моделі другого рівня), яка приймає прогнози базових моделей як вхідні дані і генерує остаточний прогноз. Під час валідації мета-модель застосовується до нових даних, для яких базові моделі роблять прогнози. Результатом є ансамбль прогнозів, і остаточний прогноз повинен бути кращим, ніж прогнози окремих моделей.

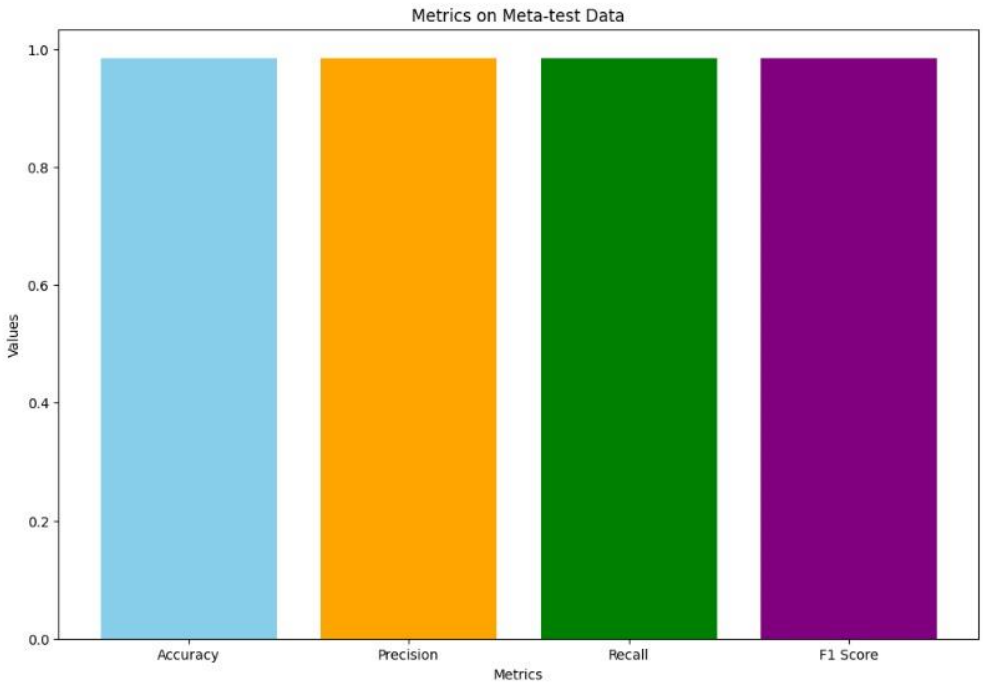

Рисунок 4.9. Гістограма метрик метамоделі

Як бачимо, результати всіх метрик збалансувалися. При цьому нова модель показує значно кращі результати:

- Accuracy: 0.9842537148
- Precision: 0.9842481423
- Recall: 0.9842537148
- F1 Score: 0.9841870644

Точність нової моделі складає 98,4%, при цьому максимальна точність серед моделей у стеку 92,4%, а середня 90,3%. Тобто, можна зробити висновок, що запропонована в роботі комбінація моделей позитивно показує себе у взаємодії. Інші метрики теж зросли до такого ж рівня, при тому, що раніше F1, наприклад, показувала 66,4%. Це в цілому говорить про стабільність роботи, адже з такими

показниками ми не будемо отримувати помилки певного виду при класифікації окремої загрози, тобто результати роботи стали більш збалансованими. В контексті аналізу ризиків це надзвичайно важливо: якби модель мала слабке місце, зловмисник, знаючи його, теоретично міг би підлаштувати атаку так, щоб ймовірність виявлення була найменшою. У нашому випадку важко казати про конкретні слабкі місця, можна говорити тільки про подальше покращення точності.

