МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

КАФЕДРА ЕЛЕКТРОНІКИ, ІНФОРМАЦІЙНИХ СИСТЕМ ТА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

Кваліфікаційна робота

другий (магістерський)
(рівень вищої освіти)

на тему **Хостова система виявлення та обробки вторгнень з використанням машинного навчання**

Виконав: студент 2 курсу, групи 8.1211-іпз-2
спеціальності 121 Інженерія програмного забезпечення
(код і назва спеціальності)
освітньої програми Інженерія програмного забезпечення
(код і назва освітньої програми)
О.О. Хаджийський
(підпис, ініціали та прізвище)
Керівник к. т. н., доцент Н.П. Полякова
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)
Рецензент директор ТОВ Дісітел П.О. Лютий
(посада, вчене звання, науковий ступінь, підпис, ініціали та прізвище)

Запоріжжя
2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ІНЖЕНЕРНИЙ НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ
ім. Ю.М. Потебні
ЗАПОРІЗЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ

Кафедра Електроніки, інформаційних систем та програмного забезпечення

Рівень вищої освіти _____другий (магістерський)_____

Спеціальність ___121_Інженерія програмного забезпечення_____
(код та назва)

Освітня програма ___Інженерія програмного забезпечення
(код та назва)

ЗАТВЕРДЖУЮ

Завідувач кафедри_____Т.В. Критська_____

“_12_” __вересня_____2022_року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Хаджийський Олег Олександрович
(прізвище, ім’я, по батькові)

1.Тема роботи __Хостова система виявлення та обробки вторгнень з використанням машинного навчання_____

керівник роботи ___Н.П. Полякова, к. т. н., доцент
(прізвище, ім’я, по батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від 02.06.2022 р. №597-с

2. Строк подання студентом кваліфікаційної роботи 5 грудня 2022 р.

3. Вихідні дані магістерської роботи

- комплект нормативних документів ;
- технічне завдання до роботи.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

- огляд та збір літератури стосовно теми кваліфікаційної роботи;
- огляд та аналіз існуючих рішень та аналогів;
- дослідження проблеми розпізнавання мов та розробка методів її вирішення;
- створення програмного продукту та його опис;
- перелік вимог для роботи програми;
- дослідження поставленої проблеми та розробка висновків та пропозицій.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)

12 слайдів презентації

6. Консультанти розділів магістерської роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата Завдання прийняв

7. Дата видачі завдання 01.09.2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської роботи	Строк виконання етапів магістерської роботи	Примітка
1	Аналіз предметної області	01.09-15.09.2022	виконано
2	Формулювання основної задачі дипломної роботи та узгодження її з науковим керівником	15.09-20.09.2022	виконано
3	Аналіз існуючих методів виявлення та обробки мережевих загроз та алгоритмів класифікації трафіку	25.09-30.09.2022	виконано
4	Узгодження подальших дій з науковим керівником	30.09-05.10.2022	виконано
5	Розробка ігрового застосунку	05.10-06.10.2022	виконано
6	Дослідження існуючих рішень й бібліотек для захоплення трафіку з мережевої карти комп'ютера	06.10-10.10.2022	виконано
7	Представлення отриманих результатів науковому керівнику та узгодження плану подальшого дослідження	10.10-11.10.2022	виконано
8	Розробка модуля з екстракції необхідних параметрів трафіку	11.10-20.10.2022	виконано
9	Розробка модуля з навчання моделі виявлення та обробки мережевих загроз	20.10-10.11.2022	виконано
10	Створення, навчання, перевірка моделі	10.11-15.11.2022	виконано
11	Представлення отриманих результатів науковому керівнику та узгодження плану подальшого дослідження	09.11-20.11.2022	виконано
12	Реалізація користувацького інтерфейсу для комп'ютерної системи	20.11-25.12.2022	виконано
13	Оформлення звіту	25.11-01.12.2022	виконано

Студент _____ О.О. Хаджийський
(підпис) (ініціали та прізвище)

Керівник роботи _____ Н.П. Полякова
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ І.А. Скрипник
(підпис) (ініціали та прізвище)

АНОТАЦІЯ

Сторінок: 70

Рисунків: 39

Таблиць: 3

Джерел: 12

Хаджийський О.О. Хостова система виявлення та обробки вторгнень з використанням машинного навчання : кваліфікаційна робота магістра спеціальності 121 «Інженерія програмного забезпечення» / наук. керівник Н. П. Полякова. Запоріжжя : ЗНУ, 2022. с.

Мета і завдання дослідження полягають у вивченні сучасних та ефективних підходів виявлення та обробки загроз для користувача персонального комп'ютера, а також у розробці комп'ютерної системи виявлення вторгнень, що буде ефективно вирішувати цю задачу, працюючи на комп'ютері та використовуючи в якості вхідних даних потік із мережевої карти комп'ютера.

У процесі дослідження була розглянута проблематика задачі виявлення

та обробки загроз для користувача персонального комп'ютера. Результатом дослідження є розроблена комп'ютерна система, навчена виявляти зловмисне ПЗ аналізуючи мережевий потік. Поставлена проблема виявлення та обробки вторгнень з використанням машинного навчання вирішена з допустимою точністю та в межах обраних методів дослідження.

Ключові слова: *комп'ютерна система захисту, аналіз трафіку, виявлення вторгнень без розшифрування, машинне навчання, виявлення вірусів, sklearn, wireshark, random forest.*

SUMMARY

Pages: 70

Figures: 39

Tables: 3

Sources: 12

Khadzhyiskyi O.O. The host-based intrusion detection system using machine learning: qualifying thesis of the master's degree in the specialty 121 "Software engineering" / science. manager N. P. Polyakov. Zaporizhzhia: ZNU, 2022. p. The goal and objectives of the research are to study modern and effective approaches to detecting and processing threats for the user of a personal computer, as well as to develop a computer system for detecting intrusions that will effectively solve this problem, working on a computer and using as input data flow from the network card of the computer.

In the course of the research, the problems of detection were considered and threat processing for a personal computer user. The result of the research is a developed computer system trained to detect malicious software by analyzing network traffic. The posed problem of detection and processing of host-based intrusions is solved with acceptable accuracy and within the chosen research methods.

Keywords: *computer protection system, traffic analysis, intrusion detection without decryption, machine learning, virus detection, sklearn, wireshark, andom forest.*

ЗМІСТ

ВСТУП

Актуальність теми

Кібертероризм у всьому світі набирає обертів і особливо зараз, в 2022 році, спланована масована кібератака державних структур може паралізувати роботу стратегічних об’єктів життєзабезпечення, або точкова хакерська атака спрямована на конкретну людину може призвести до втрати конфіденційної інформації. Кожного дня виявляються та блокуються нові загрози. Це здійснюється засобами розслідування, реагування на інциденти та усунення проблем, як за допомогою апаратних засобів різного рівня так і за допомогою користувацьких застосунків. На рівні організацій зазвичай діють апаратні рішення разом із застосунками, які фільтрують мережевий трафік для захисту від внутрішніх та зовнішніх загроз. Але кожна сучасна людина має власний пристрій з підключенням до мережеі, який представляє собою кінцеву точку, яку слід захистити. Підхід до повноцінного захисту кінцевої точки поєднує превентивний захист рішення EPP, а також функції виявлення та дослідження EDR. Endpoint protection platform (EPP) — це профілактичний інструмент, який охоплює можливості захисту від шкідливих програм, використовуючи персональний брандмауер, що оперує правилами, наприклад, контроль портів і пристроїв, чи антивірус, який відповідає за більш широкий спектр захисту, та включає сканування системи на зловмисне ПЗ чи виявлення несанкціонованих дій на хості. Endpoint detection and response (EDR) — спектр рішень для виявлення активності шкідливого ПЗ на кінцевих точках. На відміну від антивірусів, які за допомогою сигнатур масово виявляють шкідливе ПЗ, EDR попереджає про атаку, спираючись на аналіз і дані для запобігання атаці, EDR-рішення орієнтовані на виявлення цільових атак та складних загроз. Але дві технології вирішують різні завдання. Головною функцією будь-якої хостової системи безпеки має бути здатність виявити трафік або поведінку шкідливого програмного забезпечення до того, як воно порушить конфіденційність інфо-

рмації на пристрої-жертві. Шкідливі програми використовують протокол захисту транспортного рівня TLS (transport layer security), наприклад, під час звернення до command-and-control (C&C) сервера, чим створюють проблему аналізу зашифрованого мережевого трафіку. Методи, що засновані на сигнатурному співставленні не вміють виявляти нові несанкціоновані дії, що не зустрічалися раніше, а існуючі процеси дешифрування та повторного шифрування займають багато часу, що викликає відчутні затримки в передачі даних. Більшість із них використовують функції, засновані на повному потоці, тобто аналізуючи трафік деякий час, вони знаходять ознаки які вже зустріли раніше та звітують про загрозу. В такому випадку є проблема зі своєчасністю таких рішень.

Таким чином, є актуальним створення рішення, що застосовує алгоритми класифікації та методи аналізу мережевих пакетів. Програмні системи, в яких задіяні такі алгоритмами, будуть здатні реагувати на загрозу ще до здійснення нею протиправних дій.

Мета і завдання дослідження

Мета і завдання дослідження полягають у вивченні сучасних та ефективних підходів виявлення загроз для користувача персонального комп’ютера, а також у розробці комп’ютерної системи виявлення вторгнень, що буде ефективно вирішувати цю задачу, працюючи на комп’ютері та використовуючи в якості вхідних даних потік із мережевої карти комп’ютера.

Об’єкт дослідження

Об’єктом дослідження є потік мережевих даних, що надходять на мережеву карту комп'ютера.

Предмет дослідження

Предметом дослідження є виявлення загроз для стану хоста шляхом аналізу мережевого трафіку в режимі реального часу або на заздалегідь перехоплених мережевих пакетах.

Методи дослідження

Для вирішення поставленої задачі проведемо предметне обмеження проблеми, використовуючи наступні методи дослідження:

1. Аналіз наукових досліджень для систем виявлення зловмисного ПЗ на комп'ютері.
2. Аналіз зловмисного ПЗ, що має за мету несанкціонований доступ до даних на хості.
3. Огляд існуючих рішень для виявлення та обробки вторгнень на хості.
4. Аналіз методів визначення вторгнень на хості, шляхом аналізу зашифрованого мережевого трафіку на хості.
5. Аналіз та обробка датасетів для моделі класифікатора трафіку.
6. Створення класифікатора для хостової системи виявлення вторгнень.
7. Навчання та тестування моделі.
8. Аналіз ефективності рішення, своєчасності, доцільності.

Наукова новизна одержаних результатів

Наукова новизна одержаних результатів дослідження полягає у тому, що для вирішення задачі визначення зловмисного трафіку був використаний актуальний та ефективний підхід для створення класифікатора трафіку, а саме: на основі методу навчання random forest, була створена та навчена оптимальна модель машинного навчання для класифікації мережевого трафіку, яка на основі вхідних даних робить прогноз щодо зловмисності трафіку на хості.

Практичне значення одержаних результатів

Практичне значення одержаних результатів полягає у тому, що була розроблена системи виявлення загроз для комп'ютера, яка шляхом аналізу мережевих пакетів робить висновок про шкідливість захопленого інтернет-трафіку. Дана система може навчатися на прикладах захопленої мережевих пакетах, тож здатна по поведінці ПЗ виявити ще невідомі загрози. Система є оптимальною з точки зору апаратних витрат, та часу навчання моделі. Можна зробити висновок, що представлений у роботі описаний та задіяний ефективний підхід для системи виявлення та обробки вторгнень, шляхом аналізу мережевих пакетів.

Апробація одержаних результатів

Результати дослідження були представлені на XIII науково-практичній конференції студентів, аспірантів, докторантів і молодих вчених Запорізького національного університету «Молода наука-2022» [01], а також на XXV науково-технічній конференції студентів, аспірантів, магістрантів і викладачів Інженерного навчально-наукового інституту Запорізького національного університету**Помилка! Джерело посилання не знайдено.**.

Глосарій

Захист на транспортному рівні, TLS (англ. Transport Layer Security) — це криптографічний протокол, що надає можливості безпечної передачі даних в мережі Інтернет. Використовує асиметричне шифрування і сертифікати X.509.

Система виявлення й обробки вторгнень (англ. Intrusion Detection System) — система виявлення й обробки вторгнень (атак), які мають можливості автоматизованої протидії атакам.

Міжмережевий екран, брандмауер, фаєрвол, (англ. Firewall) — загальна назва програм чи фізичних пристроїв, основною задачею яких є обробка мережевого трафіку згідно з бажаним набором правил безпеки.

Антивірусна програма, антивірус — спеціалізована програма для знаходження шкідливих програм, та відновлення заражених (модифікованих) такими програмами файлів, а також для профілактики — запобігання зараженню (модифікації) файлів чи операційної системи шкідливим кодом.

Модель — модель машинного навчання, яка на основі вхідних даних та наданого їй алгоритму навчається розпізнаванню певних типів закономірностей та прогнозування.

Виявлення аномалій (англ. Anomaly detection) — процес під час аналізу даних, під час якого спостерігаються виявлення рідкісних предметів, подій або спостережень, які значно відрізняються від більшості даних і не відповідають чітко визначеним поняття нормальної поведінки.

Виявлення зловживання (англ. misuse detection) — це підхід до виявлення комп'ютерних атак, принцип якого зводиться до виявлення неправильного використання. Тобто, спочатку визначається ненормальна поведінка системи, а вся інша поведінка визначається як нормальна.

Фундаментальна істина (англ. Ground truth) — в алгоритмах машинного навчання це інформація, яка є правдивою, та отримана шляхом прямого спостереження та вимірювання та емпіричних досліджень.

Ботнет (англ. Bot network) — це мережа зламаних комп'ютерів і пристроїв, заражених зловмисним програмним забезпеченням-ботом, які керуються зловмисниками. Мережа ботів використовується для різних атак: DDoS, розсилки спаму, або захоплення управління комп'ютером. Бот-мережі також можуть керуватися без командуючих серверів, використовуючи інші канали керування, щоб передавати команди від одного бота до іншого.

Командно-контрольний сервер (англ. C&C, Command and control) — це централізований сервер або комп'ютер, який використовують зловмисники для видачі команд для керування зловмисним ПЗ і ботами, а також для отримання даних від них.

Python — це об'єктно-орієнтована, інтерпретована мова програмування високого рівня із динамічною типізацією.

Набір даних, датасет (англ. Dataset) — це інформація, що має певну структуру, щоб її можна було застосувати для навчання моделі.

Zbot (Zeus) — ботнет номер один, відомий як банківський троян, заразив понад 3,6 мільйона ПК лише в США, становлячи серйозну загрозу для фінансових установ.

Yakes Trojan — до цього сімейства троянів належать програми, які в залежності від вбудованих інструкцій можуть зробити ПК користувача частиною ботнета або вкрасти чи заблокувати персональну інформацію з комп'ютера користувача.

Машинне навчання (англ. Machine Learning) — галузь обчислювальної науки, яка займається аналізом та інтерпретацією структур даних, щоб керуючись ними приймати рекомендації та рішення.

Scikit-learn — бібліотека для машинного навчання в Python з великим функціоналом.

XGBoost — бібліотека для машинного навчання в Python. Вона реалізує алгоритм посилення градієнта під інфраструктурою Gradient Boosting.

Zeek (aka Bro IDS) — це система IDS, заснована на ідентифікації аномалій.

Wireshark — це відомий та потужний інструмент для захоплення та аналізу мережного трафіку.

Wireshark — це відомий та потужний інструмент для захоплення та аналізу мережного трафіку.

Хост — це пристрій, підключений до комп'ютерної мережі, та має унікальне визначення серед сервісів TCP/IP (IP-адреса).

Матриця плутанини — це матриця, яка надає інформацію наскільки точний алгоритм класифікації класифікує набір даних.

Random forest — це алгоритм машинного навчання, який робить прогнози шляхом комбінування результатів з багатьох окремих дерев рішень — тому їх можна назвати лісом дерев рішень.

Файл .pcap — це дамп мережевих даних, вихідний файл аналізатора Wireshark.

Препроцесор — застосунок, який виконує попередню обробку даних, для того, щоб вони могли використовуватись моделлю.

Пакет даних — в телекомунікаціях це відформатований згідно до протоколу блок даних, який передається за допомогою комутації пакетів в комп'ютерній мережі.

РОЗДІЛ 1 ДОСЛІДЖЕННЯ ПРОБЛЕМИ ВИЯВЛЕННЯ ТА ОБРОБКИ МЕРЕЖЕВИХ ЗАГРОЗ

1.1 Огляд проблеми виявлення та обробки мережевих загроз

Головною проблемою для систем виявлення вторгнень на основі мережі є протокол захисту транспортного рівня TLS (transport layer security), який широко використовується для захисту даних програм в мережі Інтернет. TLS використовує асиметричне шифрування для автентифікації, симетричне шифрування для конфіденційності та коди автентичності повідомлення для збереження цілісності повідомлення. Шкідливі програми використовують протокол TLS, наприклад, під час звернення до command-and-control (C&C) сервера, чим створюють проблему аналізу зашифрованого мережевого трафіку, а методи які засновані на співставленні шаблонів поведінки стають неефективними. У своєму дослідженні [1] Anderson B., McGrew D. стверджують, що використання TLS зловмисним програмним забезпеченням відрізняється від легітимного використання. Дослідники дійшли такого висновку в ході аналізу процесу побудови SSL/TLS з'єднання в деталях, виділяючи функції потоку з полів даних, які відкриті у пакетах даних та не потребують розшифрування. Ці відмінності були кількісно визначені в бінарні вектори, які були використані для навчання моделі логістичної регресії. В наступній роботі [2] Anderson B., McGrew D., Paul S., розглянули велику колекцію функцій потоку, які включають метадані рукостискання TLS, контекстні потоки DNS, й інші доступні параметри мережевого потоку. У висновку вони помітили проблему неточності в ground truth (перевірці на фундаментальних даних). Маючи результати декількох досліджень та висновки щодо ефективності такого підходу, в якому комп’ютерний трафік не розшифровується перед аналізом, у роботі [3] Anderson B., McGrew

D., розглянули та порівняли класифікатори машинного навчання та дійшли висновку, що алгоритм Random Forest є найнадійніший класифікатор для системи виявлення вторгнень.

Алгоритм Random Forest — це ансамблевий алгоритм машинного навчання, який робить прогнози шляхом комбінування результатів з багатьох окремих дерев рішень, тому їх можна назвати лісом дерев рішень. На цих засадах у 2019 році в роботі [4], акумулюючи результати вищезгаданих досліджень, було реалізовано алгоритм MalDetect, який надає можливість ще до закінчення процесу з'єднання TLS віднайти ознаки шкідливого ПЗ. За основу алгоритму вибрана подібна до класичної моделі випадкового лісу, модель Online Random Forest. Перевагами цієї моделі є можливість навчати класифікатор в онлайн-режимі та уникати повторного навчання та повторного розгортання, коли надходять нові зразки.

1.2 Технології для виявлення вторгнень

Існує декілька способів класифікації систем виявлення атак, кожен з яких має різний підхід до проблеми. Тип системи можна визначити, виходячи з таких характеристик:

- Спосіб контролю за системою: network-based, host-based та application based.
- Спосіб аналізу: виявлення зловживань (misuse detection) та виявлення аномалій (anomaly detection).
- Затримка в часі між між етапом аналізу та прийняттям рішення: залежно від затримки, системи виявлення атак діляться на interval-based (або пакетний режим) і real-time.

