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

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

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

магістр
(назва освітнього ступеня)

на тему: Дослідження методів машинного навчання для виявлення мережевих атак

Виконав: студент VI курсу, групи СБм-61
спеціальності 125 Кібербезпека
(шифр і назва спеціальності)

Липа Б. М.
(підпис) (прізвище та ініціали)

Керівник *Загородна Н. В.*
(підпис) (прізвище та ініціали)

Нормоконтроль *Лобур Т.Б.*
(підпис) (прізвище та ініціали)

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

Рецензент
(підпис) (прізвище та ініціали)

Тернопіль
2022

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

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

Кафедра Кібербезпеки
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Загородна Н.В.
(підпис) (прізвище та ініціали)
«____» _____ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Магістр
(назва освітнього ступеня)

за спеціальністю 125 Кібербезпека
(шифр і назва спеціальності)

Студенту Липі Борису Михайловичу
(прізвище, ім'я, по батькові)

1. Тема роботи Дослідження методів машинного навчання для виявлення мережевих атак

Керівник роботи Загородна Наталія Володимирівна, к.т.н., зав. кафедри КБ
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від «_25_» _листопада__ 2022 року № 4/7-966

2. Термін подання студентом завершеної роботи 14 грудня 2022р.

3. Вихідні дані до роботи Дослідження методів машинного навчання для виявлення мережевих атак

4. Зміст роботи (перелік питань, які потрібно розробити): Вступ, 1 Аналіз джерел даних Мережевих атак та алгоритмів машинного навчання, 1.1 Існуючі данні що можуть використовуватись для створення систем виявлення вторгнень, 1.2 Набір даних CICIDS 2017, 1.3 Набір даних Bot-IoT, 1.4 Сценарії створення трафіку в наборі даних Bot-IoT, 1.5 Аналіз поширених атак, 1.6 Алгоритми машинного навчання які використовуються в інформаційній безпеці, 2 Підходи до створення системи виявлення вторгнень, 2.1 Системи виявлення вторгнень, 2.2 Системи що використовують сигнатури для виявлення вторгнень, 2.3 Система виявлення вторгнень на основі аномалій, 2.4 Типи систем виявлення вторгнень на основі вхідних даних, 2.5 Техніки імплементації систем виявлення вторгнень на основі аномалій, 2.5.1 Техніки що базуються на використання статистики, 2.5.2 Техніки що базуються на знаннях, 2.5.3 Техніки що базуються на машинному навчанні, 2.6 Гібридні техніки в системі виявлення вторгнень, 3 Розробка системи виявлення мережевих атак та її результати, 3.1 Попередня обробка тренувального набору даних, 3.2 Результати класифікації, 3.3 Реалізація рішення для перевірки трафіку в реальному часі, 3.4 Тестування рішення для перевірки трафіку в реальному часі, 4 Охорона праці та безпека в надзвичайних ситуаціях, 4.1 Охорона праці, 4.2 Характеристика стихійних лих, аварій (катастроф) та їх наслідків, 4.3 Види стихійних лих, 4.4 Висновки до 4 розділу, Висновки, Перелік використаних джерел.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Титульна сторінка. 2 Тема. Мета. Об'єкт. Предмет дослідження. 3 Набори даних. 4 Набір даних Bot-IoT. 5 Поширені мережеві атаки. 6 Алгоритми машинного навчання. 7 Системи виявлення вторгнень. 8 На основі сигнатур. 9 На основі аномалій. 10 Порівняння систем. 11 Типи на основі вхідних даних. 12 Класи систем на основі аномалій. 13 Попередня обробка даних. 14 Розподіл трафіку в даних. 15 Балансування даних. 16 Результати незбалансованої моделі. 17 Результати збалансованої моделі. 18 Важливість характеристик. 19 Схема роботи програми. 20 Висновки. 21 Завершальний слайд.

6. Консультанти розділів роботи

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

7. Дата видачі завдання 14 листопада 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	14.11.2022-15.11.2022	*Виконано*
2.	Підбір наукових джерел про системи виявлення		
	вторгнення	16.11.2022-20.11.2022	*Виконано*
3.	Переклад та опрацювання наукових джерел про		
	дослідження методів створення системи		
	виявлення вторгнення	21.11.2022-23.11.2022	*Виконано*
4.	Виконання дослідження щодо аналізу інструментів		
	для розробки системи виявлення вторгнення		
	на основі виявлення аномалій	24.11.2022-27.11.2022	*Виконано*
5.	Оформлення розділу «Аналіз джерел даних мережевих		
	атак та алгоритмів машинного навчання»	28.11.2022-30.11.2022	*Виконано*
6.	Оформлення розділу «Підходи до створення системи		
	виявлення вторгненнь»	01.12.2022-04.12.2022	*Виконано*
7.	Оформлення розділу «Розробка системи виявлення		
	мережевих атак та її результати	05.12.2022-07.12.2022	*Виконано*
8.	Виконання завдання до підрозділу «Охорона праці»	08.12.2022-09.12.2022	*Виконано*
9.	Виконання завдання до підрозділу «Безпека в надзви-		
	чай них ситуаціях»	10.12.2022-11.12.2022	*Виконано*
10.	Оформлення кваліфікаційної роботи	12.12.2022-13.12.2022	*Виконано*
11.	Нормоконтроль	14.12.2022-15.12.2022	*Виконано*
12.	Перевірка на плагіат	9.12.2022	*Виконано*
13.	Попередній захист кваліфікаційної роботи	16.12.2022	*Виконано*
14.	Захист кваліфікаційної роботи	.12.2022	

Студент _____ Липа Б. М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Загородна Н. В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Дослідження методів машинного навчання для виявлення мережевих атак // Кваліфікаційна робота освітнього рівня «Магістр» // Липа Борис Михайлович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра кібербезпеки, група СБм-61 // Тернопіль, 2022 // С. 83, рис. – 21, табл. – 6, додат. – 3, бібліогр. – 62.

Ключові слова: МАШИННЕ НАВЧАННЯ, ШТУЧНИЙ ІНТЕЛЕКТ, СИСТЕМА ВИЯВЛЕННЯ ВТОРГНЕНЬ, МЕРЕЖА.

Кваліфікаційна робота присвячена дослідженню методів машинного навчання для виявлення мережевих атак.