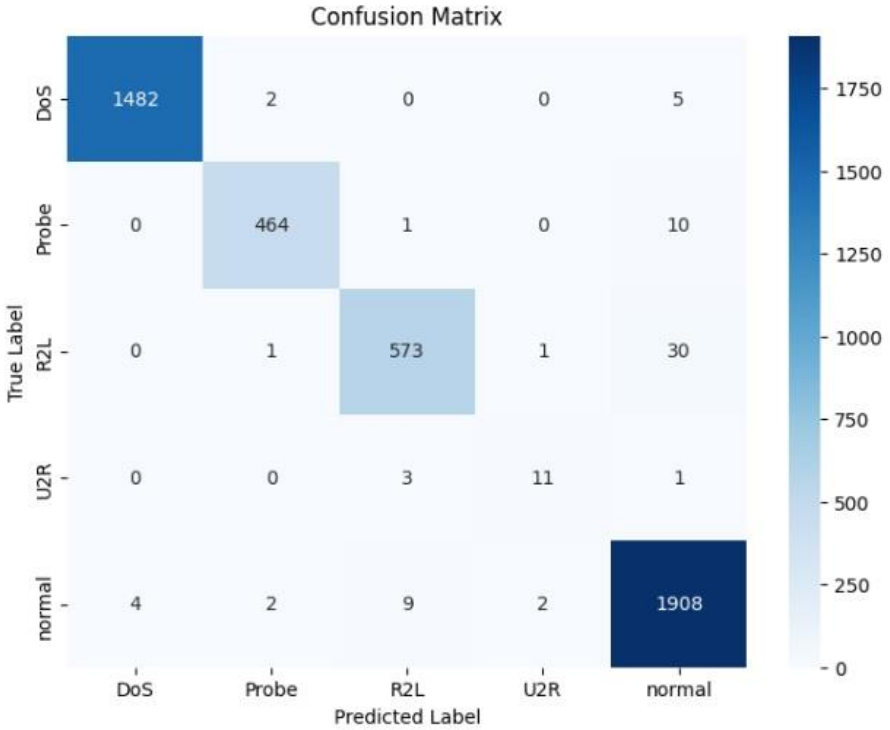

Рисунок 4.10. Матриця неузгодженостей

Одним з варіантів дослідити помилки моделей є матриця неузгодженостей (рис.4.10). Матриця неузгодженості (confusion matrix) є інструментом в оцінці результатів класифікації моделі машинного навчання. Вона використовується для аналізу того, наскільки ефективно модель класифікує екземпляри в різні класи. Як бачимо, більшість значень знаходяться на головній діагоналі, що говорить про правильність роботи. Інші значення є незначними і не можуть говорити про наявність конкретного слабкого місця в роботі, однак їх все ще можна враховувати для покращення результатів.

Висновки до розділу 4

В данному розділі було проведено повне тестування розроблено у другому розділі моделі. Спочатку були підготовлені базові моделі і дані для них. Використовувався NSL-KDD датасет, який якнайкраще підходить для аналізу у нашому випадку. Було повністю досліджено його та дані, які він містить. Усі висновки робилися послідовно крок за кроком, щоб забезпечити коректний аналіз і тренування базових моделей.

Аналогічно представленню у другому розділі, базові моделі працювали з різними частинами датасету і мали різні параметри. Зосередження кожної моделі на бінарній класифікації в результаті позитивно вплинуло на точність мета-мета-моделі. Було досліджено 5 метрик і для кожної нова модель показала високий приріст значення.

5 РОЗРОБКА СТАРТАП ПРОЕКТУ

5.1 Опис ідеї проекту

У даній магістерській роботі представлено модель для аналізу ризиків безпеки КС. Розроблено ПЗ виявлення і прогнозування загроз на її основі. Ідея проекту заключається у розробці подібних систем аналізу ризику з розширеним функціоналом. Для більш детального опису запропонованої ідеї наведено Табл.5.1

Таблиця 5.1 – Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
ПЗ виявлення і прогнозування ризиків порушення захищеності КС, на основі даних, отриманих з неї	1. IDS	Можливість виявлення вторгнень на основі аналізу комп’ютерної мережі.
	2. Аналізатори мережі	Покращення роботи мережевих аналізаторі або доповнення їх функціоналу.
	3. Захист окремих додатків	Використовуючи розроблену модель і досліджуючи трафік, створений конкретним додатком, можна оцінити його захизеність
	4. Антивірусний захист	Здатність розробленого ПЗ працювати у режимі реального часу

Наступний етап передбачає оцінку сильних і слабких сторін системи для визначення потенціалу впровадження на ринок та конкурентоспроможності. Відповідна інформація наведена в Таблиці 5.2.

Таблиця 5.2 – Сильні, слабкі та нейтральні характеристики проекту

Техніко-економічні характеристики ідеї	Потенційні товари/концепції конкурентів			Недоліки	Нейтральні сторони	Переваги
	Поточний проект	Конкурент 1	Конкурент 2			
Форма виконання	Детермінований алгоритм	Детермінований алгоритм	Детермінований алгоритм		+	
Потреба у зборі даних	Так	Ні	Так	+		
Кросплатформеність	Так	Ні	Так			+
Універсальність	Так	Ні	Так			+

Перевагами запропонованого ПЗ є універсальність моделі, на якій він побудований, але недоліки в тому, що модель треба тренувати на нових даних при боротьбі з новими ризиками.

5.2 Технічний аудит ідеї

В даному розділі буде проведено аудит технологій, що можуть бути використані для реалізації запропонованої в дипломній роботі ідеї (таблиця 5.3).