Звичайно, що найбільшої ефективності досягають у real-time режимі та network-based системи. За способом аналізу зазвичай системи поєднують оби-

два підходи для того щоб усунути недоліки, які властиві кожній системі окремо. Перевагою систем що базуються на аномальних подіях у мережі — є виявлення невідомих раніше видів атак. Абстрактний зловмисник не може стовідсотково знати що його добре запакований вірус залишиться непоміченим системою, яка для того і розроблена, щоб помічати відхилення від норми. Після реєстрації аномалії, в роботу входить модуль аналізу, який спираючись на вагу тих чи інших факторів дає оцінку, наскільки значущою є аномалія. Після цього, приймається рішення щодо подальших дій системи.

Більш застарілою вважається технологія виявлення атак на основі сигнатур. При такому підході, спочатку визначається ненормальна поведінка системи, а вся інша поведінка визначається як нормальна. Тобто така система здатна виявити всі відомі атаки, які є в базі, але не розрахована для виявлення нових, ще невідомих атак.

Переваги сигнатурної системи:

- Детектори зловживань ефективно визначають атаки і мають малий відсоток помилкових спрацювань;
- Детектори зловживань швидко й надійно діагностують використання конкретного інструментального засобу або технології атаки. Це дає змогу адміністратору скоригувати заходи для забезпечення безпеки;
- Швидкість аналізу. Якщо розробити алгоритм на низькорівневій мові програмування, швидкість виявлення буде великою.

Недоліки сигнатурної системи:

- Оскілки детектори зловживань виявляють лише ті загрози, що відомі їм, слід постійно оновлювати їхні бази даних для отримання сигнатур нових атак;
- Основна проблема методу полягає в тому, щоб визначити критерії нормальної активності. Необхідно також встановити допустиме відхилення від нормального трафіку, які ще не вважатимуться атакою;
- Хибне виявлення дії як атаки;

- Пропуск атаки, яка не підпадає під сигнатури атак. Цей випадок більш небезпечний, ніж помилкове віднесення дозволеної дії до класу атак.

Прикладами аномальної поведінки є велика кількість з'єднань за малий проміжок часу, високі завантаження системи невідомим застосунком. Якщо достатньо описати профіль нормальної поведінки суб'єкта, то будь-яке відхилення від нього можна охарактеризувати як аномальну поведінку.

Переваги системи, в основі якої є виявлення аномалій:

- Системи, що виявляють аномалії, фіксуючи несподівану поведінку системи, отримують можливість визначати симптоми атак, не маючи відомостей про їхні конкретні деталі;
- З часом, добре навчена система має базу знань, якою в подальшому можуть скористатися детектори зловживань для визначення сигнатур.

Недоліки:

- Якщо система замало навчена, тобто база знань була малою, утворюється велика кількість помилкових сигналів про загрозу в разі непередбачуваної поведінки користувачів та мережевої активності;
- Цей метод часто потребує певного етапу навчання системи, під час якого визначаються характеристики нормальної поведінки. Треба синтезувати параметри, від ваги котрих буде залежати чи відносити дію до аномальної чи ні. Якість проведення навчання суттєво впливає на подальшу ефективність системи;
- Низька швидкість роботи. Але це залежить від реалізації та системних параметрів машини;
- Завдання побудови профілів суб'єктів, відхилення від яких характеризується як аномальна поведінка.

1.3 Сфери застосування систем виявлення мережевих загроз

Для початку необхідно визначити периметр захисту, за яким буде слідкувати система. Існують два основні типи систем виявлення вторгнення: мережеві (Network-Based Intrusion Detection System, NIDS) і вузлові (Host-Based Intrusion Detection System, HIDS). На рисунку 1 зображено порівняння HIDS та NIDS.

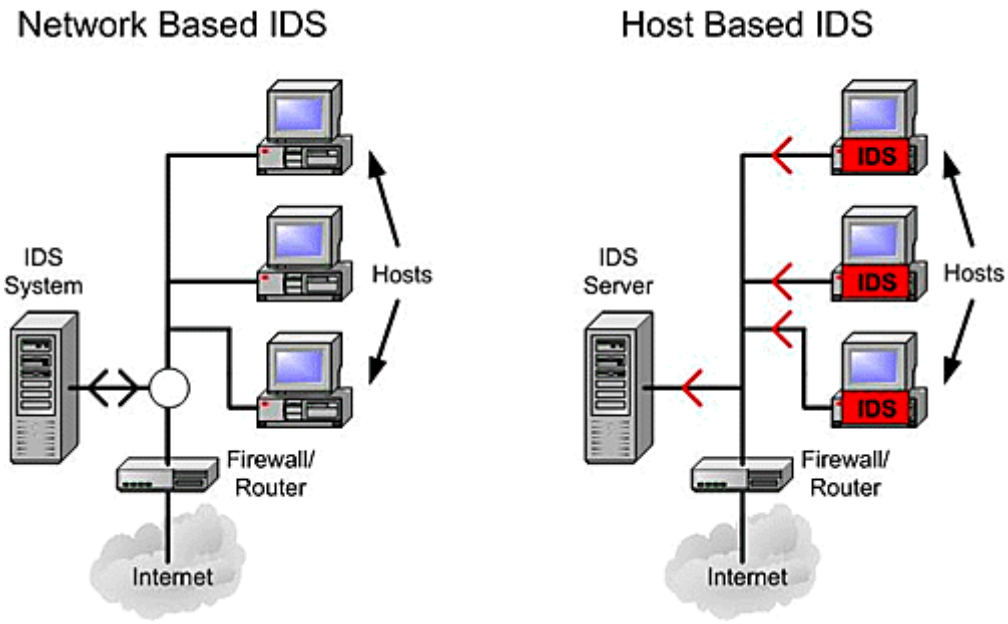


Рис *1. Порівняння HIDS та NIDS*

Система NIDS знаходиться на окремій системі, що відстежує мережевий трафік на наявність ознак атак, що проводяться в підконтрольному сегменті мережі. Як правило, NIDS встановлюється на виділеному обладнанні. Система HIDS розташовується на хості (комп'ютері) користувача і відстежує ознаки атак на цей вузол. Слід зазначити, що обидві системи займаються виявленням (detecting), але саме запобіганням займаються тип рішень, іменований системою запобігання вторгнень (Intrusion Prevention System). Тож слід розуміти, що система обробки загроз отримує сигнал від системи виявлення вторгнень, таким чином вони працюють в парі, створюючи міцний захист в глибині. Самі по собі вони мають обмеження, і ми повинні доповнити кожне з них додатковими рівнями безпеки. В корпоративних цілях слід створити таке рішення, яке буде частиною мережевого комплексу захисту, в такому випадку перевірка мережевого трафіку відбувається до того, як ці пакети досягнуть хоста. Далі, коли пакети вже досягли хоста, ми можемо і перевіряти дані, які можуть бути

зашифровані під час їх проходження по мережі, що ускладнює перевірку трафіку для мережевих рішень. Рішення на базі вузла фіксує трафік і попереджає у разі будь-якої нелегітимної діяльності. Перевага цього методу в тому, що він звужується до одного каналу, що збільшує його можливості. Таким чином, маючи на меті створити програмне забезпечення, яке може виявити атаку скануючи трафік, й відразу його заблокувати, ми маємо на увазі систему запобігання вторгненням на основі на HIDPS (Network-Host Intrusion Detection Prevention System). Це проактивна модель безпеки, яка запобігає зловмисній діяльності в програмному забезпеченні та мережевих системах хоста.

1.4 Існуючі рішення для систем виявлення загроз на захопленому інтернет-трафіку

Найдавнішим методом для захоплення й перевірки мережевих пакетів є метод MITM (Man-In-The-Middle), в якому спочатку розшифровується трафік TLS, потім перевіряє на наявність загроз, та повторно шифрує та надсилає його далі якщо він не несе загрози. Цей метод має багато недоліків. Наприклад, TLS параметри, які використовуються в кінцевих вузлах, повинні зберігатися в безпеці, а розшифрування порушує конфіденційність законного трафіку. Крім того, процеси дешифрування та повторного шифрування забирає багато часу, що негативно впливає на роботу системи загалом. В роботах [1-4] дослідники використовують більш ефективний спосіб синтезу необхідних параметрів протоколу TLS без розшифрування трафіку. Відслідкувавши стандартний алгоритм встановлення підключення TLS, можна побачити, як кілька відформатованих потокових пакетів буде передано між двома хостами, наприклад пакет привітання клієнта (Client Hello) та пакет привітання сервера (Server Hello). Ці пакети не зашифровані, та містять необхідні параметри TLS-з’єднання, які будуть перехоплені та накопичені для створення датасета, який відіграє роль вхідних даних для моделі машинного навчання.

Існує декілька інструментів, що дозволяють захоплювати та аналізувати пакети, що надходять на мережеву карту комп'ютера, вони будуть розглядатися в розділі 2.2. На рисунку 2 зображено процес побудови TLS-з’єднання між клієнтом і сервером. Нас цікавить захоплення від часу початку підключення (першого пакета ACK) та налагодження з’єднання, до моменту перед передачі даних програмою (першого пакета Application Data).

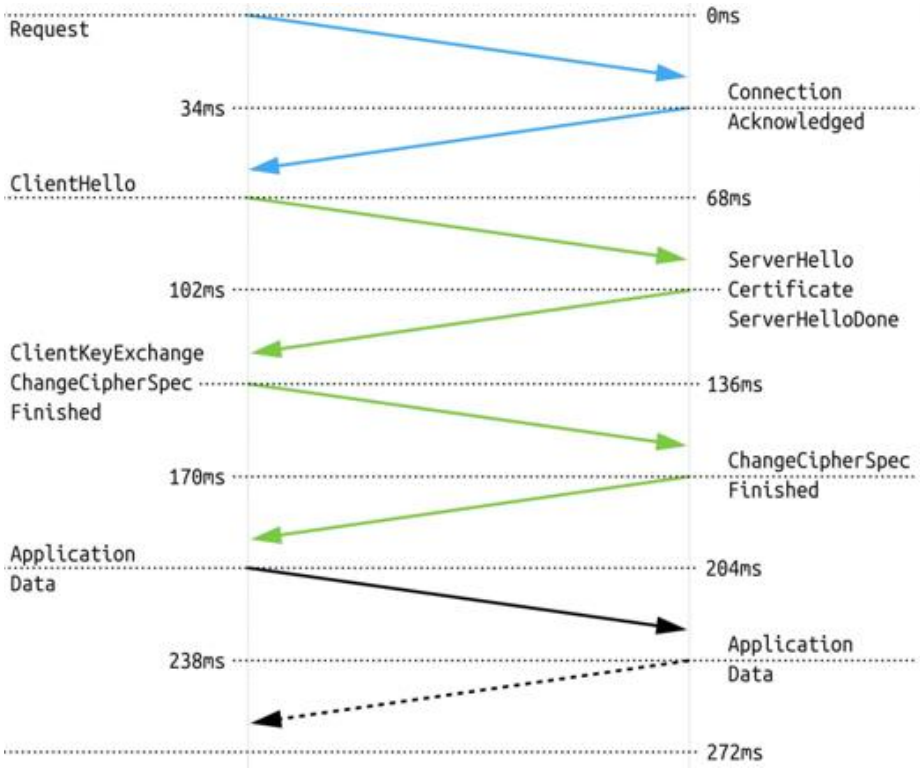


Рис. *2 Процес встановлення TLS-з’єднання (v1.2)*

1.5 Сучасні підходи виявлення загроз у зашифрованому трафіку

Для виявлення загроз у зашифрованому трафіку треба вирішити три проблеми: по-перше, треба вирішити питання захоплення та аналізу трафіку на необхідні змінні без його розшифрування; по-друге, потрібно обробити отримані дані для створення датасету; по-третє, на основі вхідного датасету, створити та навчити класифікатор.

Для вирішення першої проблеми найефективнішим буде підхід, що аналізує мережеві пакети без їх розшифрування, шляхом накопичення й аналізу

параметрів мережевого потоку. Для вирішення другого питання, слід обробити та зберегти накопичені мережеві пакети у вигляді датасету, прийнятного для навчання моделі машинного навчання. Для вирішення третього питання, функції, отримані з наведених мережевих пакетів, потрібно класифікувати. Класифікація зашифрованого трафіку здійснюється за допомогою накопичення вихідних і вхідних пакетів і характеристик часу проходження в мережі, як це розглядається в дослідженні [5], чи накопичення незашифрованих параметрів з'єднання TLS, як це розглядається в дослідженні [2].

1.5.1 Задача виявлення загроз у зашифрованому трафіку

В дослідженні [1] були проаналізовані TLS-потоки із 18 різних сімей шкідливих програм та помічені відмінності як в пакеті ClientHello зараженого клієнта, так і в пакетах ServerHello та сертифікатів шкідливого сервера, з яким з'єднується клієнт.

Розглянемо деякі з параметрів потоку, які клієнт запропонував серверу, відповідно відправивши пакет ClientHello:

1. Відмінності у запропонованих наборах шифрів, які використовуються при з'єднанні. Зловмисним ПЗ використовувались набори шифрів які слабкі або застарілі.
2. Зловмисне ПЗ зазвичай не пропонує більше одного розширення TLS, які додають функціональні можливості протоколу безпеки.
3. Довжина відкритого ключа клієнта різниться між шкідливими ПЗ та звичайними клієнтами.
4. Параметри юзер-агенту не співпадають з тими, які зазвичай використовується легітимним ПЗ при відповідних параметрах пакету.

На рисунку 3 порівнюється частота використання вищезгаданих параметрів потоку, які були запропоновані клієнтом під час налагодження TLS-з'єднання зловмисним та легітимним ПЗ.

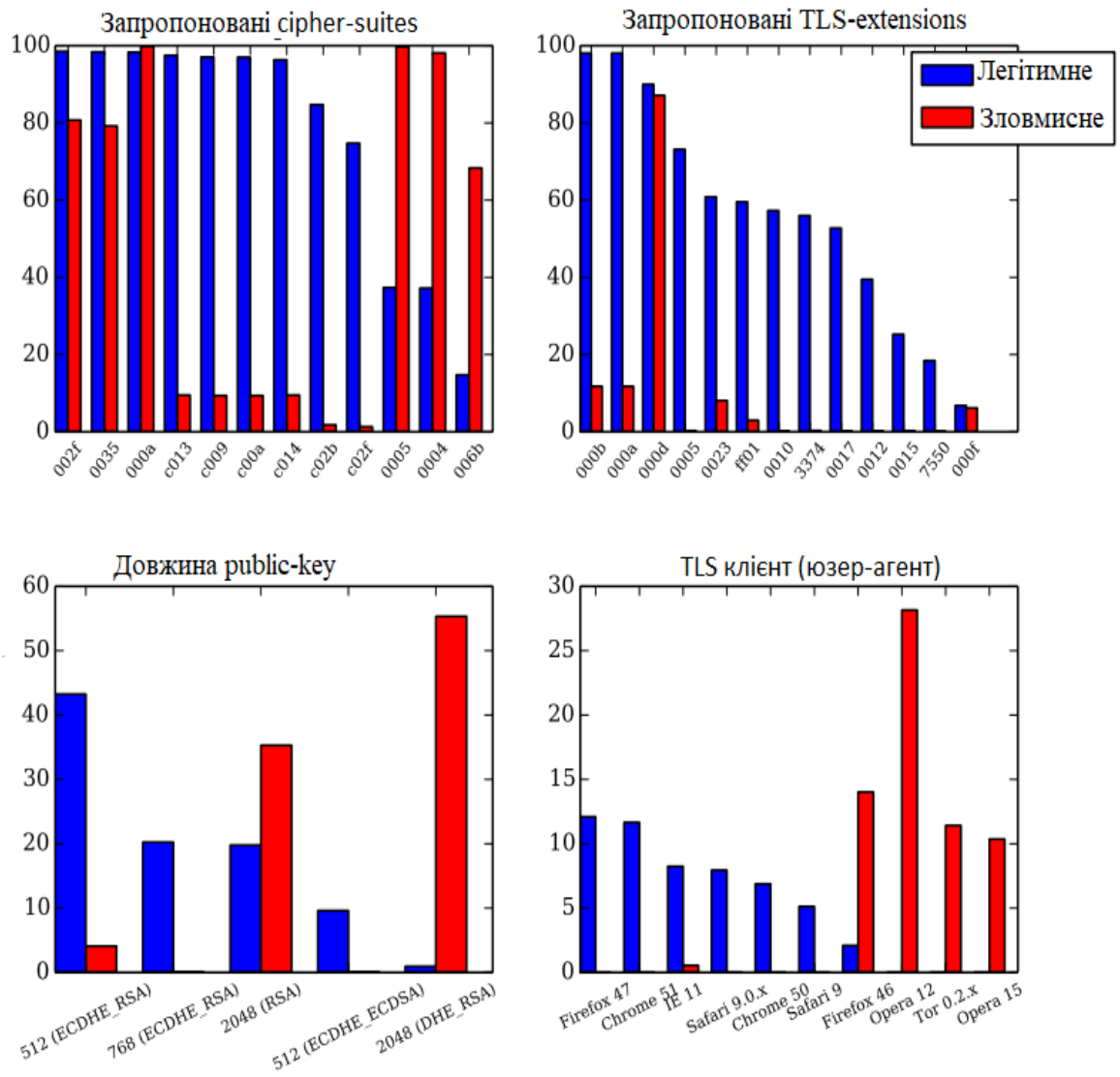


Рис. *3 Порівняння параметрів з'єднання TLS зловмисним та легітимним ПЗ під час кроку Client Hello [2]*

В пакетах ServerHello та сертифікатах шкідливого сервера, зазвичай, теж спостерігаються застарілі та нестандартні рішення при побудові TLS-з'єднання, розглянемо деякі з них:

1. Шкідливі сервери вибирають не ті набори шифрів, які пропонуються в першу чергу, чи які слабкі та застарілі.
2. Сервери, з якими з'єднується шкідливе ПЗ, вибирають не ті розширення, які очікувалися клієнтом.
3. Час між надходженнями пакетів, довжина пакетів, їх частота та деякі інші параметри пакетів є аномальними з доброякісними сеансами.

4. Шкідливі сервери використовують самопідписний SSL-сертифікат частіше, аніж легітимні.