У першому розділі проводиться аналіз даних які можуть бути використані для створення систем виявлення вторгнень та сценарії що використовувались для створення таких даних. Також було розглянуто поширені мережеві атаки та алгоритми машинного навчання що використовуються для інформаційної безпеки.

В другому розділі розглядаються системи виявлення вторгнень, варіанти їх реалізації, розміщення та методів які використовуються для їх створення.

У третьому розділі проведені експериментальні дослідження щодо створення системи виявлення мережевих вторгнень, її тестування та застосування такої системи для аналізу мережевого трафіку в реальному часі.

ANNOTATION

Study of Machine Learning Methods of Network Attacks Detection // Qualification paper of the educational level “Master” // Borys Lypa // Ternopil Ivan Puluj National Technical University, Faculty of Computer Information Systems and Software Engineering, Department of Cyber Security, SBm-61 group // Ternopil, 2022 // P. 83, fig. - 21, tables - 6, annexes - 3, references - 62.

Key words: machine learning, artificial intelligence, intrusion detection system, network

The qualification work is devoted to the study of machine learning methods of network attacks detection.

In the first section, the analysis of data that could be used for creation of intrusion detection systems, as well as scenarios that are used to create such datasets. Analysis of the well-known network attacks and algorithms that are used for the information security are also carried out.

The second section considers the intrusion detection systems, approaches to implementing an IDS, placement of the system and methods that are used in IDS.

In the third chapter, experimental studies on the creation of the intrusion detection system, testing of such system as well as applying it to analysis of the network traffic in real time.

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

AI – Artificial intelligence

AIDS – Anomaly based Intrusion Detection System

CLI – Command Line Interface

DoS – Denial of Service

DDoS – Distributed Denial of Service

IDS – Intrusion Detection System

IoT – Internet of Things

SIDS – Signature based Intrusion System

SVM – Support Vector Machines

ML – Machine Learning

БД - База даних

ПЗ - Програмне забезпечення

ЗМІСТ

ВСТУП

Актуальність теми. За останнє десятиліття Інтернет значно розширився як у розмірі, так і в складності. У той же час з'явилися нові типи підключених пристроїв. Як наслідок, мережеві системи піддаються декільком уразливостям безпеки, включаючи вторгнення через збільшення складності мережі. Згідно із звітом Internet Security Threat Report від Symantec про загрози інтернет-безпеці за 2017 рік, у 2016 році було зареєстровано понад три мільярди атак нульового дня, а їх кількість стала значно більшою [1]. В статистиці Breach Level Index опублікованій The Thales приблизно дев’ять мільярдів записів даних було втрачено або викрадено зловмисниками з 2013 року [2] та вказано що кількість таких випадків зростає.

Мета і задачі дослідження. Метою роботи є виявлення найпоширеніших мережевих атак (DDOS, Probe, U2R та ін.) використовуючи методи машинного навчання.

Для досягнення поставленої мети було потрібно виконати наступні завдання:

- Аналіз методів і літератури в сфері дослідження.
- Формування та обґрунтування вимог до інформаційної системи виявлення мережевих атак.
- Вибір засобів розробки та побудова архітектури системи.
- Написання коду модуля, його аналіз та забезпечення якості.
- Створення тест-плану для даного модуля.
- Тестування модуля за допомогою доступних даних.

Об’єкт дослідження. Мережевий трафік.

Предмет дослідження. Формування вимог до модуля виявлення атак у мережевому трафіку, його розробка та тестування.

Наукова новизна одержаних результатів кваліфікаційної роботи полягає у тому, що отримано результати експериментального дослідження створеного

модуля виявлення вторгнень на тестовому наборі даних та у реальних умовах за допомогою створеної системи виявлення аномалій в реальному часі.

Практичне значення одержаних результатів. Створена система виявлення вторгнень може використовуватись для виявлення мережевих атак у комп’ютерних мережах.

Апробація результатів магістерської роботи. Основні результати проведених досліджень обговорювались на: Дев’ятій міжнародній конференції «ENGINEER OF XXI CENTURY» (м. Б'єльсько-б'яла), Третій міжнародній конференції «CHALLENGES TO NATIONAL DEFENCE IN CONTEMPORARY GEOPOLITICAL SITUATION» (м. Вільнюс)

Публікації. Основні результати кваліфікаційної роботи опубліковано у праці конференцій (див. Додаток А)

РОЗДІЛ 1. АНАЛІЗ ДЖЕРЕЛ ДАНИХ МЕРЕЖЕВИХ АТАК ТА АЛГОРИТМІВ МАШИННОГО НАВЧАННЯ

1.1 Існуючі дані що можуть використовуватись для створення систем виявлення вторгнень

Сутність системи виявлення вторгнень полягає у виявленні аномального трафіку і для її оцінки потрібно використовувати існуючі набори даних. Часто такі набори даних не є публічними, тому що містять інформацію про мережевий трафік того хто створив набір даних, таким чином його розповсюдження може порушити конфіденційність. Однак все ж існують загальнодоступні синтетичні набори даних такі як KDD, NSL-KDD, CICIDS-2017, Bot-IoT та інші, кожен з яких має свої переваги та недоліки. Часто ці набори даних використовуються як своєрідний тест для порівняння нових систем виявлення вторгнення.

Перший загальнодоступний набір даних для використання в системах виявлення вторгнень був створений у 1998 році DARPA (Defence Advanced Research Project Agency) агенцією, пізніше вони також створили датасет KDD98. Їх метою було створити реалістичне середовище що моделювало би невелику базу ВПС США [39]. Хоча цей набір даних і досі використовується уже неодноразово було доведено що він погано відображає реальні дані та його точність багато разів ставилась під сумнів [40]. Зібрані данні являли собою мережеві пакети та журнали хостів, що були отримані із експериментального стенду який працював 2 місяці і за цей час окрім нормального трафіку було імітовано декілька вторгнень у систему.

Розмір зібраного мережевого трафіку становив близько чотирьох гігабайтів та майже 5 мільйонів записів, із мережевого трафіку було добуто 41 характеристику і кожен запис було позначено як нормальний трафік або аномалію.

Отримані дані — це серія сеансів TCP, які щоб можна було їх класифікувати як нормальний трафік або атаку відбувались лише у визначений період часу та лише з визначеної вихідної IP-адреси на цільову IP-адресу. Данні включають в себе декілька атак, імітованих у військовому мережевому середовищі. Пізніше на основі цього набору даних було створено датасет KDD Cup99, він використовувався у 3-му міжнародному конкурсі видобування знань та даних.