Таблиця 5.3 – Технологічна здійсненість проекту

Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
ПЗ виявлення і прогнозування ризиків порушення захищеності КС, на основі даних, отриманих з неї	Python	Наявна	Безкоштовна, доступна
	C++	Наявна	Безкоштовна, доступна
	C#	Наявна	Безкоштовна, доступна
	R	Наявна	Безкоштовна, доступна
	Java	Наявна	Безкоштовна, доступна
	Go	Наявна	Безкоштовна, доступна
	Julia	Наявна	Безкоштовна, доступна

Усі можливі запропоновані мови програмування і технології (бібліотеки, фреймворки), побудовані на них є безкоштовними та доступними для використання в наукових цілях.

Таким чином, об’єднання Python, C++, C#, R, Java, Go та Julia забезпечує комплексний інструментарій для реалізації запропонованого в магістерській роботі алгоритму.

5.2.1 Аналіз ринкових можливостей

Аналіз ринку має першорядне значення у формуванні стратегічного напрямку розвитку стартапу. Він дозволяє отримати уявлення про галузевий ландшафт, вподобання споживачів та позиціонування конкурентів. Проводячи ретельний

аналіз ринку, можна визначити потенційні можливості для зростання, виявити незайняті ніші та розробити ефективні маркетингові стратегії, адаптовані до своєї цільової аудиторії. У Таблиці 5.4 наведено попередню характеристику потенційного ринку для стартап-проекту.

Таблиця 5.4 – Попередня характеристика потенційного ринку для стартап-проекту

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	10
2	Загальний обсяг продаж, грн/ум.од	5000
3	Динаміка ринку (якісна оцінка)	Позитивна, зростає
4	Наявність обмежень для входу	Відсутні
5	Специфічні вимоги до стандартизації та сертифікації	Відсутні
6	Середня норма рентабельності в галузі (або по ринку), %	15%

Можемо зробити висновок, що проект має високу рентабельність.

Далі визначено перелік потенційних груп клієнтів, їх характеристики, та сформовано орієнтовний перелік вимог для кожної групи,наведено у Табл 5.5.

Таблиця 5.5 – Характеристика потенційних клієнтів стартап-проекту

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Необхідність боротьби з ризиками	Прості користувачі	Цікавить доступність	Зручність у використанні
Необхідність розвитку та поширення	Малі компанії	Потреба в можливості підтримки ПЗ	Наявність постійних оновлень
Необхідність бачити результат	Великі компанії	Цікавить надійність	Висока ефективність

Таким чином, основними вимогами до продукту є надійність, точність, зручність, оновлюваність.

Розглянемо фактори загроз та можливих реакцій на них у Табл.5.6.

Таблиця 5.6 – Фактори загроз

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Зростаюча вимогливість покупців	Зростання вимог до ефективності роботи алгоритму та точності	Проведення досліджень для покращення алгоритму

Таблиця 5.6 (продовження)

2	Збільшення витрат на технічну підтримку	Застарілість алгоритму відносно розвитку індустрії	Своєчасне оновлення алгоритму
3	Конкуренція	Вихід на ринок великої компанії	Запровадження нових стратегій
4	Зменшення кількості замовників	Зменшення зацікавленості замовників у наданих сервісах	Наявність постійних і своєчасних оновлень
5	Різкий стрибок в розвитку зловмисницького ПЗ	Неактуальність алгоритму порівняно з новими загрозами	

Було визначено основні фактори ризику для стартапу, серед яких найбільш значущим викликом є зміна вимог користувачів та революція в розвитку шкідливого ПЗ. Хоч ці ризики і малоймовірні, все одно для їх подолання необхідно своєчасно додавати нові можливості для алгоритму та слідкувати за змінами потреб в індустрії. Розглянемо фактори можливостей в Табл. 5.7.

Таблиця 5.7 – Фактори можливостей

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Зростання попиту на програмний продукт	Зростання попиту на програмний продукт за рахунок ефективності та надійності	Популяризація програмного продукту

Таблиця 5.7 (продовження)

2	Зменшення попиту до продукту конкурентів	Програмні продукти конкурентів почали втрачати свою актуальність у зв'язку з низкою факторів	Врахувати помилки конкурентів. Покращити свій програмний продукт
3	Зменшення витрат на підтримку програмного продукту	Автоматизація процесів підтримки та тестування програмного продукту	Займатись автоматизацією процесів в компанії та підвищувати кваліфікацію співробітників

Можна зробити висновок, що ключевою точкою для розвитку можливостей поширення програмного продукту та утримання цільової аудиторії користувачів є підтримка стабільної роботи продукту та своєчасна реакція на зміни ринку та потреби користувачів.

Проведемо ступеневий аналіз конкуренції на ринку. Наведено в Табл.5.8.

Таблиця 5.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Вказати тип конкуренції - олігополія	Мізерна кількість конкурентів на ринку	Врахування недоліків конкурентів. Покращення технології програмного продукту.