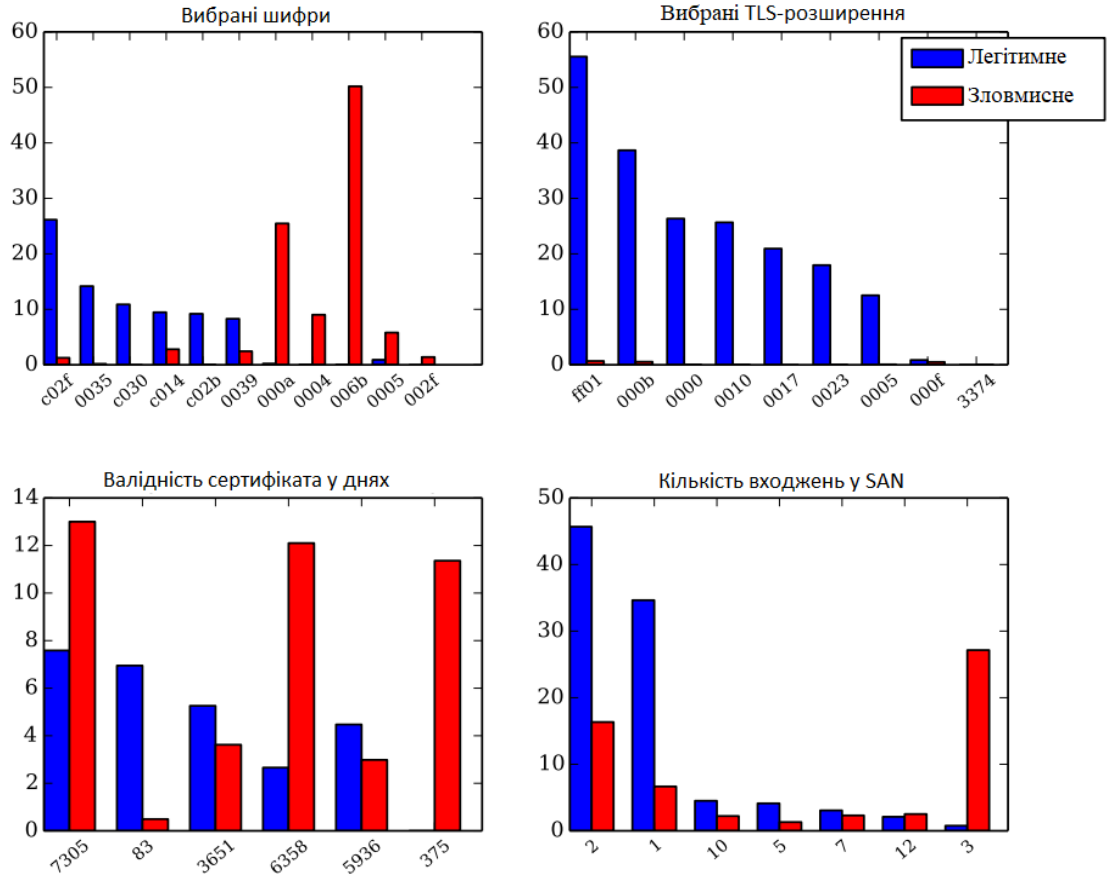
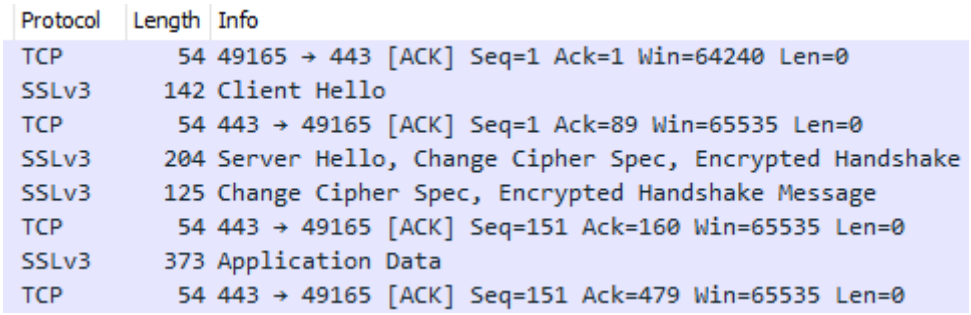


Рис. *4 Порівняння використання параметрів з'єднання TLS зловмисним та легітимним ПЗ під час кроку Server Hello [2]*

На рисунку 4 порівнюється використання параметрів потоку зловмисним та легітимним ПЗ під час кроку ServerHello.

Ознайомившись з висновками дослідження [2], було перевірено їх достовірність шляхом аналізу захопленого трафіка відомого віруса Zbot. Користуючись датасетом MCFP[10], що розглядається в розділі 2.2.7, було завантажено .pcap файл (дамп потоків мережевих даних) під назвою `2014-02-07_capture-win3.pcap`, та відкрито його за допомогою аналізатора WireShark. За допомогою фільтра `(ip.addr == 10.0.2.103 and tls or tcp)` відфільтровуємо пакети за IP-адресою інфікованого хоста та протоколами TLS і TCP.

На рисунку 5 можна спостерігати кроки протоколу TLS, які були виконані зловмисним ПЗ Zbot на хості: від першого пакета ACK до останнього Change Chiper Spec, що завершує процес налагодження з'єднання. Також можна помітити використання SSLv3, означає Secure Sockets Layer. SSLv3 — це 3 версії цього протоколу. Після SSLv3, його було перейменовано на TLS. TLS з v1.0, який є оновленою версією SSLv3. Тож можна констатувати про використання застарілого протоколу, який наразі не є безпечним мережевим протоколом.



Protocol	Length	Info
TCP	54	49165 → 443 [ACK] Seq=1 Ack=1 Win=64240 Len=0
SSLv3	142	Client Hello
TCP	54	443 → 49165 [ACK] Seq=1 Ack=89 Win=65535 Len=0
SSLv3	204	Server Hello, Change Cipher Spec, Encrypted Handshake
SSLv3	125	Change Cipher Spec, Encrypted Handshake Message
TCP	54	443 → 49165 [ACK] Seq=151 Ack=160 Win=65535 Len=0
SSLv3	373	Application Data
TCP	54	443 → 49165 [ACK] Seq=151 Ack=479 Win=65535 Len=0

Рис. *5 Сеанс налагодження з'єднання зловмисним ПЗ Zbot*

Перейшовши до деталей пакету Client Hello, на рисунку 6 можна помітити, що клієнтом було запропоновано шифр `TLS_RSA_WITH_AES_256_CBC_SHA` під кодом `0xc0005`. Також клієнтом було запропоновано декілька розширень, як от `nrenegotiation_info`, `server_name`, `supported_groups`, `ec_point_fromats`.

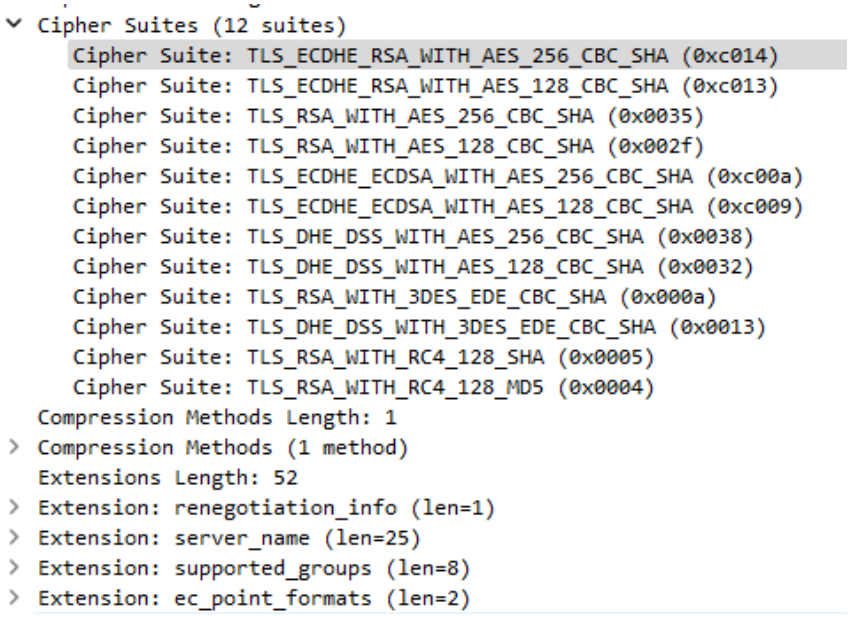


Рис. *6 Параметри пакета ClientHello ПЗ Zbot*

Проаналізувавши деталі пакету Server Hello, можна поміти використання запропонованих застарілих версій протоколу SSLv3, який визнаний застарілим вже в 2015 році (RFC 7568), та шифрування `TLS_RSA_WITH_RC4_128_SHA` під кодом `0xc0005`.

Перейдемо до більш свіжиших прикладів. Звернувшись до датасету MCFP [10] було обрано папку з лог-файлами вірусу Trojan.Yakes. Папка містить .pcap файл (дамп потоків мережевих даних) під назвою `2016-11-24_capture-win8.pcap`, та інші додаткові файли від Zeek IDS. Відкриваємо файл за допомогою аналізатора WireShark. За допомогою фільтра `(ip.addr == 192.168.1.118 and tls or tcp)` відфільтровуємо пакети за IP-адресою інфікованого хоста та протоколами TLS і TCP.

На рисунку 7 можна спостерігати кроки протоколу TLS v1.0, від якого відмовилися як від стандарту в 2020 році.

Protocol	Length	Info
TCP	54	49160 → 443 [ACK] Seq=1 Ack=1 Win=65536 Len=0
TLSv1	168	Client Hello
TCP	54	443 → 49160 [ACK] Seq=1 Ack=115 Win=29312 Len=0
TLSv1	1514	Server Hello
TLSv1	745	Certificate, Server Key Exchange, Server Hello Done
TCP	54	49160 → 443 [ACK] Seq=115 Ack=2152 Win=65536 Len=0
TLSv1	188	Client Key Exchange, Change Cipher Spec, Encrypted Handshake
TCP	54	443 → 49160 [ACK] Seq=2152 Ack=249 Win=30336 Len=0
TLSv1	113	Change Cipher Spec, Encrypted Handshake Message
TCP	54	49160 → 443 [ACK] Seq=249 Ack=2211 Win=65536 Len=0
TLSv1	379	Application Data

Рис. *7 Сеанс налагодження з'єднання зловмисним ПЗ Yakes*

Перейшовши до деталей пакету Client Hello, можна помітити, що клієнтом було запропоновано шифр `TLS_RSA_WITH_AES_128_CBC_SHA` під кодом `0xc002f` та розширення, що спостерігались у розглянутих вище дослідженнях. На рисунку 8 зображений пакет Client Hello.

```
▾ Cipher Suites (12 suites)
    Cipher Suite: TLS_RSA_WITH_AES_128_CBC_SHA (0x002f)
    Cipher Suite: TLS_RSA_WITH_AES_256_CBC_SHA (0x0035)
    Cipher Suite: TLS_RSA_WITH_RC4_128_SHA (0x0005)
    Cipher Suite: TLS_RSA_WITH_3DES_EDE_CBC_SHA (0x000a)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA (0xc013)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA (0xc009)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_128_CBC_SHA (0x0032)
    Cipher Suite: TLS_DHE_DSS_WITH_AES_256_CBC_SHA (0x0038)
    Cipher Suite: TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (0x0013)
    Cipher Suite: TLS_RSA_WITH_RC4_128_MD5 (0x0004)
  Compression Methods Length: 1
▸ Compression Methods (1 method)
  Extensions Length: 40
▸ Extension: server_name (len=20)
▸ Extension: supported_groups (len=6)
▸ Extension: ec_point_formats (len=2)
```

Рис. *8 Параметри пакету ClientHello ПЗ Yakes*

Перейшовши до деталей пакету Server Hello як зображено на рисунку 9, сервером було вибрано запропонований клієнтом шифр

TLS_RSA_WITH_AES_128_CBC_SHA, та одне розширення ec_point_formats.

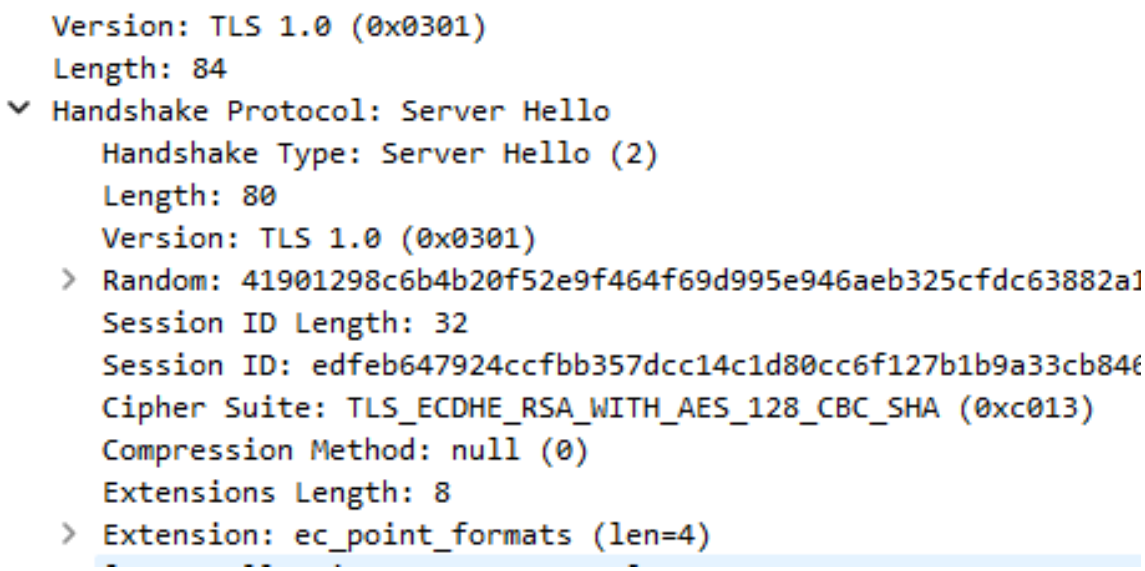


Рис. *9 Параметри пакету ServerHello ПЗ Yakes*

Проаналізувавши результати досліджень та впевневшись в них власноруч, шляхом аналізу трафіку від деяких видів зловмисного ПЗ, було помічено кореляцію запропонованих клієнтом параметрів пакету Client Hello, та параметрів пакету Server Hello, які збігаються із висновками досліджень [2-4].

1.6 Аналіз програмного забезпечення для виявлення та обробки вторгнень на хості

1.6.1 Рішення від компанії Open Information Security Foundation

Suricata — система виявлення, обробки і попередження вторгнень у галузі мережі. Програма розроблена компанією Open Information Security Foundation. Suricata являє собою модульний застосунок, в можна оперативно підключити новий модуль для перегляду та аналізу даних, який своєчасно відреагує на загрозу. Окрім вбудованих алгоритмів аналізу трафіка передбачає

задіяння бази сигнатур, що розвивається проектом Snort, а також наборів правил, як у мережевому екрані. Для збору даних, захоплення, декодування, виявлення вторгнень використовується окремий модуль. Є можливість перевіряти файли, що передаються по HTTP, проводити розпізнавання за URI, cookie, заголовками та іншими. Для перехоплення потоків використовуються потужні двигуни: Libpcap, AF_PACKET, NFQueue, IPFRing, IPFW, PF_RING. На рисунку 8 зображений модуль сповіщення Suricata IDS.

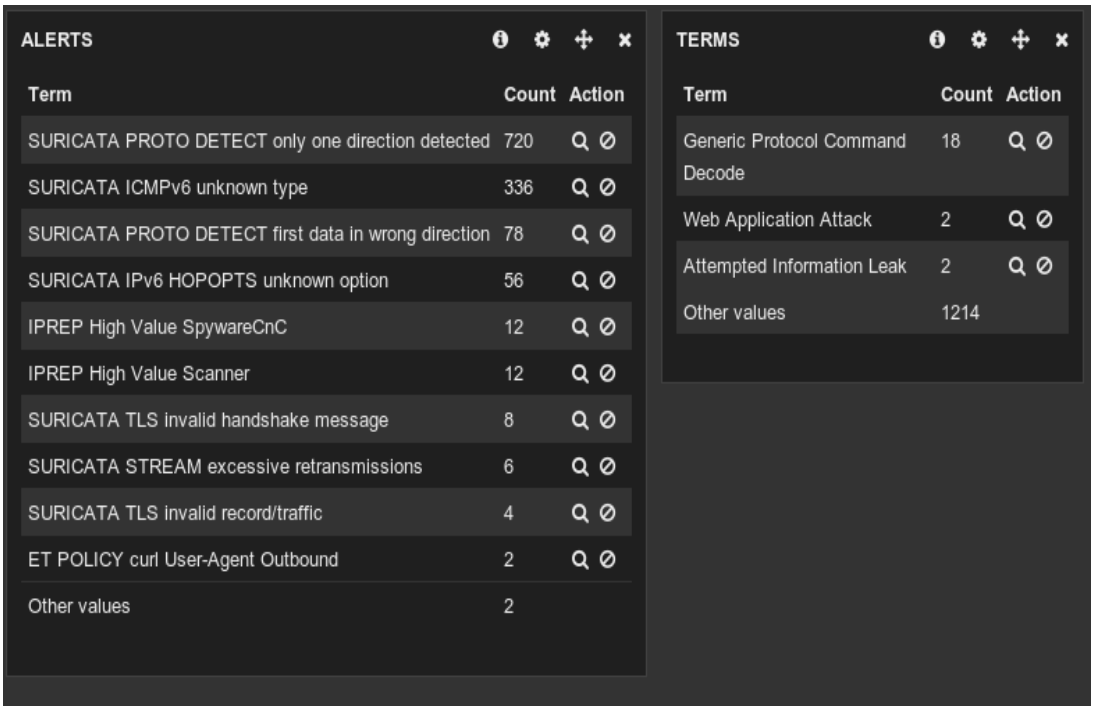


Рис. *10 Suricata IDS модуль сповіщення*

1.6.2 Рішення від компанії Cisco і SourceFire

Snort — мережева система попередження вторгнень (IPS) і система виявлення вторгнень (IDS) з відкритим вихідним кодом, яка здатна на протоколювання, аналіз, пошук за вмістом пакету, а також досить часто використовується для активного блокування або пасивного виявлення цілого ряду атак , наприклад: сканування портів, переповнення буфера й інші види атак. Програмне забезпечення зазвичай використовується для запобігання проникненню,

блокування атак, якщо вони мають місце. На рисунку 9 зображена звітність роботи Snort IDS.

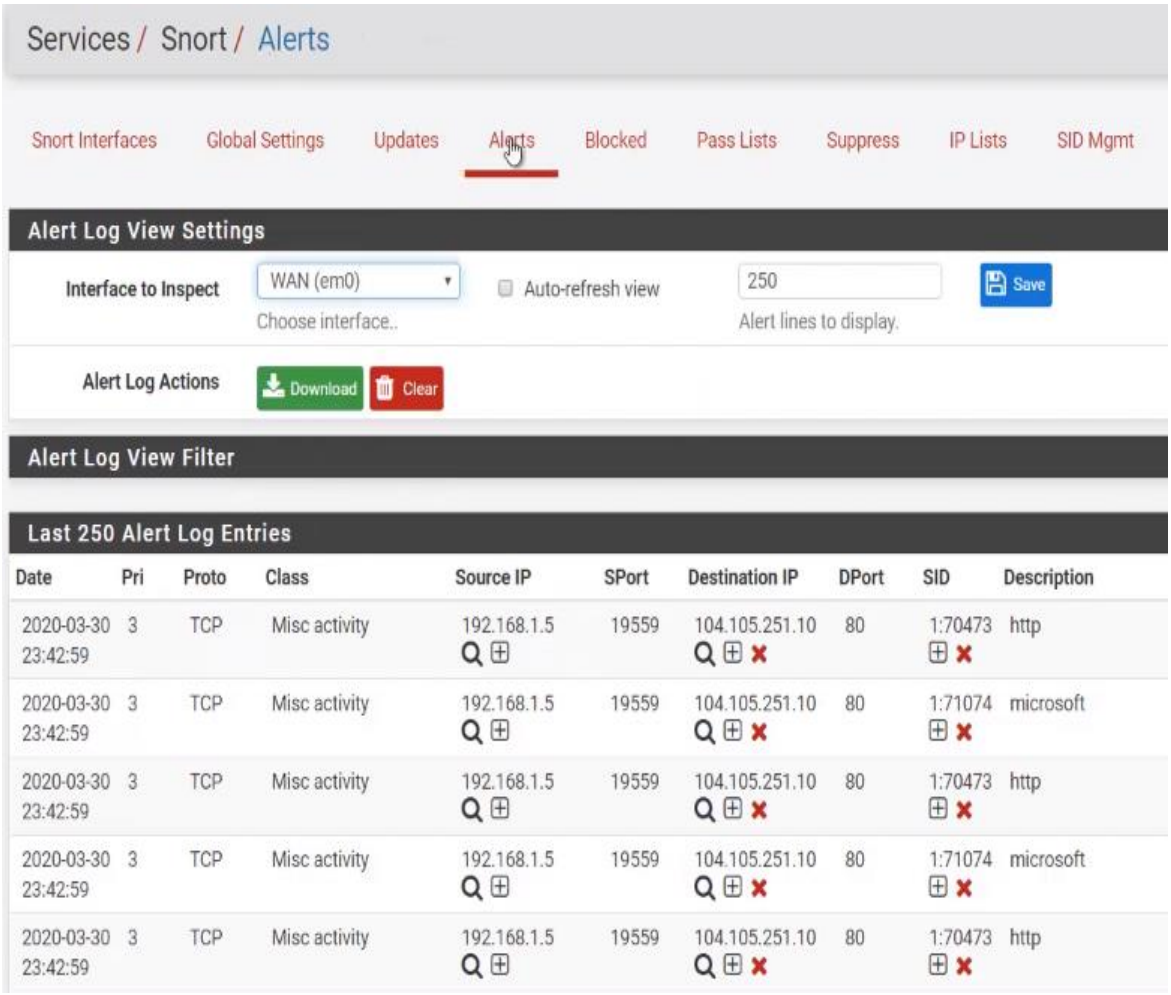


Рис. *11 Snort, встановлений на GUI phsense.*

1.6.3 Рішення Zeek-IDS

Zeek (aka Bro IDS) — це система IDS, заснована на ідентифікації аномалій. Його механізм аналізу перетворює отриманий трафік у серію подій, далі інтерпретатор згідно сценаріїв політики приймає рішення щодо подальших дій. Цей механізм політики має власну мову (Bro-Script), і він може виконувати дуже потужні та різноманітні завдання. В якості вихідних даних застосунок генерую звіти зі статистикою параметрів з'єднання, створюючи додаткові файли conn.log, http.log й інші файли, які є корисними для аналізу.