Так як мережі сильно змінились від часу створення цих наборів даних, наприклад з'явились нові атаки та нове зловмисне програмне забезпечення, тому вони вважаються застарілими. Також змінилась поведінка зловмисників, мережеві топології, швидкість мереж та об'єм даних який передається, програмне забезпечення. Незважаючи на це дослідницька спільнота систем виявлення вторгнень все ще використовує KDD99 в якості еталону для досліджень.

1.2 Набір даних CICIDS 2017

В попередній роботі ми розробляли систему що використовує машинне навчання для виявлення мережевих атак використовуючи набір даних CICIDS2017.

Набір даних CICIDS2017 містить як нормальну поведінку так і перелік сучасних атак із використанням зловмисного програмного забезпечення, серед яких є атака на відмову в обслуговуванні, розподілена атака на відмову в обслуговуванні, метод грубої сили на протокол FTP та SSH, атака на веб сторінки, атака з використанням інсайдера, використання Botnet мереж та Heartbleed вразливість [41]. Записи класифіковано як нормальний трафік чи як певну атаку за допомогою часу коли запис відбувався, проколу що був використаний, IP-адреси та портів одержувача та джерела. Для створення набору даних було створено експериментальний стенд що містить вузли на які

встановлено різні версії операційної системи Linux, MacOS та Windows (Windows XP, Windows 7, Windows 8, Windows 10), також стенд містить мережеве обладнання як модем, комутатор та маршрутизатор, та брандмауер. Із записаного трафіку було добуто 80 характеристик, набір даних містить .csv файли із трафіком у вигляді записів із характеристиками та міткою чи це нормальний трафік чи атака.

Набір даних CICIDS2017 містить нормальний трафік та найновіші поширені атаки, які нагадують справжні реальні дані (PCAP). Він також містить результати аналізу мережевого трафіку за допомогою CICFlowMeter із позначеними потоками на основі мітки часу, IP-адрес джерела та призначення, портів джерела та призначення, протоколів і атак (файли CSV). Також доступне визначення вилучених ознак.

Створення реалістичного фонового трафіку було головним пріоритетом у створенні цього набору даних. Було використано систему B-Profile для профілювання абстрактної поведінки людських взаємодій і створення реалістичного фонового трафіку. Для цього набору даних було створено абстрактну поведінку 25 користувачів на основі протоколів HTTP, HTTPS, FTP, SSH і електронної пошти.

Період збору даних розпочався о 9 годині ранку в понеділок, 3 липня 2017 року, і закінчився о 17 годині. у п’ятницю, 7 липня 2017 р., загалом 5 днів. Понеділок є звичайним днем і включає лише нормальний трафік. Реалізовані атаки включають метод підбору грубою силою FTP, метод підбору грубою силою SSH, DoS, Heartbleed, веб-атаку, проникнення, ботнет і DDoS. Їх виконували вранці та вдень у вівторок, середу, четвер і п'ятницю..

1.3 Набір даних Bot-IoT

Запропонований тестовий стенд складається з трьох компонентів, а саме: мережевих кінцевих пристроїв, змодельованих сервісів IoT, а також функцій

збору та аналітики даних. По-перше, мережеві пристрої включають звичайні та атакуючі віртуальні машини (ВМ) з додатковими мережевими пристроями, такими як брандмауер. По-друге, змодельовані послуги IoT, які містять деякі послуги IoT, такі як метеостанція. Вони моделюються за допомогою інструменту Node-red [42]. По-третє, вилучення ознак і криміналістична аналітика, де інструмент Argus [43] використовувався для вилучення ознак даних, а потім статистичні моделі та методи машинного навчання використовувалися для оцінки векторів ознак для розрізнення нормальних і ненормальних випадків.

На рисунку 1.1 наведено візуалізацію тестового стенду який використовувався для створення набору даних.

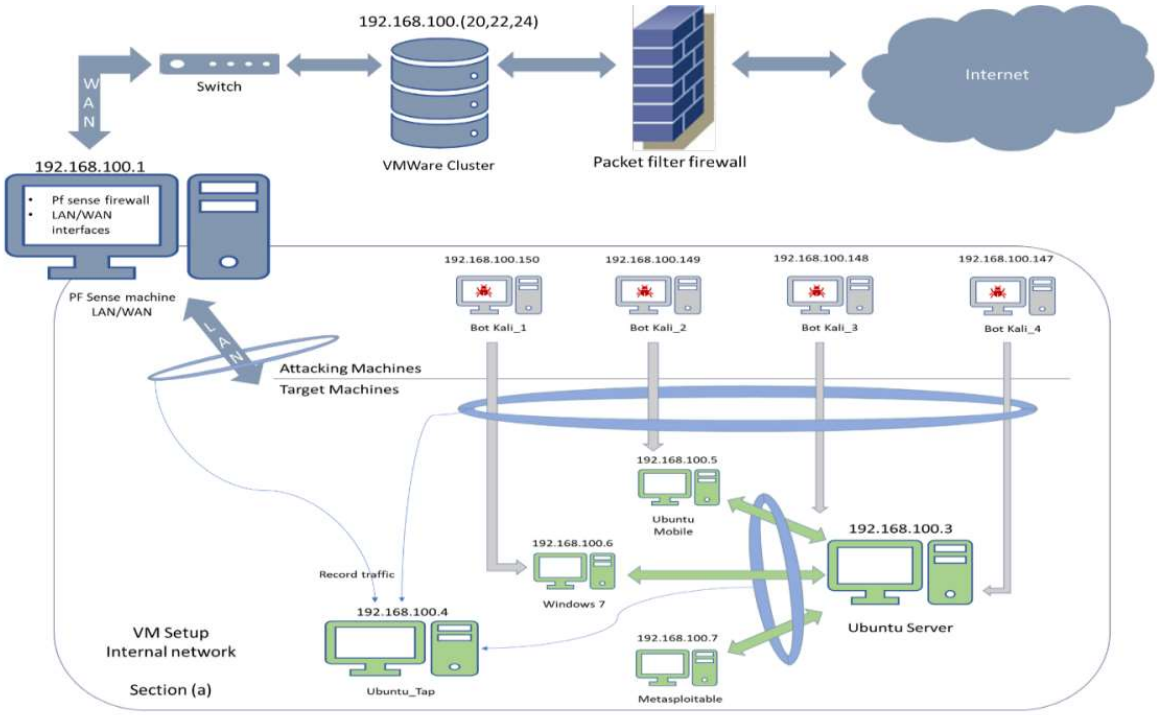

Рисунок 1.1 – Візуалізація тестового середовища

Протокол MQTT [44] використовувався в якості проколу зв'язку, тому що він використовується для комунікації між IOT пристроями (M2M). Він працює в моделі публікації/підписки, ще означає що кожен пристрій при