Таблиця 5.8 (продовження)

2. За рівнем конкурентної - інтернаціональна	Більшість компаній-конкурентів іноземні.	Врахування недоліків конкурентів. Покращення технології програмного продукту.
3. За галузевою ознакою - внутрішньогалузева	Сервіси використовуються у вузьких рамках однієї сфери	Спробувати поширити використання програмного продукту на інші сфери
4. Конкуренція за видами товарів: - товарновидова	Відмінність у програмному продукті, але схожість у функціях	Покращувати показники роботи алгоритму
5. За характером конкурентних переваг - нецінова	Висока собівартість програмного продукту	Збільшення функціоналу програмного продукту
6. За інтенсивністю - не марочна	Відсутність бренду	Побудова бренду компанії. Вихід на ринок з високою впізнаваністю.

Далі потрібно виконати аналіз конкуренції за моделлю 5 сил конкуренції Майкла Портера. Наведено в Табл. 5.9. Дана модель є стратегічним інструментом, призначеним для аналізу впливу конкурентного середовища на підприємство чи галузь. Майкл Портер виділяє п'ять основних сил, які визначають конкуренцію в даній галузі: загроза нових учасників (нові конкуренти), сила переговорів покупців, сила переговорів постачальників, загроза заміщення продуктів чи послуг, кількість та сила конкурентів в галузі.

5 сил Майкла Портера допомогають підприємствам зрозуміти динаміку конкурентного середовища та визначити стратегічні кроки для забезпечення стійкості та конкурентоспроможності.

Таблиця 5.9 – Аналіз конкуренції в галузі за М. Портером

Складові аналізу	Прямі конкуренти у галузі	Потенційні конкуренти	Постачальники	Клієнти	Товарозамінники
	Інші існуючі системи та продукти	Якість, ціни, кількість користувачів, капіталовкладення	Фактори сили постачальників	Контроль якості, порівняння цін	Сила бренду, якість, ціна, масштаби
Висновки	Конкуренція з невеликою інтенсивністю, а також підігрітий ринок	Можливості входження на ринок, нові потенційні конкуренти	Постачальники відсутні	Клієнти не диктують умови роботи на ринку	Товарозамінники відсутні

Отримавши результати аналізу конкуренції (таблиця 5.2), характеристики ідеї стартап-проекту (таблиця 5.5), характеристики потенційних клієнтів і їх вимоги до продукту (таблиця 5.6) , а також фактори ринкового середовища (таблиці 5.7 і 5.8) було сформульовано та обґрунтовано перелік факторів конкурентоспроможності (таблиця 5.10).

Таблиця 5.10 – Обґрунтування факторів конкурентоспроможності

№ п/п	Фактор конкурентоспроможності	Обґрунтування (наведення чинників, що роблять фактор для порівняння конкурентних проектів значущим)
1	Універсальність	Модель можна навчати на будь-яких даних
2	Простота у використанні	Треба лише навчитися правильно користуватися інтерфейсом
3	Якість та гарантії	Надавати найбільш якісні послуги та сервіси
4	Безкоштовний сервіс при MVP	Заявити про себе на ринку максимально швидко набравши базу клієнтів

Проведемо порівняльний аналіз слабких та сильних сторін, для наведених вище факторів.Табл. 5.11

Таблиця 5.11 – Сильні та слабкі сторони стартап-проекту

№ п/п	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів						
			-3	-2	-1	0	1	2	3
1	Універсальність	20	+						
2	Простота у використанні	16		+					
3	Якість та гарантії	12		+					
4	Безкоштовний сервіс при MVP	15			+				

Наступним кроком проведем SWOT аналіз (Табл. 5.12).

Таблиця 5.12 – SWOT- аналіз стартап-проекту

Сильні сторони: інноваційний алгоритм	Слабкі сторони: відсутність впізнаваності та бренду
Можливості: стрімкий розвиток індустрії	Загрози: конкуренти більш впізнавані і мають більше можливостей, як кадрових, так і фінансових.

За допомогою SWOT - аналізу було виявлено сильні та слабкі сторони стартап-проекту. А саме - новизна запропонованої моделі, а слабкою стороною є відсутність бренду та його впізнаваності взагалі. Стрімкий розвиток індустрії надає можливості для розвитку програмного продукту, проте, це є і загрозою, адже існують більші компанії. Вони мають великий штат інженерів та більше технічні, технологічні та фінансові можливості.

Складемо альтернативи впровадження проекту на основі SWOT аналізу. Наведено в Таблиці 5.13

Таблиця 5.13 – Альтернативи ринкового впровадження стартап-проекту

№ п/п	Альтернатива (орієнтовний комплекс заходів) ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Суттєве покращення алгоритму	70%	18 місяців
2	Розвиток бренду та підвищення впізнаваності	30%	9 місяців

У даному етапі було проведено ретельний аналіз ринку та продукту. Враховуючи результати конкурентного аналізу, визначені фактори ринку та його сприятливість, а також опис ідеї та характеристик стартап-проекту, ми робимо висновок, що умови для виходу продукту на ринок є досить сприятливими.