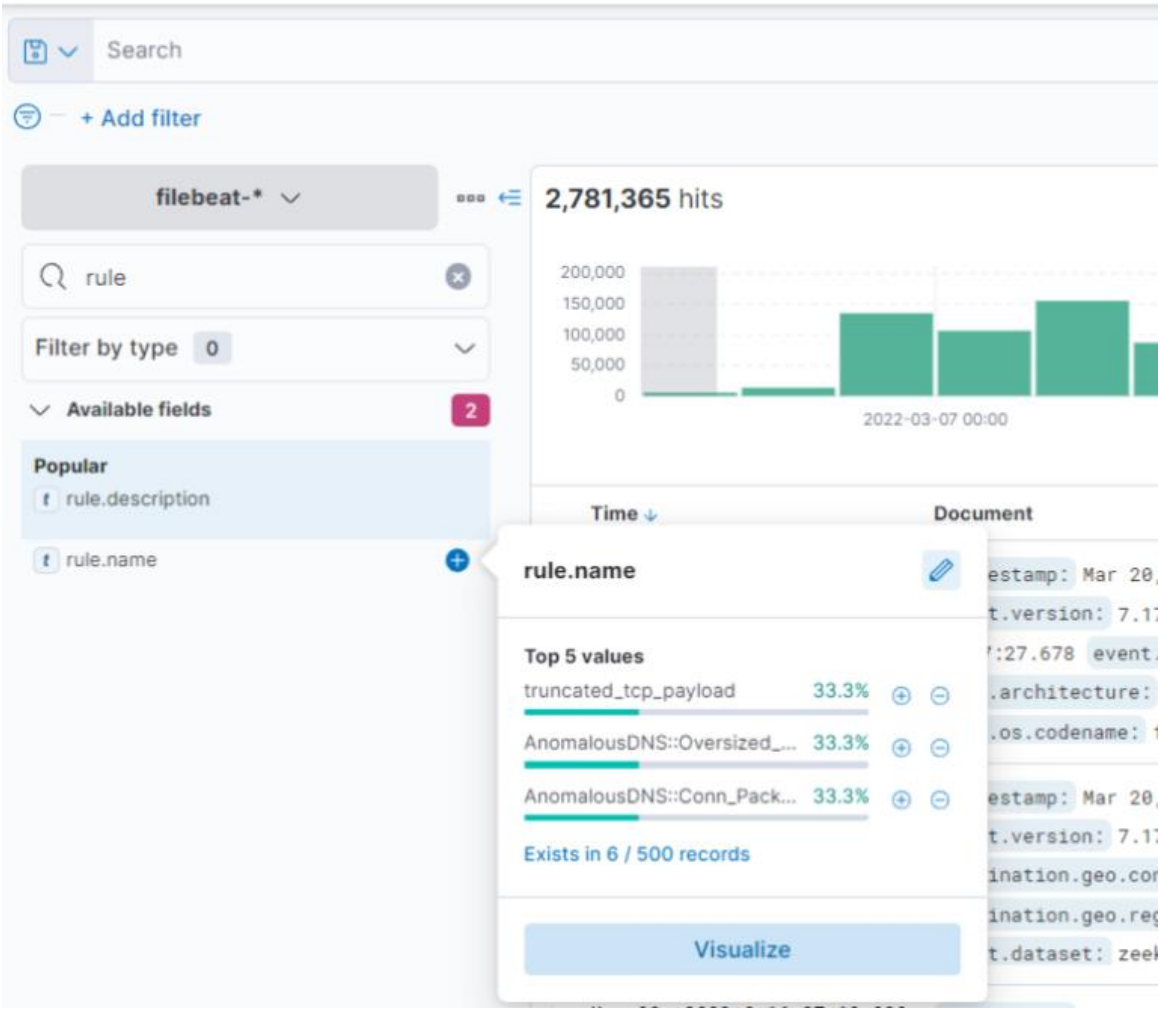


Рис. *12 Zeek IDS rules*

1.7 Висновки з розділу 1

1. Системи для виявлення та обробки вторгнень поділяються на network-based, host-based та application based. Кожний спектр рішень виконує своє завдання, та для хостової системи виявлення вторгнень в мережі, слід створити модель безпеки, яка запобігає зловмисній діяльності ПЗ на хості.
2. Головною проблема для такої системи є шифрування протоколу TLS, що забезпечує захищене з’єднання між клієнтом та сервером.
3. Проаналізувавши висновки дослідників, було обрано необхідні функції трафіку, які будуть вхідними даними для навчання класифікатора.

4. Шляхом аналізу мережевих пакетів на хості, система, що застосовує такий класифікатор, перехоплює та накопичує корисні та незашифровані функції потоку TLS, та за допомогою них проводить навчання класифікатора для подальшого його впровадження у повноцінну NHIDS.
5. Проблема виявлення та обробки вторгнень є актуальною і потребує подальшого розгляду.

РОЗДІЛ 2 ДОСЛІДЖЕННЯ ЗАСОБІВ ВИЯВЛЕННЯ ТА ОБРОБКИ МЕРЕЖЕВИХ ЗАГРОЗ

2.1 Алгоритми машинного навчання

Машинне навчання (англ. Machine Learning) — це галузь обчислювальної науки, яка займається аналізом та інтерпретацією структур даних, щоб керуючись ними приймати рекомендації та рішення. Також, якщо виявлено виправлення, алгоритм може включити цю інформацію для покращення прийняття рішень. Машинне навчання вважається підмножиною штучного інтелекту (AI). Слід виділити навчання з учителем та без учителя. Навчання з учителем передбачає наявність розрізненого набору даних, в якому відомо значення істини. Навчання без учителя передбачає витягування корисної інформації з довільних даних. Машинне навчання має застосування майже в усіх галузях людської діяльності. Схема роботи стандартного алгоритму машинного навчання наступна: спочатку в модель подаються дані параметрів, відповідь на які відома, далі запускається алгоритм і вносяться коригування, доки вихідні дані алгоритму навчання не узгодяться з відомою відповіддю. На цьому етапі вводяться все більші обсяги даних, щоб допомогти системі навчатися та обробляти більш високі обчислювальні рішення.

На основі аналізу найбільш популярних алгоритмів машинного навчання розглянемо список різноманітних алгоритмів для вибору найбільш точної моделі, здатної класифікувати мережевий трафік:

- Машина опорних векторів
- Алгоритм випадкового лісу (random forest)
- K-найближчі сусіди
- Логістична регресія
- Підвищення

Запропоновані алгоритми машинного навчання охоплюють найпопулярніші методи вирішення подібних проблем. Це дозволить порівняти результати роботи алгоритмів між собою, що дасть краще розуміння у виборі ефективної моделі.

2.1.1 Модель Random forest

За результатами дослідження [5], команда Cisco протестувала моделі логістичної регресії та випадкового лісу і дійшла висновку, що алгоритм випадкового лісу виконував найкращий за результатами порівняння різних моделей. Моделі навчалися на позначених наборах даних: один доброякісний, зібраний з потоку на підприємстві, інший складається з шкідливих потоків. Також була проведена 10-кратна перехресна перевірка для оцінки помилки тесту. Точність коли класифікатор допускає лише одну помилку на 10 000 доброякісних потоків також була підрахована, що відображено на таблиці 1.

Алгоритм	Точність виявлення у %	Хибно позитивна похибка 0.01%	Хибно позитивна похибка < 0.01%
Random forest	99.6	-	86.8
Логістична регресія	99.6	87.4	86.4

Таблиця 1. *Точність для різних алгоритмів і хибнопозитивні показники*

Алгоритм випадкових лісів [6] робить прогнози шляхом комбінування результатів з багатьох окремих дерев рішень — тому їх можна назвати лісом дерев рішень. Можна виділити наступні кроки:

1. У випадковому лісі n кількість випадкових записів береться з набору даних, що має k кількість записів.
2. Для кожної вибірки вхідних файлів будуються окремі дерева рішень.
3. Кожне дерево рішень генерує вихідні дані.

4. Остаточний результат розглядається на основі голосування більшості або усереднення для класифікації та регресії відповідно.

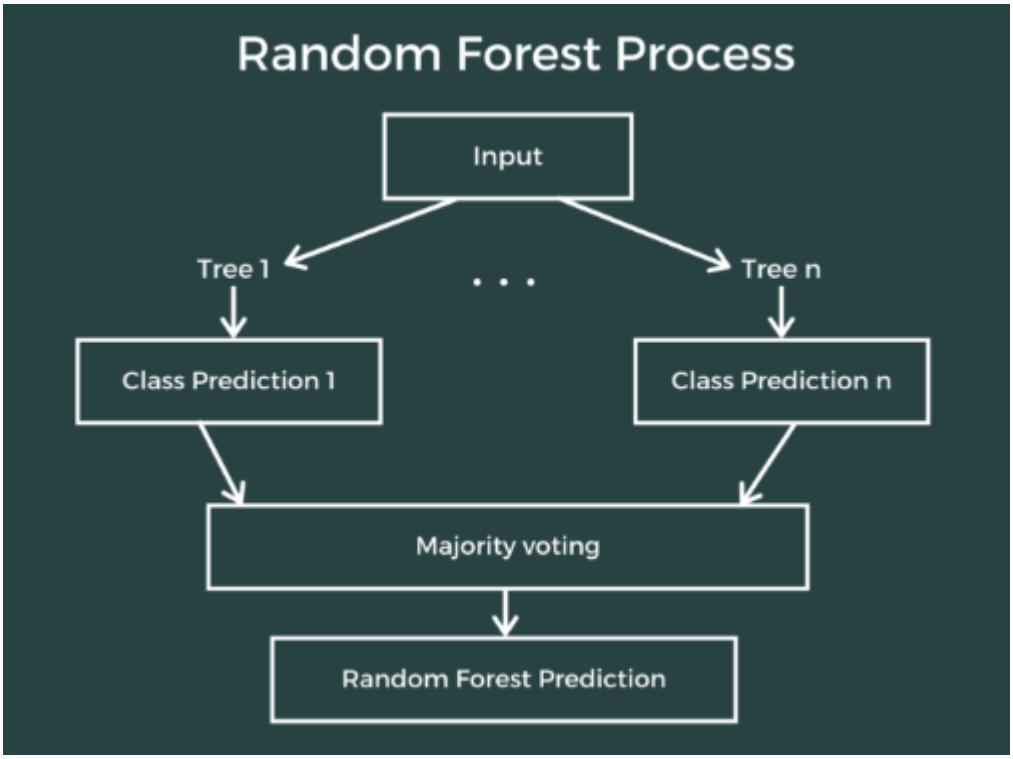


Рис. *13 Алгоритм Random Forest*

2.2 Фреймворки для захоплення й обробки мережевого трафіку

2.2.1 Бібліотека libpcap

Library Packet Capture забезпечує інтерфейс високого рівня для систем захоплення пакетів. Через цей механізм доступні всі пакети в мережі, навіть ті, які призначені для інших хостів. Бібліотека libpcap дозволила розробникам писати код для отримання пакетів канального рівня (рівень 2 у моделі OSI) на різних версіях операційних систем UNIX, не турбуючись про особливості мережевих карт і драйверів різних операційних систем. По суті, бібліотека libpcap захоплює пакети безпосередньо з мережевих карт, що дозволило розробникам писати програми для декодування, відображення або реєстрації пакетів. Програма TCPDump робить саме це. Аналізатор, створений для аналізу

проблем продуктивності TCP, TCPDump дозволяв вам захоплювати пакети а потім декодувати та відображати їх. Одного дня, розчарований обмеженнями та вихідними форматами TCPDump, розробник Марті Реш написав Snort як заміну TCPDump. Формат файлу libpcap є основним форматом файлу захоплення, який використовується в TcpDump / WinDump , Snort та багатьох інших мережевих інструментах. Він повністю підтримується Wireshark/TShark, але тепер вони генерують файли pcapng за замовчуванням.

Файл з розширенням `.pcap` має глобальний заголовок, що містить деяку глобальну інформацію, за яким слідує нуль або більше записів для кожного захопленого пакета, що виглядає так:

- Глобальний заголовок. Цей заголовок запускає файл libpcap, а за ним слідуватиме перший заголовок пакета
- Заголовок пакета. Містить дату і час, коли цей пакет був захоплений, кількість байтів пакетних даних, довжину пакета
- Пакетні дані. Будуть одразу слідувати за заголовком пакета у вигляді блобів даних incl_len байтів без конкретного вирівнювання байтів.
- Заголовок пакета
- Пакетні дані

Цей формат файлу є дуже простим форматом для збереження захоплених мережевих даних. Libpcap і порт Windows libpcap, WinPcap , використовують той самий формат файлу.

2.2.2 Бібліотека PcapPlusPlus

PcapPlusPlus — це крос-платформна бібліотека C++ для захоплення й аналізу мережевих пакетів. Бібліотека має функціонал читання та запису мережевих пакетів у файлах pcap чи pcapng, декодування та обробки великої кількості мережевих протоколів. Бібліотека надає прості у використанні обгортки C++ для найпопулярніших механізмів обробки пакетів, таких як згаданого вище libpcap, WinPcap Npcap, DPDK і PF_RING. За допомогою функціоналу

збірки пакета, PcapPlusPlus містить унікальну реалізацію методів повторного складання пакетів (TCP Reassembly), він підтримує повторну передачу TCP, реагування на пакети TCP, що не відповідають порядку та відсутні дані TCP. Також бібліотека має функціонал фрагментації та дефрагментації IP-пакетів для створення та повторного з'єднання фрагментів IPv4 та IPv6.

2.3 Фреймворки та бібліотеки машинного навчання

2.3.1 Бібліотека Scikit-learn

Бібліотека Scikit-learn — дуже потужна бібліотека для машинного навчання в Python. Вона пропонує широкий вибір алгоритмів для машинного навчання, масштабування даних, статистичного моделювання, включаючи класифікацію, регресію, кластеризацію. Побудована на NumPy, SciPy та Matplotlib. Алгоритми машинного навчання в даній бібліотеці як вхідні дані приймають широкий спектр їх представлення: дата у вигляді таблиці (.csv), дата як масив (чи вектор), дата як матриця. Бібліотека надає потужний API для логування результатів відпрацювання моделей: графіки, аналіз точності, ваги моделі й інші.

2.3.2 Бібліотека XGBoost

Бібліотека XGBoost — бібліотека для машинного навчання в Python, ліцензія Apache License. Вона є сучасним алгоритмом машинного навчання для роботи зі структурованими даними. Реалізує алгоритм посилення градієнта від Chen & Guestrin (2016) під інфраструктурою Gradient Boosting, яка забезпечує паралельне прискорення дерева, тож його можна розділити на графічні процесори і в мережі комп'ютерів, що вказує на його продуктивність на великих наборах даних. Має потужний функціонал: параметри для перехресної перевірки, регуляризації, різний вид вхідних даних, цільових функцій, параметрів

дерева, API що сумісне з scikit-learn, інструменти для логування результатів роботи.

2.4 Датасети для навчання моделі машинного навчання

2.4.1 Датасет CTU-13

Мета набору даних полягала в тому, щоб охопити великий обсяг реального трафіку ботнету, змішаного зі звичайним і фоновим трафіком. Набір даних CTU-13 [9] складається з тринадцяти захоплень (так званих сценаріїв) різних зразків ботнету. Зараз налічується 349 архівів із різними зразками malware. У кожному сценарії автори виконували конкретне шкідливе програмне забезпечення, яке використовувало кілька протоколів і виконувало різні дії. Кожен сценарій був записаний у файл pcap, який містить усі пакети трьох типів трафіку. Ці файли pcap були оброблені для отримання інформації іншого типу, наприклад NetFlows, WebLogs тощо. Перший аналіз набору даних CTU-13, описаний та опублікований у статті «Емпіричне порівняння методів виявлення бот-мереж», використовував односпрямовані NetFlows для представлення трафіку та призначення міток.

2.4.2 Датасет MCFP

Ці набори даних [10] були отримані в Університеті CTU в Чеській Республіці. Папка, в якій зберігається кожен набір даних, містить більше інформації про неї, наприклад файли NetFlow, журнали HTTP та інформацію про DNS. Ці файли регулярно оновлюються, коли витягується нова інформація. Вони вирішують проблему розмежування між клієнтом і сервером, і включають більше інформації, а також вони містять набагато більш детальні мітки. Папка, в якій зберігається кожен набір даних, містить більше інформації про неї, наприклад файли NetFlow, журнали HTTP та інформацію про DNS. Ці файли регулярно оновлюються, коли витягується нова інформація.

Звичайний датасет містить файли .pcap з хоста, на якому вони записані, опис сценарію цього датасету, IP-адресу хоста та сервера. Датасет також може містить та декілька файлів статистичної інформації, наприклад:

- capture20110810.pcap. Це захоплення pcap з усім трафіком (фоновим, звичайним та ботнетом) Цей файл pcap не був оприлюднений, оскільки містить занадто багато приватної інформації про користувачів мережі. Цей файл був захоплений на головному маршрутизаторі мережі університету.
- botnet-capture-20110810-neris.pcap.Захоплення тільки трафіком ботнету. Вона оприлюднюється. Цей файл був захоплений на інтерфейсі зараженої віртуальної машини.
- capture20110810.pcap.netflow.labeled. Цей файл має мережеві потоки породжені односпрямованим argus.
- Bro – папка з файлами для Bro-IDS
- Детальні-двонаправлені-потоки-мітки
- .html. Це графічна сторінка html, зроблена за допомогою CapTipper HTTP-запитів у захопленні.
- Json. Файл, необхідний Captipper для html-сторінки.
- *.binetflow.2format

2.5 Висновки з розділу 2

1. Найбільш оптимальним алгоритмом машинного навчання є Random forest. Він є оптимальним рішенням згідно декількох досліджень [1-4] та дає високий відсоток передбачення загрози в галузі мережевих пакетів.
2. В якості датасета найбільш інформативними й актуальними є датасети MCFP.