наявності даних публікує їх брокеру (серверу) MQTT із певним заголовком (темою), тема використовується для групування та організації наявних даних, також клієнти можуть підключатись до сервера та отримувати інформацію із потрібним заголовком. В наборі даних було застосовано такі сценарії Інтернету речей у тестовому стенді:

- Метеостанція, яка генерує інформацію про тиск повітря, вологість і температуру.
- Розумний холодильник, який вимірює температуру холодильника і за необхідності регулює її нижче порогу.
- Індикатори, активовані рухом, які вмикаються або вимикаються на основі псевдовипадкового згенерованого сигналу.
- Дистанційно активовані двері гаража, які відкриваються або закриваються на основі імовірнісного введення.
- Розумний термостат, який регулює температуру в будинку, запускаючи систему кондиціонування.

1.4 Сценарії створення трафіку в наборі даних Bot-IoT

У середовищі тестового стенда розроблено типову конфігурацію розумного дому. Спочатку змодельовано п'ять розумних пристроїв, які працювали локально. Програма Node-Red була використана щоб підключити пристрої інтернету речей та потрібної інфраструктури хмари щоб створити нормальний мережевий трафік. Конфігурація тестового стенду було створено таким чином щоб максимально точно відображати реалістичну мережу розумного дому який використовує п’ять пристроїв інтернету речей які можуть використовуватись в реальних розумних будинках, серед яких розумне освітлення, розумні гаражні двері, розумний холодильний, система моніторингу погоди та розумний термостат. Статистика звичайного трафіку, включена в набір даних, наведена в таблиці 1.1.

Таблиця 1.1 – Статистика нормального трафіку в наборі даних

Протокол	Кількість
UDP	7225
TCP	1750
ARP	468
IPV6-ICMP	88
ICMP	9
IGMP	2
RARP	1
Загалом	9543

Було використано чотири віртуальні машини Kali Linux для паралельного запуску кібератак для реалізації різних сценаріїв ботнету, як показано на рисунку 1.1. Кібератаки та їхні інструменти, розглянуті в наборі даних Bot-IoT:

Зондування [47] — це зловмисна діяльність, яка збирає інформацію про жертви шляхом сканування віддалених систем, також так званий відбиток пальців [48]. Типи зондування, включені в набір даних, обговорюються нижче.

Зондування можна розділити на підкатегорії, першу на основі дій, які виконуються під час зондування, а другу — на основі мети збору інформації. По-перше, за виконуваними діями зондування можна розділити на пасивне та активне [48]. Під час пасивного зондування зловмисник просто захоплює будь-які доступні пакети в мережі, таким чином діючи приховано [48]. З іншого боку, під час активного зондування зловмисник генерує мережевий трафік, націлений на систему, і записує відповіді, порівнюючи їх з відомими відповідями, що дозволяє йому робити висновки про служби та ОС [48]. Що стосується мети дослідження, то існує дві основні підкатегорії: ОС і відбитки пальців служби. У відбитках ОС сканер збирає інформацію про ОС віддаленої

системи, порівнюючи її відповіді з уже існуючими або на основі відмінностей у реалізації стеку TCP/IP. Під час відбитків пальців (сканування) служби сканер визначає служби, які працюють за системними портами (0-65535), надсилаючи пакети запитів [48]. Тут було використано активне сканування, оскільки пасивне сканування створює майже нульовий обсяг згенерованого трафіку.

З теоретичної точки зору, включення безглуздих або надлишкових змінних для опису явища не повинно призвести до погіршення продуктивності класифікаторів. Але на практиці алгоритми навчання за замовчуванням ігнорують основний розподіл даних і зазвичай змушені знаходити рішення шляхом апроксимації задач оптимізації NP-складності через наявність надлишкових функцій [54]. Ця проблема може бути серйозною для класифікаторів на основі нейронних мереж або дерев рішень, які шукають оптимальність.

Інша проблема, пов’язана з надлишком ознак, відома як прокляття розмірності, тобто складність багатовимірного вхідного простору експоненціально зростає з кожною новою змінною, таким чином вектори стають розрідженими та несхожими у величезному всесвіті, а дослідження класифікаторами стає важчим. Ця проблема сильно впливає на деякі методи класифікації, але на кластеризацію та виявлення аномалій загалом [55].

Зниження продуктивності, викликане нерелевантними або надлишковими ознаками, залежить від методу класифікації. Наївний баєсів класифікатор надійний в роботі з нерелевантними змінними, але дуже вразливі до додавання корельованих ознак (навіть якщо вони релевантні) [54]. Навпаки, моделі SVM і kNN сильно погіршуються шумними та нерелевантними змінними. Крім того, деякі лінійні моделі, такі як LAR, дуже чутливі до зашумлених даних, а також корельованих змінних [56].

Тому при виборі набору даних нами було вибрано саме Bot-IoT адже його автори визначили найважливіші ознаки та створили версію набору даних який містить лише їх.

1.5 Аналіз поширених атак

Іноді люди, які мають дозвіл на використання хмарної служби, вирішують пройти інсайдерський шлях. В основному це робиться з наміром використання несанкціонованих привілеїв і розкриття інформації іншим клієнтам або на ринку. Внутрішню атаку планують переважно співробітники конкурентів або адміністратор хмари в компанії клієнта домену, що має право доступу до них. Вони також беруть участь у зміні інформації та документів компанії. Найвідомішим прикладом цієї внутрішньої атаки є Amazon Elastic Compute Cloud (EC2) – внутрішня атака DoS. [4]

У цій атаці зловмисник має на меті не тільки отримати доступ до інформації, але й отримати контроль над даними клієнта. Для цього зловмисник створює власний модуль реалізації сервісу для встановлення його в клієнтську хмарну систему. Для цього використовується метод SaaS/PaaS або екземпляр віртуальної машини в рішенні IaaS. Щоб призвести до зловмисної діяльності, зловмисник, якщо досягне успіху у своїй роботі із забруднення хмари, хмара автоматично прийме та надішле модуль хакера користувачеві. Завдяки чому починається зловмисна діяльність зловмисника.