5.3 Розробка ринкової стратегії проекту

Розробка ринкової стратегії проекту починається з визначення визначення цільової аудиторії проекту. Почнемо з визначення цільових груп. Розглянуто в Табл.5.14

Таблиця 5.14 – Вибір цільових груп потенційних споживачів

№ п/п	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Складність входу у сегмент
1	Прості користувачі	Висока	Середній	Не інтенсивна	Середня
2	Малі компанії	Висока	Високий	Інтенсивна	Середня
3	Великі компанії	Висока	Високий	Інтенсивна	Середня

Було обрано цільові групи: будь-які великі та малі компанії, які потребують захисту їхньої програмної інфраструктури. Маючи це представлення, визначимо базову стратегію розвитку стартап проекту. Наведено в Табл.5.14.

Таблиця 5.15 – Визначення базової стратегії розвитку

№ п/п	Обрана альтернатива розвитку проекту	Стратегія охопленн яринку	Ключові конкуренто-спроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
1	Великі та малі компанії, що потребують кіберзахисту інфраструктури	Диференційованого маркетингу	Масштабування та максимізація	Оптимальних витрат

Далі буде описано вибір стратегії конкурентної поведінки, наведено в Табл. 5.16.

Таблиця 5.16 – Визначення базової стратегії розвитку

№ п/п	Чи є проект «першопроходцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
1	Ні	Так	Надійність, точність	Оборонна

Визначимо стратегію позиціонування проекту, що допоможе користувачам ідентифікувати програмний продукт Табл. 5.17.

Таблиця 5.17 – Визначення стратегії позиціонування

№ п/п	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкуренто-спроможні позиції власного стартап-проекту	Вибір асоціацій, які мають сформувати комплексну позицію власного проекту (три ключових)
1	Надійність, то-чність, просто-та використан-ня	Оптимальних витрат	Якість та ефективність	Надійність, то-чність, простота використання

У даному розділі було визначено стратегію позиціонування програмного продукту, ними є ефективність, точність та простота використання алгоритму.

5.3.1 Розробка маркетингової програми

Для того щоб виконати розробку маркетингової програми повною мірою, найперше- потрібно визначити маркетингову концепцію товару, який пропонується споживачу. У Табл.5.18 наведено підсумок аналізу конкурентоспроможності товару, що була наведена вище.

Таблиця 5.18 – Визначення ключових переваг концепції потенційного товару

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити
1	Якісний захист від загроз	Якісні результати прогнозування	Постійне покращення та перенавчання моделей, які зможуть аналізувати більше прикладних областей
2	Універсальність	Модель можна перенавчати	Такого виду модель можна навчити аналізувати загрози на будь-яких правильно оброблених даних
3	Підтримка	Система буде оновлюватись часто та залежно від потреб користувачів	Зазист системи буде актуальним постійно

Розробимо трирівневу маркетингову модель товару. Табл. 5.19.

Таблиця 5.19 – Визначення ключових переваг концепції потенційного товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Створення ПЗ виявлення і прогнозування ризиків безпеки КС		
II. Товар у реальному виконанні	Властивості / характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	Надійність Точність Простота	М М М	Тл Тл Тл
	Використовує унікальну комбінацію моделей ML і їх параметрів		

Визначимо цінові межі за допомогою експертного метода. Табл. 5.20

Таблиця 5.20 – Визначення меж встановлення ціни

№ п/п	Рівень цін на товари- замінники	Рівень цін на товари- аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	5000$/рік	20000$/рік	200 000$/рік	3000 – 6000$/рік

Визначимо оптимальну систему збуту. Табл.5.21.

Таблиця 5.21 – Формування системи збуту

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
1	Постачання товару з усіма необхідними засобами	Просте встановлення і оплата послуг	Розробник - користувач	Канал збуту одного рівня

Остання частина – це побудова концепції маркетингової комунікації, що спирається на попередньо обрану основу для позиціонування, визначену специфіку поведінки клієнтів. В таблиці 5.22 зобразимо концепцію маркетингових комунікацій.

Таблиця 5.22 – Формування системи збуту

№ п/п	Специфіка поведінки цільових клієнтів	Канали комуніка- цій, якими користу- ються цільові клієнти	Ключові позиції, обрані для позиціону- вання	Завдання рекламного повідомлення	Концепція рекламного звернення
1	Клієнти шукають нові засоби аналізу ризиків	Конференції, Інтернет	Ефективність, підтримка, універсальність	Демонстрація найкращих показників	Швидкий аналіз на виявлення атак

Висновки до розділу 5

У цьому розділі було розглянуто маркетинговий аналіз – важливий етап розробки стартап-проекту. Було визначено основну ідею роботи, проведено технологічний аудит, в якому розглянуто доцільність використання різних технологій. Було проведено аналіз ринку, аналіз ризиків та можливостей, визначено сильні та слабкі сторони. Також був проведений аналіз цільової аудиторії. На основі всіх цих даних була розроблена стратегія виходу на ринок та маркетингова програма.