3. Навчений класифікатор за допомогою системи правил обробки дій можна використовувати як повноцінну хостову систему виявлення й обробки загроз.
4. Для найбільшої ефективності обраних засобів розробки, слід реалізувати комп’ютерну систему на якомога нижчому мережевому та програмному рівні.

РОЗДІЛ 3 РОЗРОБКА ХОСТОВОЇ СИСТЕМИ ВИЯВЛЕННЯ ТА ОБРОБКИ ВТОРГНЕНЬ З ВИКОРИСТАННЯМ МАШИННОГО НАВЧАННЯ

3.1 Навчання моделі Random Forest

Згенерований алгоритмом Random Forest «ліс» рішень, який складається дерев рішень, зазвичай навчаються методом «мішків». Дерева рішень здатні прогнозувати з певним ступенем точності, але в поєднанні один з одним вони стають значно надійніші в прогнозуванні. Велика кількість дерев у лісі призводить до більш високої точності та запобігає проблемі перенавчання. Ансамблеві алгоритми мають деякі характеристики:

- Методи пакування (Bootstrap Aggregation або Bagging) — це ансамблевий мета-алгоритм. Ідея створення пакетів полягає в об’єднанні прогнозів кількох базових учнів для створення більш точних результатів;
- Гіперпараметри — в моделі випадкового лісу використовуються для збільшення прогностичної здатності моделі, або для її прискорення;

3.1.1 Налаштування середовища для навчання

Першим етапом, необхідним для навчання моделі випадкового лісу, є налаштування робочого середовища.

Навчання було проведено на комп’ютері з наступними характеристиками:

1. Шести-ядерний процесор Intel Core i7-8750.
2. Графічний процесор NVIDIA GeForce GTX 1050.
3. 16 ГБ оперативної пам’яті.
4. Операційна система Windows 11.

Для навчання моделі випадкового лісу [6], треба реалізувати алгоритм в його автентичному вигляді, або скористатись бібліотеками по типу scikit-learn. Розглянемо останній випадок:

1. Встановлено Zeek (aka Bro IDS) в якості препроцессора для трафіку.
2. Встановлено мову програмування Python.
3. Встановлено середовище розробки Visual Code.
4. Встановлені модулі numpy, pandas, math, matplotlib, sklearn
5. Завантажено датасети MCFP. Виконана їх обробка
6. Навчання моделі

3.1.2 Підготовка датасетів зловмисного ПЗ

Датасети представляють собою .pcap файли, що безпосередньо містять перехоплені пакети, та допоміжні файли від Zeek. Файли .pcap мають бути масштабовані у вектор, що містить перелік корисних функцій, що експортуються із потоку.

Датасети від MCFP [10] містять записи звичайного трафіку на хості та записи трафіку, при відкритому зловмисному ПЗ. Таким чином, даний датасет дає змогу методологію «навчання з вчителем», де в ролі вчителя виступає залежна змінна, яка вказує алгоритму, чи зловмисний трафік наданий моделі, чи ні.

Для підготовки датасету MCFP були виконані наступні кроки:

1. Завантажені з офіційного сайту MCFP [10] архіви, які містять файли .pcap, та допоміжні файли зі статистикою.
2. Розроблено і виконано скрипт на мові програмування Python, який експортує таблицю .csv у прийнятному для моделі вигляді.

Кожний рядок у вихідній таблиці (матриці) представляє собою вектор, який можна переглядати, редагувати, та оцінити рішення системи. Він містить усі функції, що можна експортувати з мережевих дампів (файлів .pcap).

На таблиці 2 зображено оброблений датасет, де зібрано 439 функцій потоку, витягнутих з файлів .pcap

Src.Port	Dst_Port	Bytes_in	…	isMalware
1337	443	12222	…	1
43303	443	5432	…	0

Таблиця 2. *Оброблений датасет, зібраний з бази MCFP*

На рисунку 14 можна побачити співвідношення зловмисних та звичайних потоків у датасеті:

Рис. *14 Оброблений датасет MCFP*

Характеристики датасету:

- Звичайне ПЗ : 24999 прикладів
- Зловмисне ПЗ: 16467 прикладів
- Всього: 41466
- Функцій: 439, від Src_Port до isMalware
- Типи даних: dtypes: float64(3), int64(436)
- Використання пам’яті: 138.9 MB

На рисунку 15 можна побачити декілька прикладів з завантаженого датасету:

Характеристики датасету:

	Src_Port	Dst_Port	Bytes_in	...	ec_pts_0	ec_pts_1	isMalware
0	49754	443	6856	...	1	0	1
1	49769	449	1578	...	1	0	1
2	49777	449	1541	...	1	0	1
...	...	...	...	...	...	...	...
41463	43303	443	1813	...	1	0	0
41464	46723	443	1813	...	1	0	0
41465	43075	443	1674	...	1	0	0

[41466 rows x 439 columns]

Рис. *15 Датасет, приклад*

На рисунках 16-21 зображені декілька параметрів обробленого датасету.

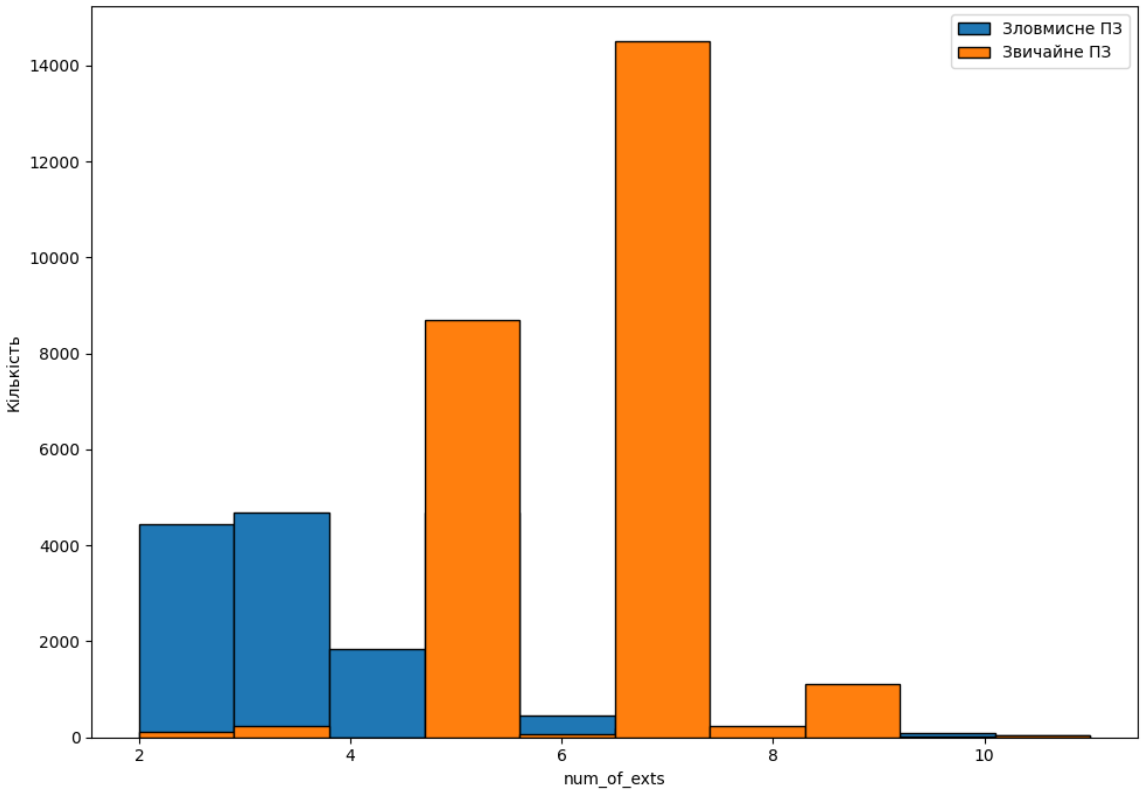


Рис. *16 Статистика параметру: кількості розширень*

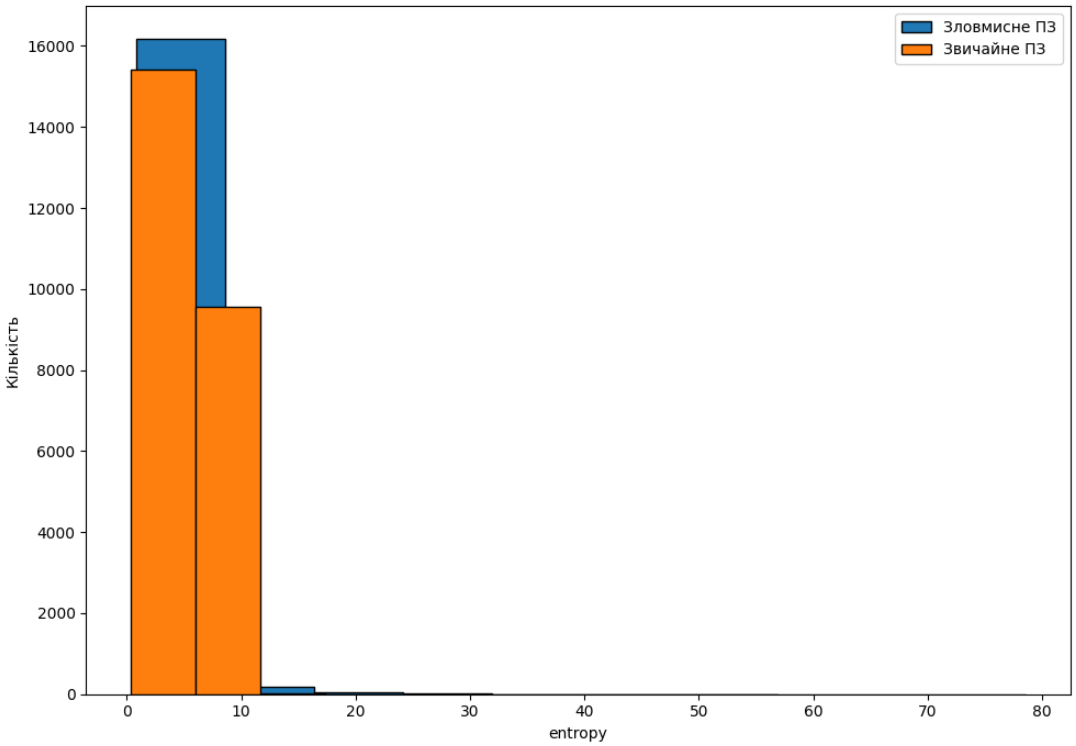


Рис. *17 Статистика параметру: ентропія сеансу*

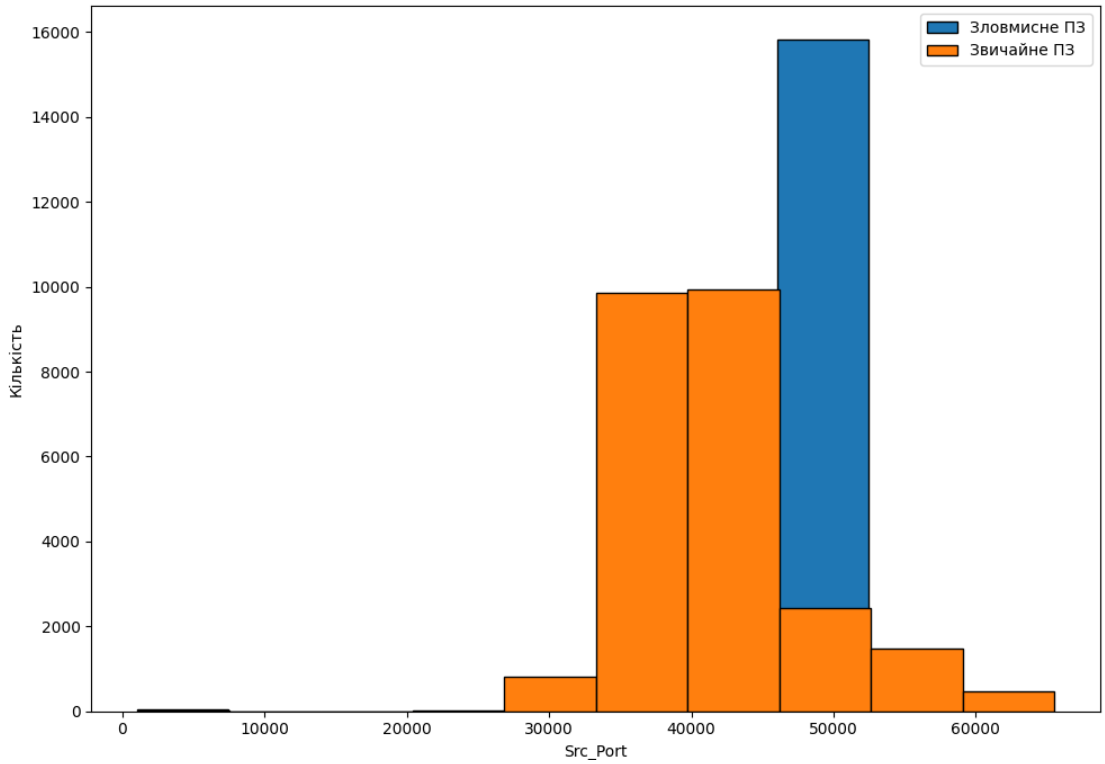


Рис. *18 Статистика параметру: вхідний порт*

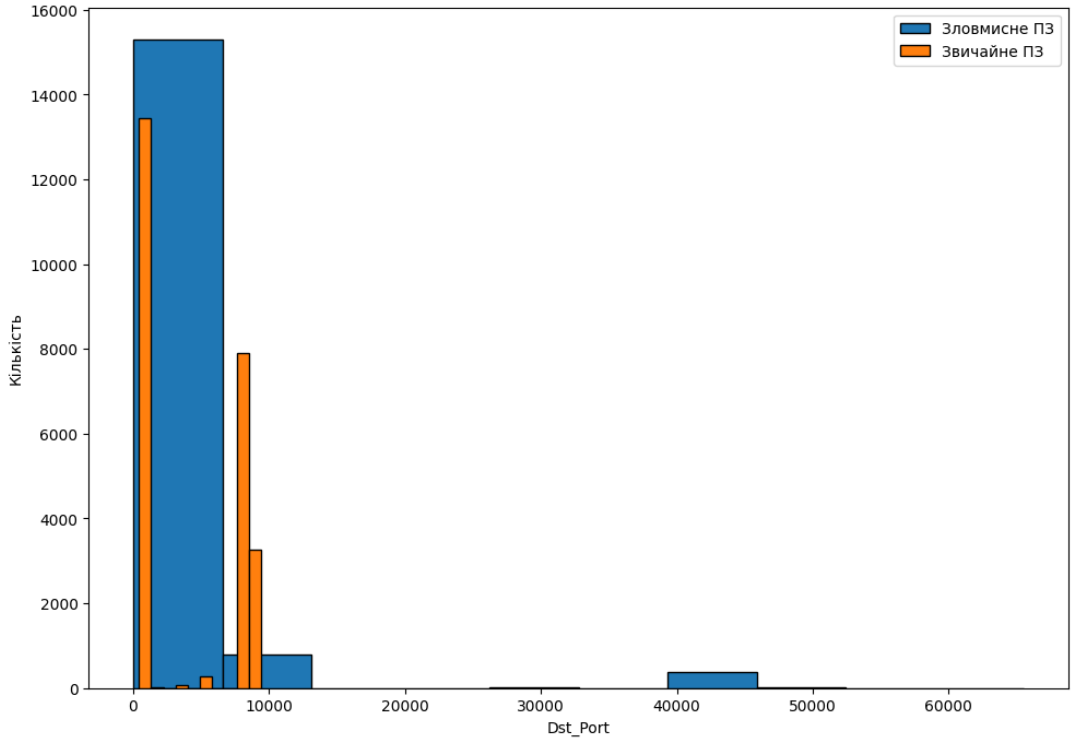


Рис. *19 Статистика параметру: вихідний порт*

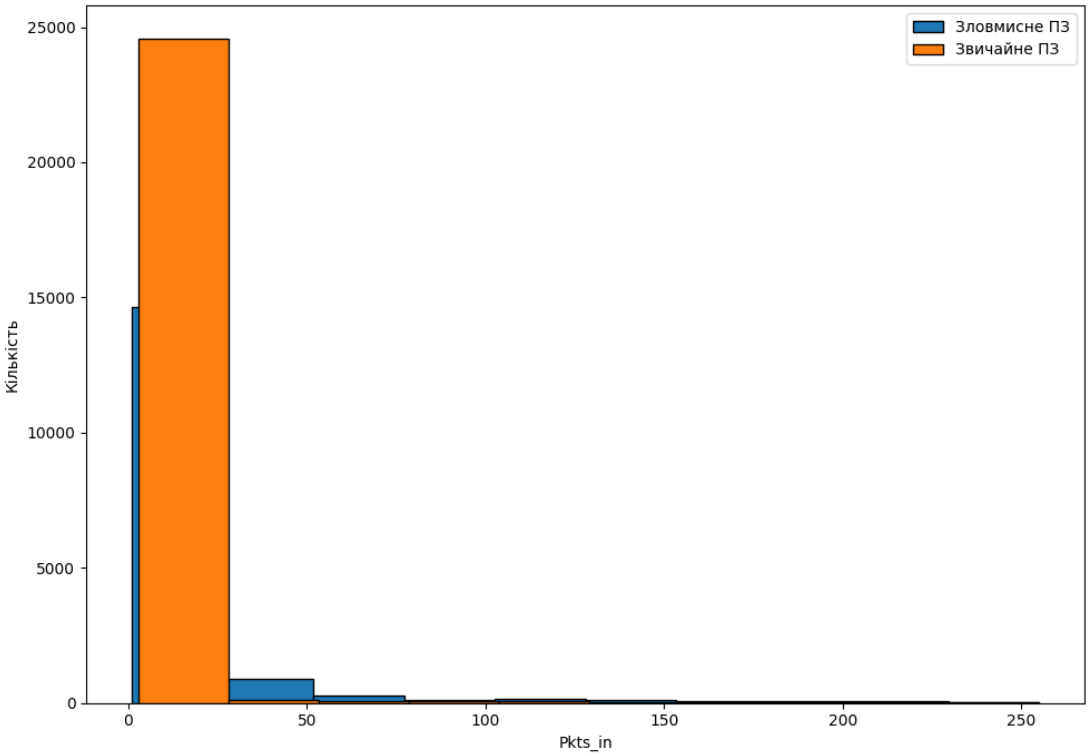


Рис. *20 Статистика параметру: пакетів отримано*

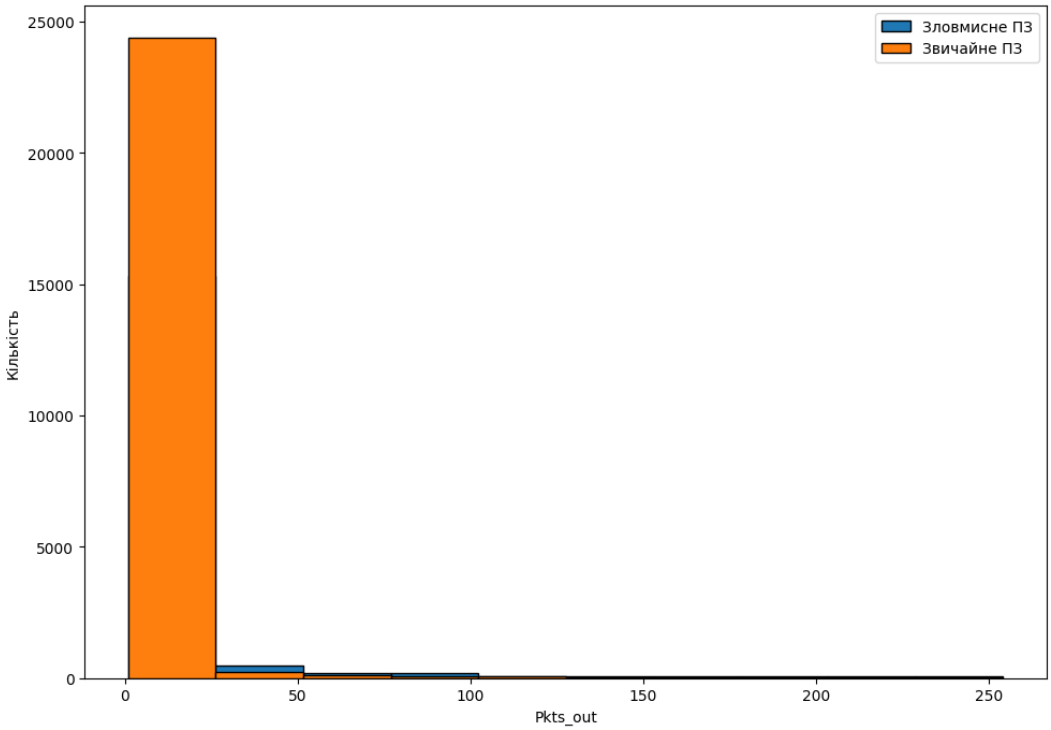


Рис. *21 Статистика параметру: пакетів відправлено*

3.1.3 Запуск процесу навчання

Після підготовки датасетів та встановлення sklearn, була виконано навчання за допомогою оптимізованої таблиці гіперпараметрів моделі GridSearchCV.