Типи атак цієї категорії:

- Cross site scripting attack: XSS використовує HTML для атаки, під час якої шкідливий код впроваджується в дані за допомогою Flash, JavaScript або інших.
- Атака SQL-ін’єкції: у цій атаці зловмисник використовує поле введення бази даних користувача. Найпоширенішим прикладом таких атак є атака на веб-сайт Sony play station у 2008 році.

- Command injection attack: назва цієї атаки вказується відповідно до її ролі, оскільки вона впроваджує команду, і ці команди виконуються відповідно до середовища виконання або можуть викликати термінал.

Основна відмінність зловживання та зловмисного використання хмарних служб від інсайдерської атаки – це позиція зловмисників, в іншому все спільне. Під час внутрішньої атаки зловмисник є авторизованим користувачем даних, тоді як у цій атаці зловмисник є хакером, який атакує менш захищену базу даних або погані хмари. Оскільки завдяки цьому не потрібно використовувати дорогий DoS і робити атаки грубої сили на ціль.

Атака «Відмова в обслуговуванні» (DoS) — це атака, що спрямована на виведення з ладу комп'ютера або мережі, щоб вони стали недоступними для користувачів. DoS-атаки досягають цього, надсилаючи великі обсяги даних які система не в змозі обробити або використовуючи вразливість системи надсилають данні які виводять її з ладу. В обох випадках DoS-атака позбавляє законних користувачів (тобто співробітників, членів або власників облікових записів) послуги або ресурсу, які вони очікували.

Жертви DoS-атак часто націлені на веб-сервери відомих організацій, таких як банківські, комерційні та медіа-компанії, або державні та торгові організації. Хоча DoS-атаки зазвичай не призводять до крадіжки або втрати значної інформації чи інших активів, вони можуть коштувати жертві багато часу та грошей.

DoS-атаки мають два основні способи проведення: перенасичення служб трафіком або збій служб. Атаки переповнення трафіком виникають, коли система отримує надто багато трафіку для буферизації сервера, що сповільнює роботу й, зрештою, припиняє роботу. До популярних нападів Flood належать:

Атаки переповнення буфера – найпоширеніша DoS-атака. Концепція полягає в тому, щоб відправити на мережеву адресу більше трафіку, система може обробити. це включає атаки, перелічені нижче, на додаток до інших, які

розроблені для використання помилок, характерних для певних програм або мереж

- ICMP-флуд – використовує неправильно налаштовані мережеві пристрої, надсилаючи підроблені пакети, які виконують ping на кожному комп'ютері цільової мережі, а не на одному конкретному комп'ютері. Потім мережа запускається для посилення трафіку. Ця атака також відома як пінг смерті.
- SYN flood – надсилає запит на підключення до сервера, але ніколи не завершує рукостискання. Триває, доки всі відкриті порти не будуть переповнені запитами, і жоден з них не стане доступним для підключення законних користувачів.

Інші DoS-атаки просто використовують уразливості, які спричиняють збій цільової системи чи служби. Під час цих атак надсилається вхідна інформація, яка використовує помилки в цілі, які згодом призводять до збою або серйозно дестабілізують систему, тому до неї неможливо отримати доступ або використовувати її.

Додатковим типом DoS-атаки є розподілена атака на відмову в обслуговуванні (DDoS). DDoS-атака виникає, коли кілька систем організовують синхронізовану DoS-атаку на одну ціль. Суттєва відмінність полягає в тому, що замість того, щоб атакувати з одного місця, ціль атакується з багатьох місць одночасно. Розподіл хостів, який визначає DDoS, надає зловмиснику численні переваги.

1.6 Алгоритми машинного навчання, які використовуються в інформаційній безпеці

Як випливає з назви, прихована модель Маркова включає ланцюг Маркова (як правило, першого порядку). ПММ також включає набір спостережень, які ймовірнісно пов'язані з цим процесом Маркова. Таким

чином, зі спостережень ми можемо отримати корисну (імовірнісну) інформацію про марковський процес, що лежить в основі.

Кожну з наступних трьох проблем можна ефективно вирішити в рамках ПММ.

- Використовуючи ПММ і послідовність спостережень, ми можемо порівняти послідовність із моделлю.
- Ми можемо розкрити «найкращу» послідовність прихованих станів. Для ПММ «найкращою» визначається послідовність станів, яка максимізує очікувану кількість правильних станів. Навпаки, динамічна програма визначає загальний шлях із найвищими балами.
- Ми можемо навчити модель відповідати даній послідовності спостереження. Метод навчання ПММ можна розглядати як дискретний підйом на пагорб у просторі параметрів (високої розмірності).

Приховані марковські моделі знайшли широке застосування у величезній кількості галузей. У сфері інформаційної безпеки ПММ виявилися корисними для виявлення та аналізу зловмисного програмного забезпечення серед багатьох інших застосувань. Сьогодні найбільш широко використовуваний метод виявлення зловмисного програмного забезпечення заснований на зіставленні шаблонів із заздалегідь визначеними сигнатурами. Деякі просунуті форми зловмисного програмного забезпечення можуть уникнути сканування на основі сигнатур, і ПММ успішно застосовуються в багатьох випадках, коли сканування сигнатур не вдається. ПММ також є основним елементом у дослідженнях систем виявлення вторгнень (IDS). Велика кількість дослідницьких робіт зосереджена на виявленні маскараду на основі команд UNIX. [57]

Випадкові ліси (RF), безумовно, є одними з найпопулярніших методів машинного навчання. Оскільки RF базуються на деревах рішень, основні концепції легко зрозуміти, але деталі дещо заплутані. Дерева рішень надзвичайно прості та потребують небагато даних для побудови. Крім того,

дерева рішень є інтуїтивно зрозумілими та легко поєднуються з іншими методами. Основним недоліком дерева рішень є те, що воно має тенденцію до перенавчання навчальних даних, що зазвичай небажано в алгоритмах навчання.