ВИСНОВКИ

У представленій магістерській дисертації було розглянуто задачу аналізу ризиків інформаційої безпеки. В першу чергу було проведено загальний аналіз традиційних методів захисту інформації комп'ютерних систем, визначено та класифіковано ризики для інформаційної безпеки. Було розглянуто різні види аналізу ризиків в КС. Виявлено, що алгоритми машинного навчання мають найбільший потенціал у цій сфері.

Створено модифіковану модель для їх виявлення і прогнозування на основі Ensemble Stacking, що є потужним методом ML. Запропонована модель базується на комбінації сильних і вдало підібраних алгоритмів класифікації та їх параметрів, адже саме на великих даних розкриваються переваги кожного алгоритму, що входить у стек. У якості мета-моделі було вибрано логістичну регресію, оскільки вона є простою та інтерпретованою моделлю, що дозволяє аналізувати ваги кожного параметра.

На основі натренованої моделі було створено програму типу IDS, яка здатна отримувати дані з КС для їх подальшого аналізу і передбачення загроз. Крім того, модель було успішно поєднано із уже існуючим програмним забезпеченням – мережевими аналізаторами. Більшість подібних програм можуть ефективно витягувати дані з трафіку КС, але зазвичай не застосовують сучасні методи аналізу на предмет загроз. Розроблена модель була перевірена на практиці і продемонструвала ефективні результати.

Крім того, вона має широку сферу застосування, так як потреба у кіберзахисті існує скрізь, де є комп’ютерні системи. ПЗ, яке потенційно може застосовувати розроблену модель, включає системи виявлення вторгнень, системи боротьби зі спамом, засоби антивірусного захисту.

У даній роботі об’єктом дослідження стали дані NSL-KDD датасету, оскільки вони найкраще описують існуючі загрози безпеці КС на рівні інцидентів у мережевому трафіку. Мета-модель під час тестування на цих даних показала значно кращі результати за усіма метриками, ніж її вхідні алгоритми, що говорить про правильність вибору всіх параметрів та методів обробки даних.

Однак необхідно врахувати, що жодна система не є абсолютною чи повністю невразливою. Хоча мета-модель і показала гарний результат, і її можна навчати на будь-якому відповідно обробленому наборі даних, потрібно завжди враховувати інші аспекти, такі як соціальна інженерія, внутрішні загрози, непередбачувані обставини. У сучасному цифровому світі для підтримки безпеки необхідний комплексний підхід, який об'єднує технічні і людські аспекти.

ПЕРЕЛІК ПОСИЛАНЬ

1. What is CVE and CVSS. URL: https://www.benq.com/en-us/business/resource/trends/what-is-cve-and-cvss.html

2. Ensemble Stacking for Machine Learning and Deep Learning. URL: https://www.analyticsvidhya.com/blog/2021/08/ensemble-stacking-for-machine-learning-and-deep-learning/

3. A Machine Learning-based Tool for Cybersecurity Risk Assessment URL: https://www.researchgate.net/publication/365418936_SecRiskAI_a_Machine_Learning-Based_Approach_for_Cybersecurity_Risk_Prediction_in_Businesses

4. Introduction Random Forest Classification By Example. URL: https://medium.com/@mrmaster907/introduction-random-forest-classification-by-example-6983d95c7b91

5. Information Security Risk Analysis, Second Edition, Thomas R. Peltier. URL: https://www.academia.edu/40140524/Information_security_risk_analysis_thomas_r_peltier

6. Ievgeniia Kuzminykh , Bogdan Ghita Security Risk Assessment. URL: https://www.researchgate.net/publication/353436973_Information_Security_Risk_Assessment#fullTextFileContent

7. Mouna Jouinia Classification of security threats in information systems. URL: https://www.researchgate.net/publication/315714820_Classification_of_security_threats_in_information_systems#fullTextFileContent

8. Analysis of Computer Network Security Problems and Countermeasures. URL: https://www.researchgate.net/publication/320177233_Analysis_of_Computer_Network_Security_Problems_and_Countermeasures

9. Intrusion Detection VS Prevention Systems: What’s The Difference? URL: https://purplesec.us/intrusion-detection-vs-intrusion-prevention-systems/

10. Implementation of Combined Machine Learning Method for IDS. URL: https://www.researchgate.net/publication/326983606_Implementation_and_Analysis_of_Combined_Machine_Learning_Method_for_Intrusion_Detection_System

11. Cyber-Risk Forecasting using Machine Learning Models and Generalized

Extreme Value Distributions. URL: https://hal.science/hal-03814979/document

12. Mete Eminagaoglu A Qualitative Information Security Risk Assessment Model using Machine Learning Techniques URL: https://www.researchgate.net/publication/258769344_A_Qualitative_Information_Security_Risk_Assessment_Model_using_Machine_Learning_Techniques