Для початку відокремимо залежну змінну `isMalware`, яка вказує алгоритму, чи зловмисне ПЗ у наборі даних що розглядається, чи ні.

Лістинг *1 Відокремлення залежної змінної*

```
reduced_y = mixed_Dataframe['isMalware']
reduced_x = mixed_Dataframe.drop(['isMalware'], axis=1);
train_X, test_X, train_Y, test_Y =
train_test_split(reduced_x, reduced_y, test_size=0.25)
```

Модель тренується на певній випадково вибраній частині даних, перевіряється на окремому наборі даних, а потім тестує себе на наборі даних, що

були відокремлені від датасету. Це може призвести до деяких проблем, оскільки відокремлюючи тестовий набір даних, можна випадково усунути частину спостережень, які були б ключовими для навчання оптимальної моделі. Збереження відсотка даних поза навчанням, допоможе б нашій моделі тренуватися більш ефективно. Якщо ще більш вдосконалити цей метод, то застосовується перехресна валідація, яка працює шляхом поділу вхідного набору даних на випадкові групи, утримання однієї групи як тесту та навчання моделі на інших групах. Цей процес повторюється для кожної групи k-разів, що проводиться в якості тестової групи.

GridSearchCV — це метод пошуку через кращі значення параметрів із заданого набору сітки параметрів. В основному це метод перехресної перевірки. модель і параметри потрібно вводити. Витягуються найкращі значення параметрів, а потім робляться прогнози. Якщо не використовувати гіпермараметри, можемо зіткнутися з перенавчанням, що означає, що наша модель машинного навчання занадто конкретно навчається на наборі навчальних даних коли застосовується до наборів даних для тестування. Або ми можемо зіткнутися з недостатнім навчанням, а це означає, що наша модель недостатньо тренується, або працює виняткових для моделі навчальних даних. Це також призводить помилок при застосуванні до тестових наборів даних.

Лістинг *2 Створення класифікатора з використанням гіперпараметрів*

```
objRF = RandomForestClassifier()
params = {
     "n_estimators": [7,20,50,100,200],
     "max_depth": [10,30,50,90,120],
     "min_samples_leaf" : [2, 4, 6, 8, 12]
     }
# Навчання моделі, використовуючи таблицю гіперпараметрів
clf = GridSearchCV(objRF, param_grid=params, cv=5)
clf.fit(train_X,train_Y)
```

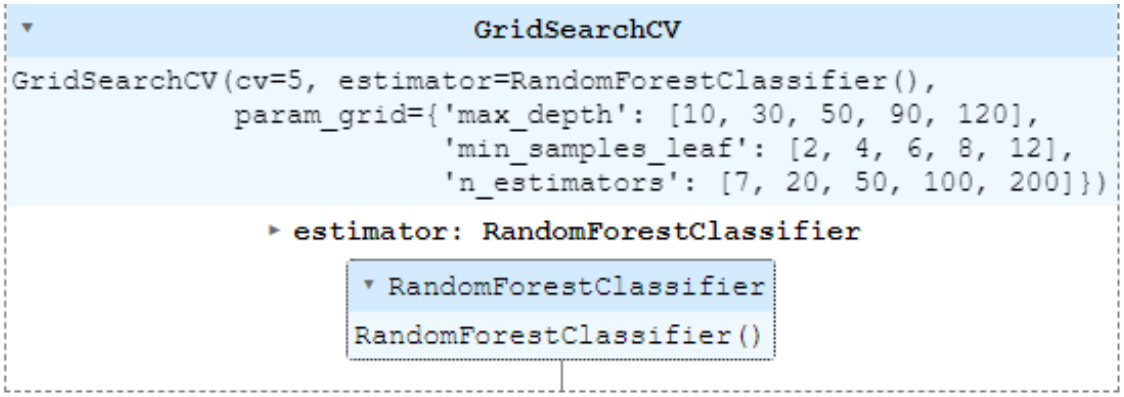


Рис. *22 Параметри таблиці GridSearchCV*

В результаті було отримано найкращі гіперпараметри моделі:

`{'max_depth': 30, 'min_samples_leaf': 2, 'n_estimators': 50}`

3.1.4 Перевірка якості навчання

Після успішного навчання можна проаналізувати ваги моделі. На рисунку 23 зображено найвагоміші параметри трафіку, що можуть бути експортовані у відкритому строковому чи hex-форматі.

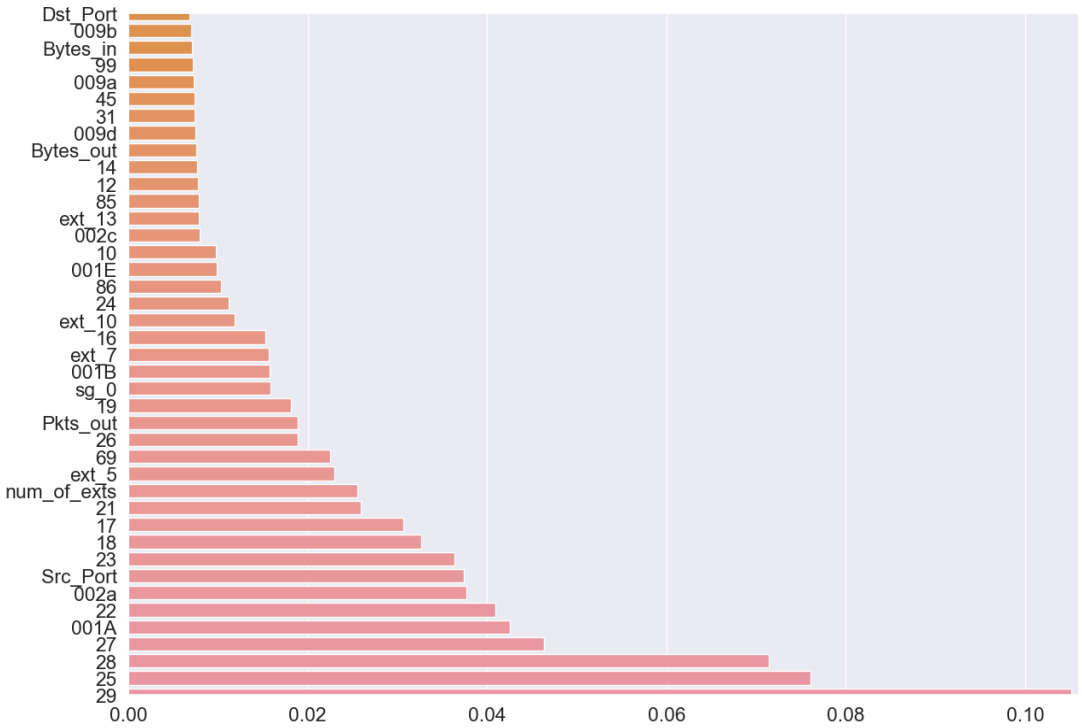


Рис. *23 Найвагоміші параметри навченої моделі*

3.2 Розробка та функціонал комп'ютерної системи для виявлення та обробки вторгнень

3.2.1 Налаштування середовища для розробки системи

Для розробки системи були використані наступні технології:

1. Мова програмування Python 3.10 — для самостійної реалізації функціоналу препроцессінгу трафіку. Zeek для автоматичного препроцессінгу трафіку.
2. Бібліотека sklearn — для створення й оперування моделлю Random Forest.
3. Бібліотека nmap та Wireshark – для захоплення та аналізу трафіка

3.2.2 Налаштування середовища для впровадження системи

В якості системи для тестування обрана Windows 11. Для роботи класифікатору потребується виконати наступні кроки:

1. Забезпечити підтримку мови програмування Python (3.10)
2. Необхідна підтримка бібліотеки sklearn, numpy, pandas, seaborn, joblib, matplotlib
3. Wireshark для дослідження файлів .pcap
4. Zeek IDS для інспекції пакетів в ручному режимі

3.2.3 Програмна архітектура

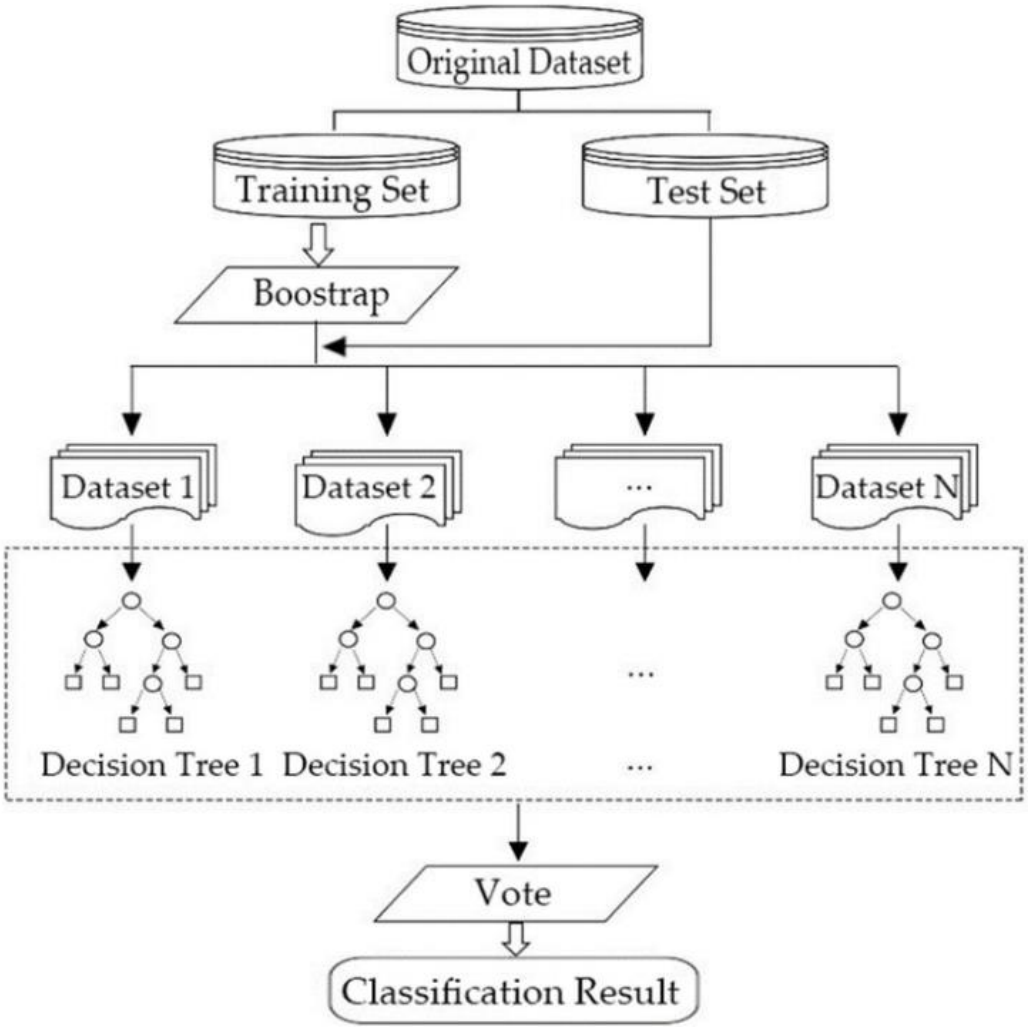


Рис. *24 Архітектура моделі*

3.2.4 Реалізація основного функціоналу

Для реалізації моделі була використана мова програмування Python. Розглянемо таблицю імпорту:

Лістинг *3 Таблиця імпорту*

```
import numpy as np
import pandas as pd
from math import sqrt;
import seaborn as sns
import joblib as joblib
import matplotlib.pyplot as plt
from sklearn import preprocessing
```

```
from sklearn.tree import plot_tree
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import ConfusionMatrixDisplay, accuracy_score, r2_score, confusion_matrix, mean_absolute_error, mean_squared_error, f1_score, log_loss, classification_report, confusion_matrix
from sklearn.model_selection import GridSearchCV, train_test_split
```

За допомогою модуля `pandas`, ми завантажуємо датасет у `DataFrame` прямо з csv таблиці.

Лістинг *4 Завантаження датасету та характеристики*

```
mixed_Dataframe = pd.read_csv('mixed_flows.csv')
print('Характеристики датасету: ')
with pd.option_context('display.max_rows', 7,
'display.max_columns', 7,
'display.precision', 5,):print(mixed_Dataframe)
mixed_Dataframe.isna().sum()
# Статистика
plot_dataset(mixed_Dataframe)
```

Робимо прогноз на тестових даних.

Лістинг *5 Матриця плутанини та предикт на тестових даних*

```
testPredict = clf.predict(test_X)
print(classification_report(test_Y, testPredict))

print('Точність на тестовій вибірці: ', accuracy_score(test_Y, testPredict))
cm = confusion_matrix(test_Y, testPredict)
print('Матриця плутанини:')
sns.heatmap(cm,annot=True,fmt="d")
```

Лістинг *6 Найважливіші функції та їх вага*

```
    print('На:')
    plot_features(clf.best_estimator_.feature_importances_,
train_X.columns)
```

Лістинг *7 Найкращі гіперпараметри моделі*

```
print('Найкращі гіперпараметри моделі:', clf.best_params_)
```

Вже навчена модель може бути збережена у файл формату .pkl, та відновлена з нього для подальшого використання.

Лістинг *8 Збереження й завантаження моделі*

```
print('Збережено модель: ')
print(clf)
joblib.dump(clf, "trained_model_1.pkl")
print('Завантажено модель: ')
model = joblib.load("trained_model_1.pkl")
print(model.best_estimator_.feature_names_in_)
```

Після завантаження моделі, можемо перевірити її на наявні функції моделі. На рисунку 25 зображено результат зберігання та відновлення моделі.

```
Збережено модель:
GridSearchCV(cv=5, estimator=RandomForestClassifier(),
             param_grid={'max_depth': [10, 30, 50, 90, 120],
                         'min_samples_leaf': [2, 4, 6, 8, 12],
                         'n_estimators': [7, 20, 50, 100, 200]})
Завантажено модель:
['Src_Port' 'Dst_Port' 'Bytes_in' 'Bytes_out' 'Pkts_in' 'Pkts_out'
 'entropy' 'byte_dist_std' 'byte_dist_mn' 'num_of_exts' '0' '1' '2' '3'
 '4' '5' '6' '7' '8' '9' '000a' '000b' '000c' '000d' '000e' '000f' '10'
 '11' '12' '13' '14' '15' '16' '17' '18' '19' '001A' '001B' '001E' '001F'
 '20' '21' '22' '23' '24' '25' '26' '27' '28' '29' '002a' '002b' '002c'
 '002d' '002e' '002f' '30' '31' '32' '33' '35' '36' '37' '38' '39' '003a'
 '003b' '003c' '003d' '003e' '003f' '40' '41' '42' '43' '44' '45' '46'
 '67' '68' '69' '006a' '006b' '006c' '006d' '84' '85' '86' '87' '88' '89'
 '008a' '008b' '008c' '008d' '008e' '008f' '90' '91' '92' '93' '94' '95'
 '96' '97' '98' '99' '009a' '009b' '009c' '009d' '009e' '009f' '00a0'
 '00a1' '00a2' '00a3' '00a4' '00a5' '00a6' '00a7' '00a8' '00a9' '00aa'
 '00ab' '00ac' '00ad' '00ae' '00af' '00b0' '00b1' '00b2' '00b3' '00b4'
 '00b5' '00b6' '00b7' '00b8' '00b9' '00ba' '00bb' '00bc' '00bd' '00be'
 '00bf' '00c0' '00c1' '00c2' '00c3' '00c4' '00c5' '00c6' '00c7' '00ff'
 '1301' '1302' '1303' '1304' '1305' '5600' 'c001' 'c002' 'c003' 'c004'
 'c005' 'c006' 'c007' 'c008' 'c009' 'c00a' 'c00b' 'c00c' 'c00d' 'c00e'
 'c00f' 'c010' 'c011' 'c012' 'c013' 'c014' 'c015' 'c016' 'c017' 'c018'
 'c019' 'c01a' 'c01b' 'c01c' 'c01d' 'c01e' 'c01f' 'c020' 'c021' 'c022'
 'c023' 'c024' 'c025' 'c026' 'c027' 'c028' 'c029' 'c02a' 'c02b' 'c02c'
...
 'ext_46' 'ext_47' 'ext_48' 'ext_49' 'ext_50' 'ext_51' 'ext_52' 'sg_0'
 'sg_1' 'sg_2' 'sg_3' 'sg_4' 'sg_5' 'sg_6' 'sg_7' 'sg_8' 'sg_9' 'sg_10'
 'sg_11' 'sg_12' 'sg_13' 'sg_14' 'sg_15' 'sg_16' 'sg_17' 'sg_18' 'sg_19'
 'sg_20' 'sg_21' 'sg_22' 'sg_23' 'sg_24' 'sg_25' 'ec_pts_0' 'ec_pts_1']
```

Рис. *25 Збереження й відновлення моделі з файлу формату .pkl та відображення функцій завантаженої моделі*

Також розглянемо корисну функцію збереження ваг функцій моделі. На рисунку 26 зображено вихідну таблицю зі всіма функціями та їх вагами.

Лістинг *9 Збереження таблиці ваг функцій*

```
fi_df.to_csv('features.csv')
```

No	feature_names	feature_importance
49	29	0.105150138
45	25	0.076049082
48	28	0.07137013
47	27	0.046368955
36	001A	0.042558746
42	22	0.040926007
50	002a	0.037680586
0	Src_Port	0.037428
43	23	0.036373176
34	18	0.03269597
33	17	0.030707575
41	21	0.025910425
9	num_of_exts	0.025531605
362	ext_5	0.022943907

Рис. *26 Збереження функцій моделі та їх ваг у файл .csv*

3.3 Висновки з розділу 3

1. Налаштоване середовище розробки. Встановлені необхідні інструменти та модулі для розробки застосунку.
2. Був оброблений вхідний датасет, згідно до вимог конструктора створення моделі.
3. Були реалізовані необхідні методи для створення архітектури моделі.
4. Була розроблена комп’ютерна система, яка в якості вхідних даних для навчання приймає вектор параметрів зловмисного та легітимного трафіку.
5. Модель в якості основного функціоналу режим навчання та режим тестування, в якому на основі вхідного мережевого потоку система має дати рішення щодо легітимності такого трафіку.