Щоб проілюструвати дерево рішень, припустімо, що у нас є позначений навчальний набір, що складається зі шкідливих програм і безпечних зразків. З цього навчального набору ми помітили, що зразки зловмисного програмного забезпечення, як правило, менші за розміром і мають вищу ентропію порівняно з доброякісними зразками. Ми могли б використати цю інформацію для побудови дерева рішень, де порогові значення для «великого» проти «малого» (розмір) і «високого» проти «низького» (ентропія) базувалися б на даних навчання. Потім це дерево рішень можна використовувати для класифікації будь-якого зразка як зловмисного або доброякісного програмного забезпечення на основі його розміру та ентропії.

Інформаційний приріст, наданий ознакою, можна визначити як очікуване зменшення ентропії, коли ми розгалужуємося на цій конкретній ознаці. У контексті дерева прийняття рішень приріст інформації можна обчислити як ентропію батьківського вузла мінус середньозважену ентропію його дочірніх вузлів. При побудові дерева рішень ми можемо виміряти приріст інформації для кожної ознаки та вибирати ознаки жадібним чином на основі цього показника. Таким чином, ознаки з найбільшим посиленням будуть найближче до кореня. Це бажано, оскільки ми будемо зменшувати ентропію якомога швидше. Додаткова перевага такого підходу полягає в тому, що він дає нам змогу спростити дерево шляхом урізання ознак, які забезпечують незначний приріст інформації або взагалі його не надають.

Випадкові ліси широко застосовувалися для вирішення проблем інформаційної безпеки. Залучення RF для інформаційної безпеки включають виявлення вторгнень, виявлення зловмисного програмного забезпечення і

шкідливих PDF-файлів, виявлення фішингу, розпізнавання облич і виявлення вразливого програмного забезпечення та інші. [57]

Як і багато інших методів машинного навчання, SVM є концептуально простим, але отримати глибоке розуміння може бути напрочуд важко. З точки зору високого рівня, SVM просто відокремлює позначені навчальні зразки за допомогою гіперплощини (якщо це можливо), водночас максимізуючи «відступ», тобто відстань від гіперплощини до класів. У SVM дані, як правило, відображаються у просторі з більшою вимірністю, що полегшує розділення за допомогою гіперплощини. Так званий «трюк ядра» дозволяє нам працювати у просторі з більшою вимірністю, не отримуючи суттєвої втрати продуктивності. У цьому сенсі SVM є цікавим контрастом до PCA, де головним завданням є зменшення розмірності. Перевірку деталей трюку ядра можна здійснити за допомогою множників Лагранжа. Хоча це не просто, це варте зусиль, оскільки хитрість ядра є ключем до розуміння SVM.

SVM довели свою ефективність у сфері виявлення шкідливих програм. Інші програми безпеки, де SVM успішно використовувалися, включають виявлення та аналіз спаму зображень, виявлення вторгнень та аналіз спаму (на основі тексту). Аналіз мережевих атак, стеганографічний аналіз і біометрична ідентифікація є додатковими прикладами численних застосувань SVM, пов’язаних із безпекою. З іншого боку, через геометричну природу процесу навчання SVM цілеспрямована атака, заснована на забрудненні навчальних даних відносно проста, що можна вважати слабкою стороною в порівнянні з іншими техніками машинного навчання, особливо в програмах безпеки. [57]

РОЗДІЛ 2. ПІДХОДИ ДО СТВОРЕННЯ СИСТЕМИ ВИЯВЛЕННЯ ВТОРГНЕНЬ

2.1 Системи виявлення вторгнень

Вторгнення можна визначити як будь-який вид несанкціонованої діяльності, яка завдає шкоди інформаційній системі. Тобто будь яка загроза для цілісності, доступності або конфіденційності інформації може вважатись вторгненням. Наприклад якщо в результаті атаки на відмову в обслуговуванні сервер не відповідає користувачам це можна класифікувати як вторгнення. Система виявлення вторгнень (IDS) це система яка може бути реалізована як програмними так і апаратними методами для виявлення зловмисних дій в комп'ютерних мережах і цим забезпечує безпеку [5]. Така система використовується з метою виявлення зловмисного мережевого трафіку чи поведінки комп'ютера що не може бути ідентифіковано як вторгнення традиційним брандмауером. Використання такого роду систем є надзвичайно важливим для підвищення рівня захисту від зловмисних дій спрямованих на порушення цілісності, доступності або конфіденційності комп'ютерних мереж. За принципом роботи системи виявлення вторгнень можна поділити на системи на основі аномалій та на основі сигнатур.

2.2 Системи що використовують сигнатури для виявлення вторгнень

Система виявлення вторгнень на основі сигнатур (SIDS) або системи виявлення вторгнень на основі знань (Knowledge-based Detection) [6], вони використовуючи відомі шаблони атак зіставляють їх із наявними даними для виявлення вторгнень. Таким чином тривога спрацьовує якщо сигнатура вторгнення присутня в базі даних. Також система виявлення вторгнень на основі сигнатур може виявляти зловмисне програмне забезпечення

перевіряючи історичні дані журналів хоста в пошуку послідовностей дій чи команд які визначені як програмне забезпечення.

Рисунок 2.1 демонструє схематичну ідею систем виявлення вторгнень на основі сигнатур. Підхід полягає в створені бази даних сигнатур різних вторгнень в якій потім буде виконуватись пошук поточного набору дій, і якщо такий набір дій присутній в базі даних подається тривога. Можна сформулювати це правилом «якщо: причина тоді: наслідок» може призвести до «якщо (IP-адреса джерела == IP-адреса призначення), тоді позначити як атаку».