13. Cyber-Risk Forecasting using Machine Learning Models and Generalized Extreme Value Distributions URL: https://hal.science/hal-03814979/document

14. A P Chukhnov, Y S IvanovAlgorithms for Detecting and Preventing Attacks on Machine Learning Models in Cyber-Security Problems URL: https://iopscience.iop.org/article/10.1088/1742-6596/2096/1/012099/pdf

15. Research on threat detection in cyber security based on machine learning URL: https://iopscience.iop.org/article/10.1088/1742-6596/2113/1/012074/pdf

16. Mohammad Al-Fawareh Cyber threat intelligence using PCA-DNN model to detect abnormal network behavior URL: https://pdf.sciencedirectassets.com/280281/1-s2.0-S1110866522X00030/1-s2.0-S1110866521000785/main.pdf

17. Cyber-attack detection in network traffic using machine learning URL: https://repository.rit.edu/theses/11320/

18. Building Machine Learning-based Threat Hunting System from Scratch URL: https://dl.acm.org/doi/full/10.1145/3491260

19. A Review of Insider Threat Detection: Classification, Machine Learning Techniques, Datasets, Open Challenges, and Recommendations URL: https://www.mdpi.com/2076-3417/10/15/5208

20. Аналіз ризиків в задачах інформаційної безпеки. / Северин М.С., Мухін В.Є.// Системні науки та інформатика: збірник доповідей II науково-практичної конференції з нагоди 125-річчя КПІ ім. Ігоря Сікорського «Системні науки та інформатика», 4–8 грудня 2023 року, Київ. – К., НН ІПСА КПІ ім. Ігоря Сікорського, 2023. – с. 347-352.