РОЗДІЛ 4 ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ РОБОТИ ХОСТОВОЇ СИСТЕМИ ВИЯВЛЕННЯ ТО ОБРОБКИ ВТОРГНЕНЬ З ВИКОРИСТАННЯМ МАШИННОГО НАВЧАННЯ

4.1 Результати навчання моделі

4.1.1 Аналіз результатів навчання моделі

У процесі розробки комп'ютерної системи була розроблена система для виявлення та обробки загроз на основі мережі з використанням моделі Random Forest. На рисунку 27 наведений графік точності навчання моделі на датасеті MCFP.

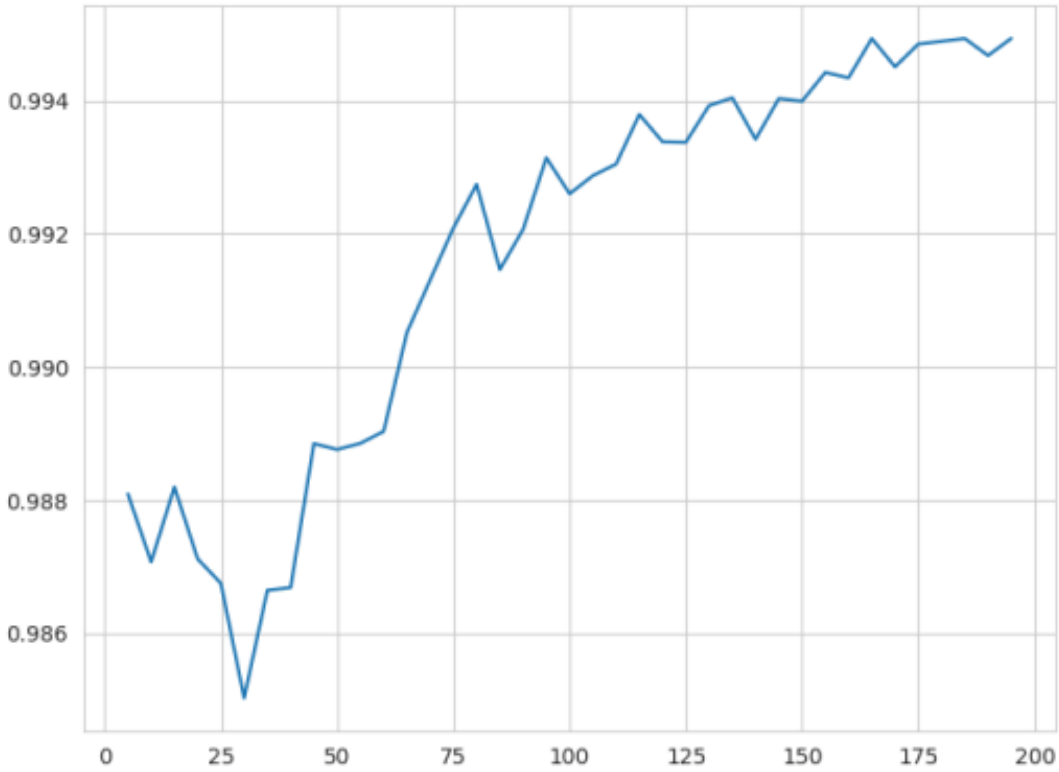


Рис. *27 Графік навчання моделі на датасеті MCFP*

Як можна побачити, коефіцієнт точності моделі досягає 0.992 вже при 75 параметрах, витягнутих з захопленого трафіку.

4.1.2 Аналіз точності роботи навченої моделі

Матриця плутанини — це матриця, яка надає інформацію наскільки точний алгоритм класифікації класифікує набір даних. Матричні сітки плутанини:

- Істинне позитивне (TP) і справжнє негативне значення (TN) – це значення, правильно передбачені алгоритмом класифікації. Алгоритм показав 6248 істинно позитивних значень та 4112 справжніх негативних значень.
- Помилково-позитивний (FP) і помилково-негативний (FN) – це значення, які неправильно передбачені класифікатором.

Найкраща точність моделі: 0.9991639716272454. Модель показала 1 помилково-позитивний та 6 помилково-негативний. На рисунку 28 зображена матриця плутанини.

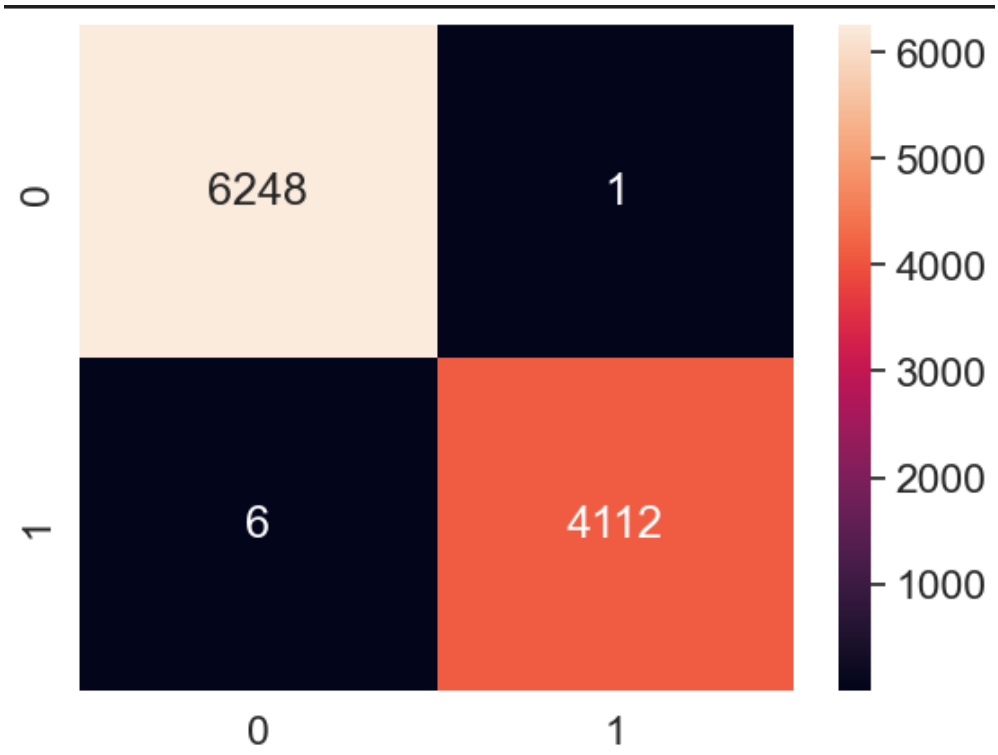


Рис. *28 Матриця плутанини*

На рисунку 29 зображено класифікаційний звіт моделі, що відображає показники точності, відкликання, F1 та підтримки моделі.

	precision	recall	f1-score	support
0	1.00	1.00	1.00	6249
1	1.00	1.00	1.00	4118
accuracy			1.00	10367
macro avg	1.00	1.00	1.00	10367
weighted avg	1.00	1.00	1.00	10367

Рис. *29 Класифікаційний звіт*

Precision (точність) — відображає, який відсоток ваших прогнозів був правильним. Точність — здатність класифікатора не позначати екземпляр позитивним, що насправді є негативним. Для кожного класу він визначається як відношення істинних позитивів до суми істинно позитивного і хибнопозитивного спрацьовування. Точність = TP/(TP + FP).

Відкликання (Recall) — який відсоток позитивних випадків модель вловила. Відкликання —здатність класифікатора знаходити всі позитивні екземпляри. Для кожного класу він визначається як відношення істинних позитивів до суми істинних позитивів і помилкових негативів. Відкликання = TP/(TP+FN).

Оцінка F1 — відповідає за відсоток позитивних прогнозів, що були правильними. Оцінка F1 є середньозваженим гармонійним значенням точності і нагадує таке, що найкращий бал - 1.0, а найгірший - 0.0 Оцінка F1 = 2*(Відкликання * Точність) / (Відкликання + Точність).

Підтримка (Suppoirt) — відповідає за кількість фактичних входжень класу в зазначений набір даних. Незбалансована підтримка в навчальних да-

них може свідчити про структурні слабкі місця в заявлених балах класифікатора. Підтримка не змінюється між моделями, а натомість діагностує процес оцінки.

4.2 Результати роботи розробленої комп'ютерної системи виявлення та обробки мережевих загроз на хості

В результаті була розроблена модель машинного навчання, На таблиці 3 зображено назву та вагу перших 10 найвпливовіших функцій моделі.

Назва параметру hex / string	Вага
0029 (TLS_KRB5_EXPORT_WITH_DES_CBC_40_MD5)	10.5
0025 (TLS_KRB5_WITH_IDEA_CBC_MD5)	7.60
0028 (TLS_KRB5_EXPORT_WITH_RC4_40_SHA)	7.13
0027 (TLS_KRB5_EXPORT_WITH_RC2_CBC_40_SHA)	4.63
001A (TLS_DH_anon_WITH_DES_CBC_SHA)	4.25
0022 (TLS_KRB5_WITH_DES_CBC_MD5)	4.09
0002 (TLS_RSA_WITH_NULL_SHA)	3.76
Src_Port	3.74
23 (TLS_KRB5_WITH_3DES_EDE_CBC_MD5)	3.6
18 (TLS_DH_anon_WITH_RC4_128_MD5)	3.2

Таблиця 3. *Таблиця функцій моделі*

На рисунках 30-38 розглянемо співвідношення у датасеті найвпливовіших функцій між зловмисним та звичайним ПЗ.