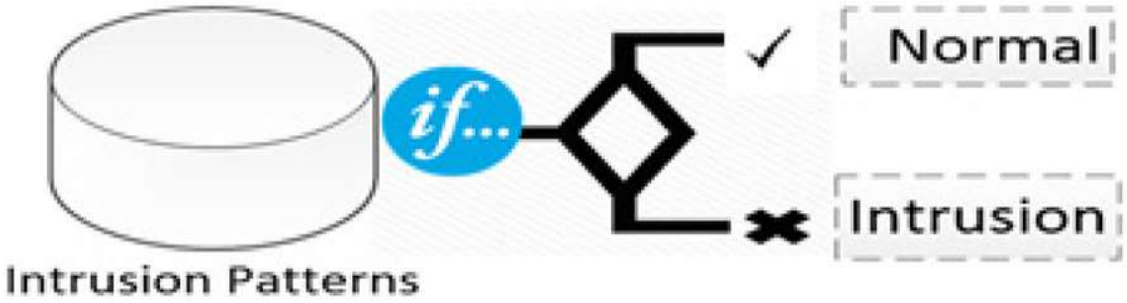

Рисунок 2.1 – Візуалізація роботи системи виявлення вторгнень на основі сигнатур

Такий підхід до виявлення вторгнень на практиці має високу точність виявлення вторгнень які присутні в базі даних сигнатур [7]. Але варто також зазначити що системи виявлення вторгнень на основі знань мають проблеми з виявленням нових атак, наприклад атак нульового дня, так як їх сигнатура відсутня у базі даних, вторгнення буде залишатись непоміченим поки сигнатура не буде добута та додана в базу даних. Також системи виявлення вторгнень на основі сигнатур використовуються в багатьох поширених інструментах, наприклад, Snort [8] і NetSTAT [9].

Іншим недоліком систем виявлення вторгнень на основі сигнатур є те що вона зіставляє мережеві пакети із базою сигнатур для їх перевірки. Однак такий підхід не здатний визначити вторгнення яке може складатись із

декількох пакетів, а так як нове зловмисне програмне забезпечення часто є більш складним ніж раніше то для виявлення вторгнень може знадобитись витягти інформацію про сигнатури із декількох пакетів. Для цього система виявлення вторгнень повинна проаналізувати попередні пакети. Для створення нових сигнатур існує низка методів, у яких сигнатури створювалися як кінцеві машини [10], шаблони формальних мовних рядків або семантичні умови [11].

Так як кількість атак нульового дня зростає [1] то системи виявлення вторгнень на основі сигнатур втрачають ефективність у зв'язку із вразливістю перед атаками які відбуваються вперше або настільки нові що ще не додані в бази даних сигнатур. Зловмисне програмне забезпечення що використовує послідовність пакетів та зростання кількості атак що є добре спланованими та мають визначену ціль ще більше погіршують становище систем на основі знань. Можливою панацеєю цих проблем можуть бути системи виявлення вторгнень на основі аномалій, вони визначають нормальний трафік, і усе що від нього відхиляється вважається аномалією – вторгненням.

2.3 Система виявлення вторгнень на основі аномалій

Системи виявлення вторгнень на основі аномалій зацікавили багатьох, адже вони потенційно не є вразливими до проблем систем виявлення вторгнень на основі знань. Системи на основі аномалій моделюють нормальну поведінку у системі використовуючи штучний інтелект, часто використовується саме машинне навчання, статистичні методи або методи на основі знань. Якщо система спостерігає різницю поведінки та нормальної моделі то це визначається як аномалія, аномалії вважаються вторгненнями. Для роботи цієї моделі потрібно щоб зловмисна поведінка мала значні відмінності від нормальної поведінки. Для створення системи виявлення вторгнень на основі аномалій існує два етапи, етап тренування та тестування

системи. В процесі навчання модель визначає нормальну поведінку в мережі, тоді як в етапі тестування модель перевіряється на наборі даних яких вона раніше не бачила, таким чином перевіряється здатність моделі до узагальнення. Системи виявлення вторгнень на основі аномалій на основі методу для навчання моделі можна поділити на статистичні, методи машинного навчання або методи на основі знань [12].

Так як система виявлення вторгнень на основі аномалій не використовує базу даних сигнатур вона теоретично може визначати атаки нульового дня [13]. Тривога в системі виявлення вторгнень на основі аномалій спрацьовує якщо поведінка яку спостерігає система не схожа на нормальну поведінку вивчену системою. Також варто зазначити що система виявлення вторгнень на основі аномалій має деякі інші переваги, наприклад якщо зловмисник отримавши доступ до облікового запису користувача починає виконувати дії які незвичні для цього користувача то вони розпізнаються як вторгнення, а зловмиснику у свою чергу важко визначити нормальну поведінку користувача, тому що в процесі визначення нормальної поведінки він швидше за все викличе тривогу.

У таблиці 2.1 наведено порівняння систем виявлення вторгнень на основі сигнатур і на основі аномалій. Система виявлення вторгнень на основі сигнатур може ідентифікувати лише добре відомі вторгнення, тоді як система виявлення вторгнень на основі аномалій може виявити атаки нульового дня. Однак система виявлення вторгнень на основі аномалій може призвести до високого рівня хибних позитивних результатів, оскільки аномалії можуть бути просто новою нормальною діяльністю, а не справжнім вторгненням.

Таблиця 2.1 – Порівняння систем виявлення вторгнень на основі сигнатур та на основі аномалій

Метод виявлення	Переваги	Недоліки
Система виявлення вторгнень на основі сигнатур	Дуже ефективна у виявленні вторгнень з мінімальною кількістю помилкових тривог. Швидко визначає вторгнення. Краще для виявлення відомих атак. Простий дизайн	Необхідно часто оновлювати нові сигнатури. SIDS призначений для виявлення атак за відомими сигнатурами. Якщо попереднє вторгнення було дещо змінено на новий варіант, тоді система не зможе ідентифікувати це нове відхилення подібної атаки. Не вдається виявити атаку нульового дня. Не підходить для виявлення багатоетапних атак. Мале розуміння проникливості атак
Система виявлення вторгнень на основі аномалій	Може використовуватися для виявлення нових атак. Можна використовувати для створення сигнатури вторгнення	AIDS не може обробляти зашифровані пакети, тому атака може залишитися непоміченою та представляти загрозу. Висока кількість хибних спрацьовувань. Важко створити нормальний профіль для дуже динамічної комп'ютерної системи. Некласифіковані оповіщення. Потребує навчання.

2.4 Типи систем виявлення вторгнень на основі вхідних даних

Попередні розділи класифікували системи виявлення вторгнень на основі методів, які використовуються для ідентифікації вторгнень. Системи виявлення вторгнень можуть бути класифіковані і використовуючи джерело даних. Класифікуючи системи вторгнення на основі джерела даних можна виділити дві основі технології: виявлення вторгнень використовуючи данні хоста і виявлення вторгнень використовуючи дані мережевого трафіку.