ДОДАТОК А. ЛІСТИНГИ ПРОГРАМИ

Analyzer.py

```
from scapy.all import sniff, wrpcap, rdpcap
import tkinter as tk
from tkinter import ttk, Entry, messagebox
import pandas as pd
import pickle

df = pd.DataFrame()  # Initialize an empty DataFrame

def packet_callback(packet):
    if packet.haslayer('IP'):
        print(packet.summary())

def start_scan():
    global df
    captured_packets = sniff(prn=packet_callback, timeout=10, lfilter=lambda x: x.haslayer('IP'))
    wrpcap('captured_traffic.pcap', captured_packets)

    pcap_file_path = 'captured_traffic.pcap'
    packets = rdpcap(pcap_file_path)

    duration, src_bytes, dst_bytes, land, wrong_fragment, urgent, num_failed_logins = ([] for _ in
range(7))
    logged_in, root_shell, su_attempted, num_root, num_file_creations, num_shells, num_access_files =
([] for _ in range(7))
    is_host_login, is_guest_login, count, srv_count, srv_serror_rate = ([] for _ in range(5))
    srv_rerror_rate, same_srv_rate, diff_srv_rate, srv_diff_host_rate, dst_host_count = ([] for _ in
range(5))
    dst_host_srv_count, dst_host_diff_srv_rate, dst_host_same_src_port_rate = ([] for _ in range(3))
    dst_host_srv_diff_host_rate, protocol_type_icmp, protocol_type_tcp = ([] for _ in range(3))
    true_labels, predicted_labels = ([] for _ in range(2))

    with open('stacking_model.pkl', 'rb') as model_file:
```

```
    stacking_model = pickle.load(model_file)

for packet in packets:
    raw = packet
    duration.append(packet.time)
    src_bytes.append(len(packet.payload))
    dst_bytes.append(len(packet.payload.payload))
    land.append(int(packet['IP'].fields['options'] == b'\x02\x04\x04\x04'))
    wrong_fragment.append(1 if 'DF' in packet['IP'].flags else 0)
    urgent.append(1 if 'RF' in packet['IP'].flags else 0)
    num_failed_logins.append(0)
    logged_in.append(0)
    root_shell.append(0)
    su_attempted.append(0)
    num_root.append(0)
    num_file_creations.append(0)
    num_shells.append(0)
    num_access_files.append(0)
    is_host_login.append(0)
    is_guest_login.append(0)
    count.append(1)
    srv_count.append(0)
    srv_serror_rate.append(0)
    srv_rerror_rate.append(0)
    same_srv_rate.append(1)
    diff_srv_rate.append(0)
    srv_diff_host_rate.append(0)
    dst_host_count.append(1)
    dst_host_srv_count.append(0)
    dst_host_diff_srv_rate.append(0)
    dst_host_same_src_port_rate.append(1)
    dst_host_srv_diff_host_rate.append(0)
    protocol_type_icmp.append(1 if packet.haslayer('ICMP') else 0)
    protocol_type_tcp.append(1 if packet.haslayer('TCP') else 0)
    true_labels.append(0)
    predicted_labels.append(0)
```

```
data = {
    'duration': duration,
    'src_bytes': src_bytes,
    'dst_bytes': dst_bytes,
    'land': land,
    'wrong_fragment': wrong_fragment,
    'urgent': urgent,
    'num_failed_logins': num_failed_logins,
    'logged_in': logged_in,
    'root_shell': root_shell,
    'su_attempted': su_attempted,
    'num_root': num_root,
    'num_file_creations': num_file_creations,
    'num_shells': num_shells,
    'num_access_files': num_access_files,
    'is_host_login': is_host_login,
    'is_guest_login': is_guest_login,
    'count': count,
    'srv_count': srv_count,
    'srv_serror_rate': srv_serror_rate,
    'srv_rerror_rate': srv_rerror_rate,
    'same_srv_rate': same_srv_rate,
    'diff_srv_rate': diff_srv_rate,
    'srv_diff_host_rate': srv_diff_host_rate,
    'dst_host_count': dst_host_count,
    'dst_host_srv_count': dst_host_srv_count,
    'dst_host_diff_srv_rate': dst_host_diff_srv_rate,
    'dst_host_same_src_port_rate': dst_host_same_src_port_rate,
    'dst_host_srv_diff_host_rate': dst_host_srv_diff_host_rate,
    'protocol_type_icmp': protocol_type_icmp,
    'protocol_type_tcp': protocol_type_tcp,
    'True_Labels': true_labels,
    'Predicted_Labels': predicted_labels
}
```

```
    df = pd.DataFrame(data)

    # Update the Treeview widget with the new DataFrame
    update_treeview()

def update_treeview():
    global df
    tree.delete(*tree.get_children())

    for i, row in df.iterrows():
        tree.insert("", i, values=tuple(row))
    root.update_idletasks()

def detect_threat():
    threat_types = ['DoS', 'R2L', 'Probe', 'U2R']
    detected_threats = []
    print(df.head)
    for threat_type in threat_types:
        if any(df['Predicted_Labels'] == threat_type):
            detected_threats.append(threat_type)

    if detected_threats:
        message = f"Malicious traffic detected. Threats: {', '.join(detected_threats)}"
        messagebox.showwarning("Threat Detected", message)
    else:
        messagebox.showinfo("No Threats", "Traffic is safe.")

# GUI setup
root = tk.Tk()
root.title("Packet Data Viewer")
root.geometry("800x400")

frame = ttk.Frame(root)
frame.pack(expand=True, fill='both', padx=10, pady=10)

tree = ttk.Treeview(frame)
```

```
tree["columns"] = tuple(df.columns)
for col in df.columns:
    tree.column(col, anchor="center", width=100)
    tree.heading(col, text=col, anchor="center")

tree.pack(expand=tk.YES, fill=tk.BOTH)

x_scrollbar = ttk.Scrollbar(frame, orient="horizontal", command=tree.xview)
x_scrollbar.pack(side=tk.BOTTOM, fill="x")
tree.configure(xscrollcommand=x_scrollbar.set)

other_frame = ttk.Frame(root)
other_frame.pack(expand=False, fill='both', pady=10)

protocol_type_icmp_var = tk.IntVar()
protocol_type_icmp_checkbox = ttk.Checkbutton(other_frame, text="ICMP",
variable=protocol_type_icmp_var)
protocol_type_icmp_checkbox.grid(row=0, column=0, padx=5, pady=5)

protocol_type_tcp_var = tk.IntVar()
protocol_type_tcp_checkbox = ttk.Checkbutton(other_frame, text="TCP",
variable=protocol_type_tcp_var)
protocol_type_tcp_checkbox.grid(row=0, column=1, padx=5, pady=5)

protocol_type_udp_var = tk.IntVar()
protocol_type_udp_checkbox = ttk.Checkbutton(other_frame, text="UDP",
variable=protocol_type_udp_var)
protocol_type_udp_checkbox.grid(row=0, column=2, padx=5, pady=5)

duration_label = ttk.Label(other_frame, text="Scan Duration (seconds):")
duration_label.grid(row=1, column=0, padx=5, pady=5, sticky=tk.E)

duration_entry = Entry(other_frame)
duration_entry.grid(row=1, column=1, padx=5, pady=5)

num_packets_label = ttk.Label(other_frame, text="Max Number of Packets:")
```

```
num_packets_label.grid(row=1, column=2, padx=5, pady=5, sticky=tk.E)

num_packets_entry = Entry(other_frame)
num_packets_entry.grid(row=1, column=3, padx=5, pady=5)

scan_button = ttk.Button(other_frame, text="Scan", command=start_scan)
scan_button.grid(row=2, column=0, columnspan=2, pady=10)

detect_button = ttk.Button(other_frame, text="Detect Threat", command=detect_threat)
detect_button.grid(row=2, column=2, columnspan=2, pady=10)

root.mainloop()
```