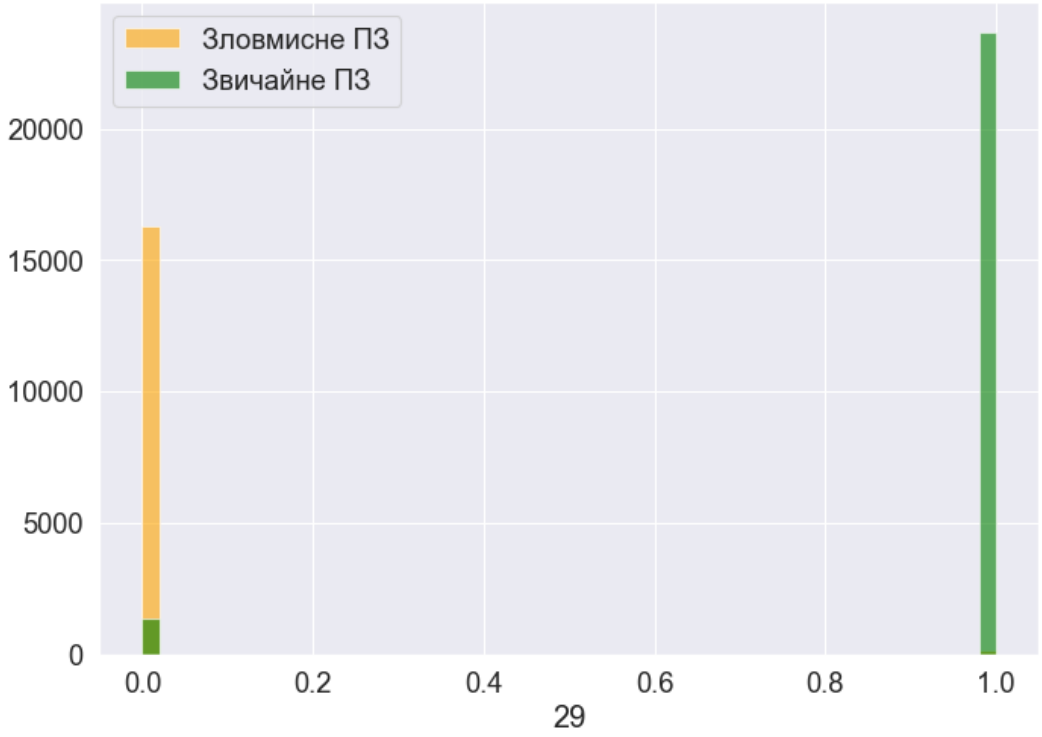


Рис. *30 Параметр навченої моделі: шифр TLS_KRB5_EX-PORT_WITH_DES_CBC_40_MD5*

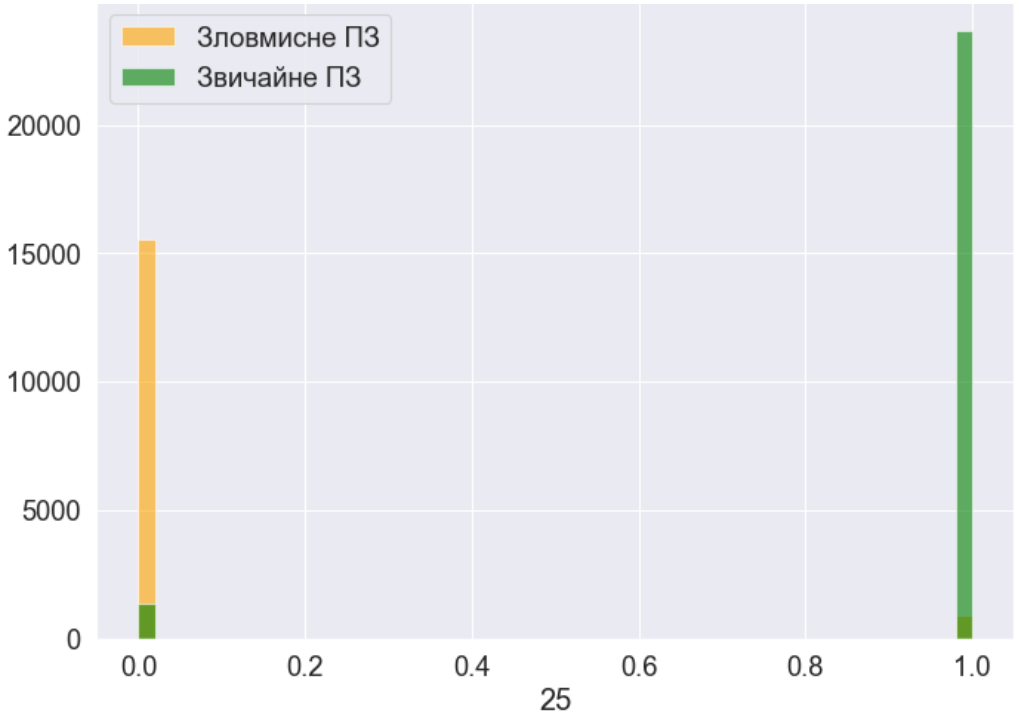


Рис. *31 Параметр навченої моделі: шифр TLS_KRB5_WITH_IDEA_CBC_MD5*

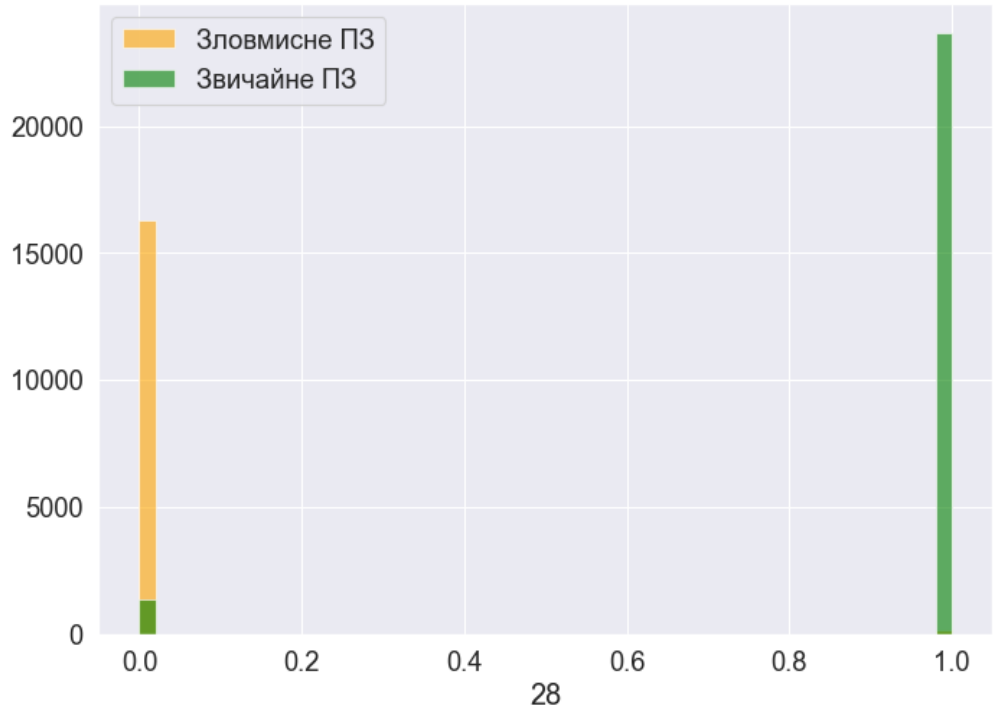


Рис. *32 Параметр навченої моделі: шифр TLS_KRB5_EX-PORT_WITH_RC4_40_SHA*

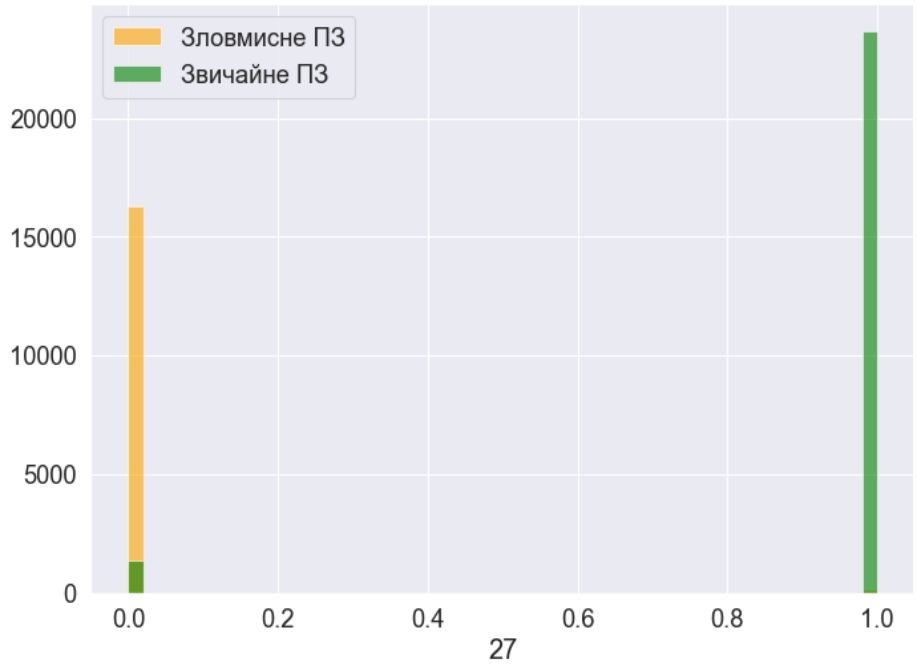


Рис. *33 Параметр навченої моделі: шифр TLS_KRB5_EX-PORT_WITH_RC2_CBC_40_SHA*

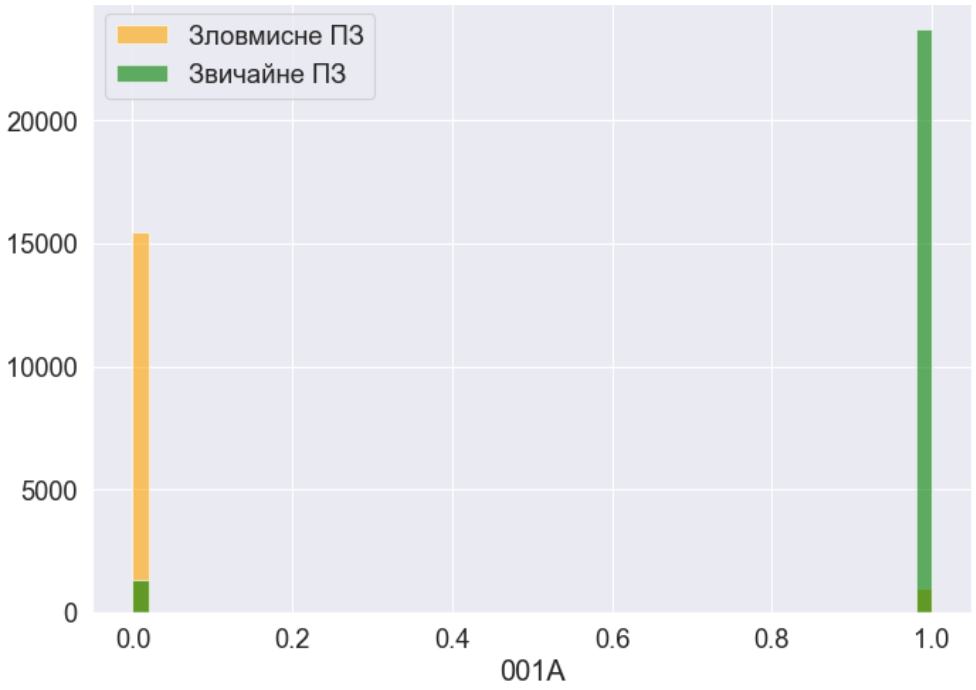


Рис. *34 Параметр навченої моделі: шифр TLS_DH_anon_WITH_DES_CBC_SHA*

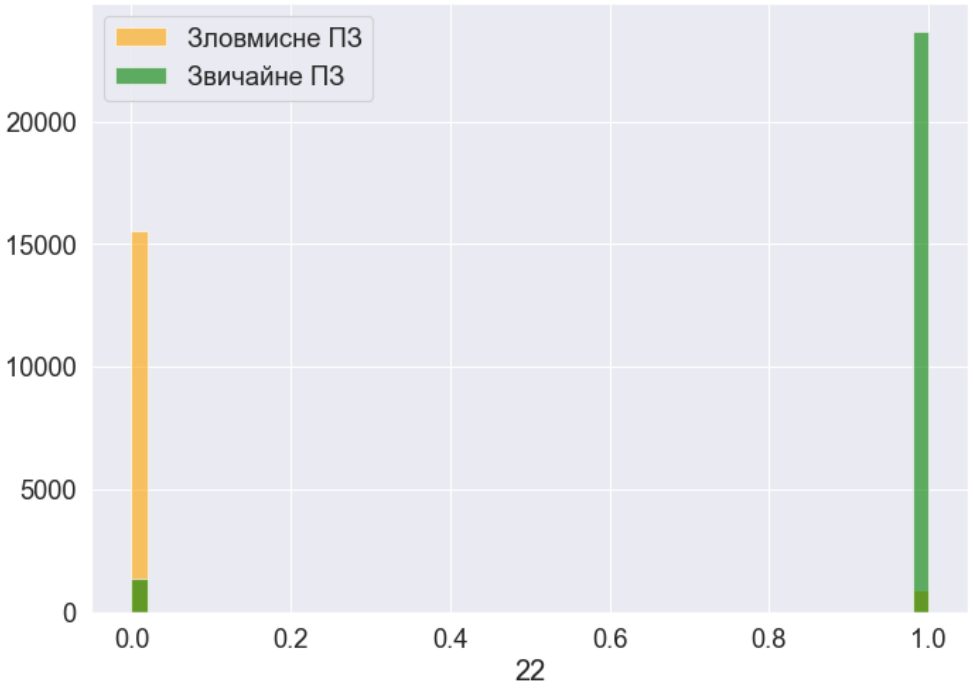


Рис. *35 Параметр навченої моделі: шифр TLS_KRB5_WITH_DES_CBC_MD5*

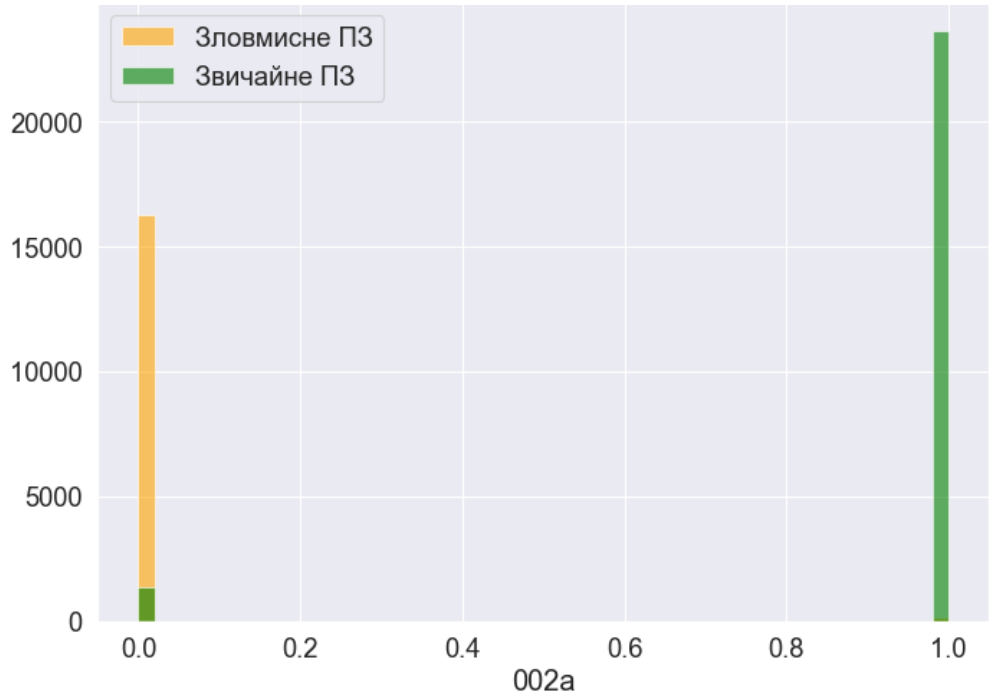


Рис. *36 Параметр навченої моделі: шифр TLS_RSA_WITH_NULL_SHA*

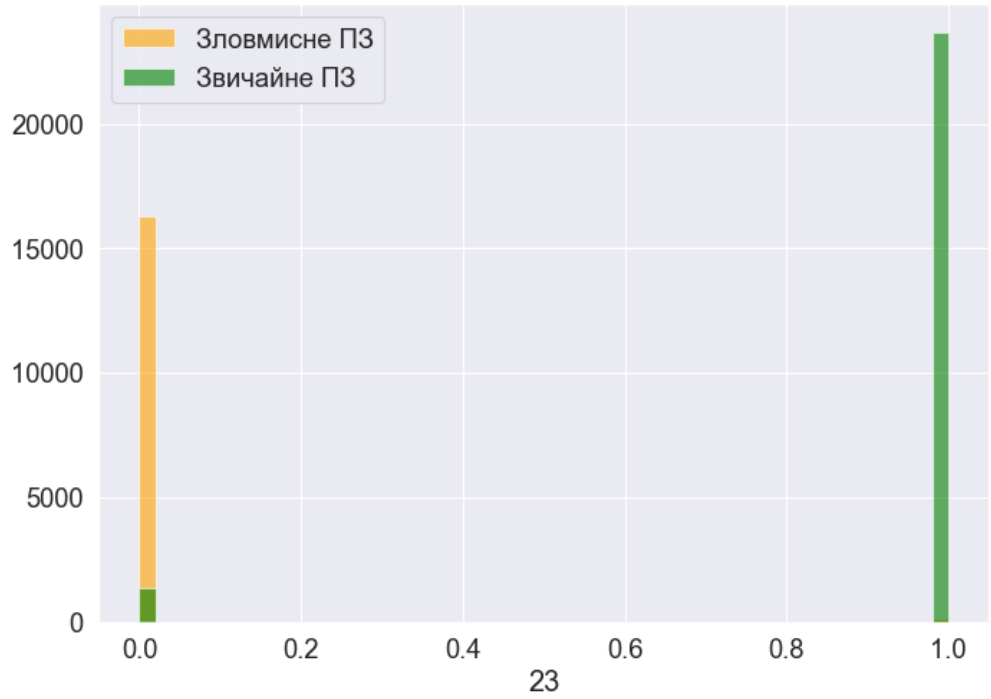


Рис. *37 Параметр навченої моделі: шифр TLS_KRB5_WITH_3DES_EDE_CBC_MD5*

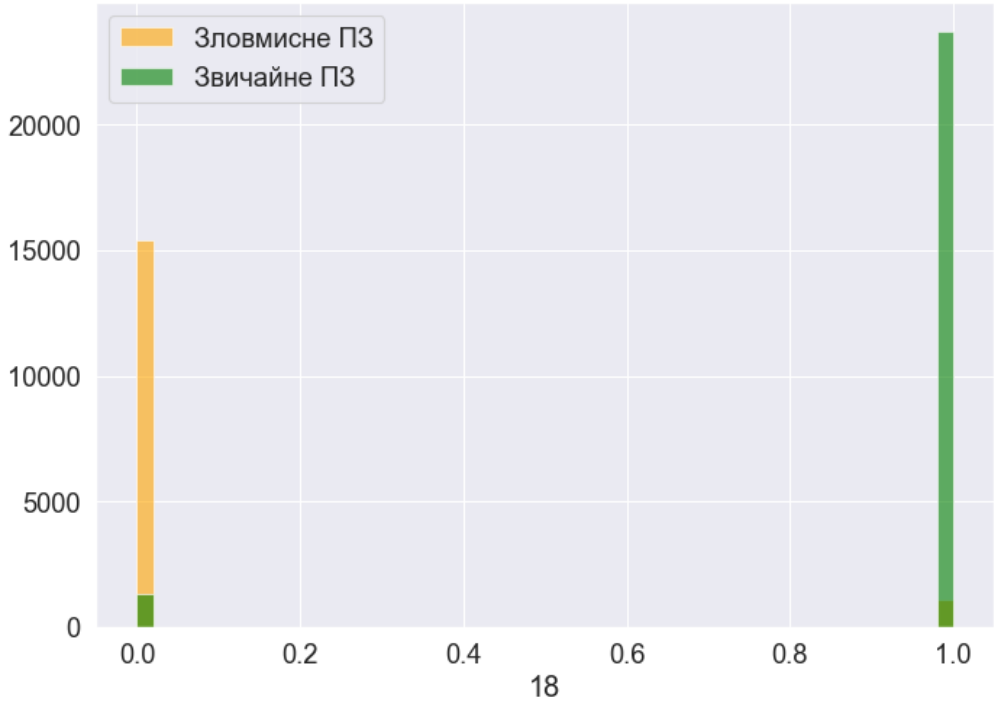


Рис. *38 Параметр навченої моделі: шифр TLS_DH_anon_WITH_RC4_128_MD5*

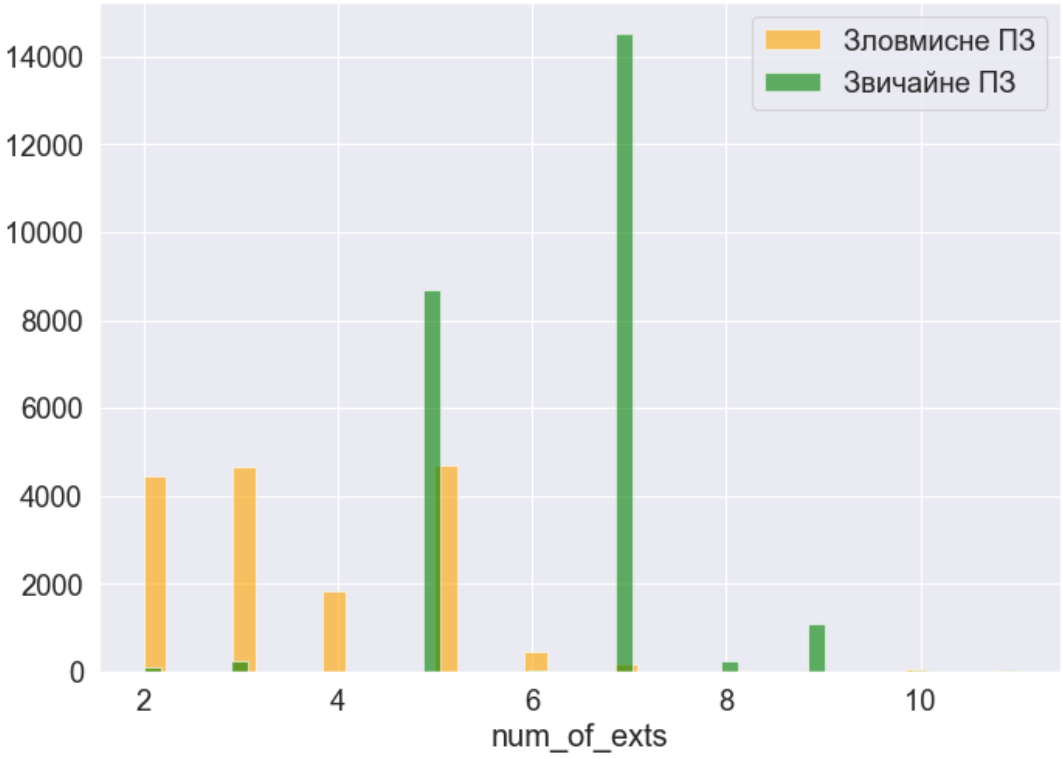


Рис. *39 Параметр навченої моделі: num_of_extension*

4.2.1 Аналіз швидкості навчання та використання пам’яті системою

Навчання моделі на обробленому датасеті займає близько хвилини. Сам датасет займає 191 мегабайт пам’яті в комп’ютері. При використанні таблиці гіперпараметрів з кросвалідаційною валідацією, час навчання значно збільшується, але цього потребує методологія навчання для більш якісної моделі.

4.2.2 Варіанти покращення точності роботи системи

Точність роботи розробленої системи насамперед залежить від якості навчання моделі машинного навчання. Загалом, можна виділити наступні рекомендації щодо покращення моделі:

- Експериментувати із різними датасетами, та гіперпараметрами моделі
- Автоматизувати збір файлів захоплення як доброякісних, так і шкідливих.
- Використовувати інший набір функцій для підвищення надійності та TPR .
- Використовувати функції інших протоколів (DNS, HTTP) для підвищення надійності та TPR.
- Порівняння класифікатора з іншими типами класифікаторів.

4.3 Висновки з розділу 4

1. Була розгорнута модель машинного навчання, яка навчалася на датасеті MCFP, показала високу точність при навчанні.
2. Були представлені характеристики та найбільш вагомі функції моделі.
3. Для покращення точності роботи навчених моделей існують рекомендації, описані в розділі 4.2.2
4. Отримана при тестуванні системи найкраща точність моделі становить 0.9991639716272454 з 1. Модель показала 1 помилково-позитивний та 6

помилково-негативних результатів на тестових даних, що є гарним результатом.

5. Застосований підхід до навчання моделі з вчителем, та використанням таблиці гіперпараметрів моделі доказує свою ефективність в рішенні задачі класифікації трафіку.

ВИСНОВКИ

1. Досліджена проблема виявлення та обробки вторгнень на основі мережі.
2. Обрані засоби для вирішення поставленої проблеми: з розглянутих моделей машинного навчання, обрана найбільш ефективна — модель випадкового лісу;з розглянутих бібліотек для навчання та використання алгоритмів машинного навчання — вибрана бібліотека sklearn; з розглянутих датасетів із захопленими мережевими потоками, обрані та оброблені актуальні датасети.
3. Виконано навчання моделі: налаштоване середовище для навчання, підготовлені вхідні дані для навчання та тестування, виконано навчання й тестування.
4. Розроблена комп'ютерна система, яка виявляє зловмисне ПЗ шляхом навчання й тренування моделі.
5. Досліджені результати навчання: точність навченої моделі, список з вагами параметрів, навантаження на комп'ютер при різних режимах роботи ПЗ. Модель навчалася на датасетах MCFP, які містять близько 400 прикладів захопленого трафіку від зловмисного ПЗ. На навчених прикладах модель виявляє близько 18 сімейств ботнетів та інших типів зловмисного ПЗ.
6. Досліджені результати роботи розробленої системи: система працює з гарною точністю, якщо в якості прикладів використовувати ті ж сімейства зловмисного ПЗ, яким модель була навчена. Точність рішення, яке видає система, залежить від якості навчання від якості та актуальності датасету.
7. Поставлена проблема хостової системи виявлення та обробки вторгнень з використанням машинного навчання вирішена з допустимою точністю та в межах обраних методів дослідження.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Anderson B., McGrew D. Identifying encrypted malware traffic with contextual flow data. *2016 ACM Workshop on Artificial Intelligence and Security.* October 2016. P. 35-46.
2. Anderson B., McGrew D., Paul S. Deciphering Malware's use of TLS (without Decryption). *Journal of Computer Virology and Hacking Techniques*. 2016. Vol. 14, P. 195-211.
3. Anderson B., McGrew D. Machine learning for encrypted malware traffic classification: accounting for noisy labels and non-stationarity. *23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017. P. 1723-1732
4. Jiyuan Liu, Yingzhi Zeng, Jiangyong Shi, Yuexiang Yang, Rui Wang, Liangzhong. MalDetect: A Structure of Encrypted Malware Traffic Detection. Computers, Materials & Continua. 2019. Vol.60, No.2. P. 721–739
5. Saffari A., Leistner C., Santner J., Godec M., Bischof H. On-line random forests. *IEEE 12th International Conference on Computer Vision Workshops*. 2009. P. 1393-1400.
6. Leo Breiman. Random forests. Mach. Learn. 2011. P. 5–32
7. Milenkoski A., Vieira M., Kounev S., Avritzer A., Bryan D. Payne. Evaluating Computer Intrusion Detection Systems: A Survey of Common Practices. ACM Computing Surveys. September 2015. Vol. 48, No 12. P. 1–41.
8. XGBoost: A Scalable Tree Boosting System. Tianqi Chen, Carlos Guestrin. *KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016. P 785–794.
9. Garcia, Grill, Stiborek. CTU-13 dataset. 2014. URL: https://www.stratosphereips.org/datasets-malware
10. Erquiaga, García and García Garino. MCFP dataset. 2017. URL: https://mcfp.weebly.com/analysis

11. Хаджійський О., магістрант 1 курсу, Полякова Н.П. доцент — науковий керівник. Олег ХОСТОВА СИСТЕМА ВИЯВЛЕННЯ ТА ОБРОБКИ МЕРЕЖЕВИХ ЗАГРОЗ. Молода наука-2022 : зб. наук. праць студентів, аспірантів і молодих вчених. Запоріжжя : ЗНУ, 2022.176-178.
12. Хаджійський О.О. магістрант 2 курсу, Полякова Н.П. доцент — науковий керівник. Хостова система виявлення та обробки мережевих загроз. 18-20 жовтня 2022 року II Всеукраїнська науково-практична конференція за участю молодих науковців «АКТУАЛЬНІ ПИТАННЯ СТАЛОГО НАУКОВО-ТЕХНІЧНОГО ТА СОЦІАЛЬНО-ЕКОНОМІЧНОГО РОЗВИТКУ РЕГІОНІВ УКРАЇНИ»: зб. наук. праць студентів, аспірантів і молодих вчених. Запоріжжя : ЗНУ, 2022. С.317-318.