Системи виявлення вторгнень що використовують дані мережевого трафіку отримують дані використовуючи захоплення мережевих пакетів, потоків та ін. Його перевагою є те що якщо система перевіряє дані мережі то вона може виявити вторгнення на будь-якому з її компонентів що підключені до мережі. До того ж якщо система перевіряє трафік що входить у мережу вона може заблокувати його до того як він потрапить в мережу і атака розповсюдиться на всю мережу. Але звісно такий варіант має і свої недоліки, як наприклад така система має працювати достатньо швидко або мати достатні потужності щоб перевіряти трафік в мережах із високою пропускною здатністю [15]. Комбінація системи виявлення вторгнень що використовує дані мережевого трафіку та системи що використовує дані хоста може забезпечити багаторівневий захист від зовнішніх і внутрішніх атак.

Система виявлення вторгнень що використовує дані хоста отримує дані які доступні системі хоста і які збираються за допомогою аудиту, наприклад це журнали операційної системи, серверів, баз даних, брандмауерів та додатків. Перевагою такої системи є те що вона потенційно може виявити атаки зсередини що проводяться без використання мережі [14].

Далі наведено порівняння HIDS та NIDS.

HIDS:

- Переваги:
    - HIDS може перевіряти наскрізне шифрування зв'язку.
    - Не вимагає додаткового обладнання.
    - Виявляє вторгнення, перевіряючи файлову систему хоста, системні виклики або мережеві події.
    - Кожен пакет збирається повторно.
    - Переглядає весь елемент, а не лише потоки
- Недоліки:
    - Затримки у звітах про атаки
    - Споживає ресурси хоста

- Потрібно встановити на кожному хості.
- Він може відстежувати атаки лише на машині, де його встановлено.

- Джерела даних:
  - Записи аудиту, файли журналів, інтерфейс прикладної програми (API), шаблони правил, системні виклики.

NIDS:

- Переваги:
  - Виявляє атаки шляхом перевірки мережевих пакетів.
  - Не потрібно встановлювати на кожному хості.
  - Можна перевіряти різні хости одночасно.
  - Здатність виявляти найширші діапазони мережевих протоколів
- Недоліки:
  - Завдання полягає в ідентифікації атак із зашифрованого трафіку.
  - Потрібне спеціальне обладнання.
  - Підтримує лише ідентифікацію мережевих атак.
  - Важко аналізувати високошвидкісну мережу.
  - Найсерйознішою загрозою є внутрішня атака.
- Джерела даних:
  - Простий протокол керування мережею (SNMP)
  - Мережеві пакети (TCP/UDP/ICMP),
  - База інформації про управління (MIB)
  - Записи маршрутизатора NetFlow

2.5 Техніки імплементації систем виявлення вторгнень на основі аномалій

Методи реалізації систем виявлення вторгнень можна поділити на системи що використовують статистику [17], знання [18] і машине навчання

[19]. Підхід, заснований на статистиці, передбачає створення статистичної моделі нормальної поведінки використовуючи усі доступні дані із набору даних. З іншого боку, метод заснований на знаннях використовуючи наявні дані визначає необхідні протоколи та інші характеристики екземплярів мережевого трафіку щоб визначити нормальний трафік. Методи машинного навчання в залежності від алгоритму що використовується створюють шаблон нормального трафіку і трафіку атаки використовуючи характеристики навчальних даних. Наведені системи показані на рис. 2.2.

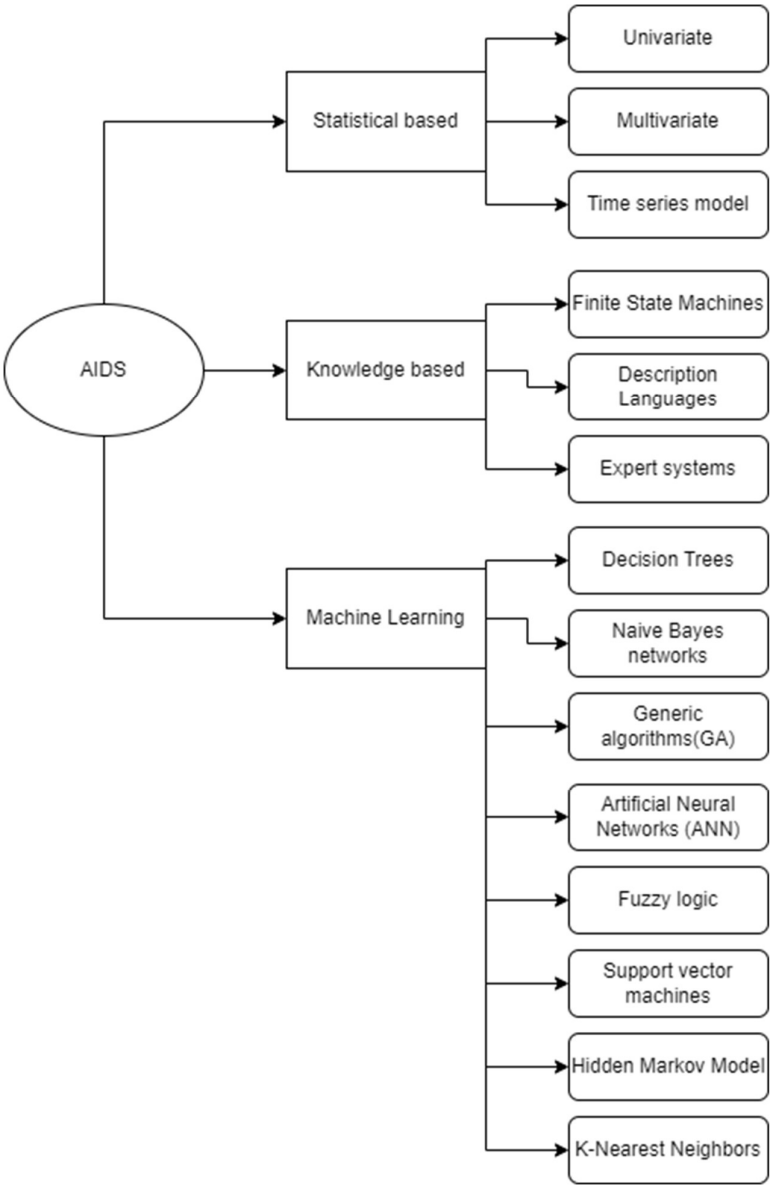

Рисунок 2.2 – Класи систем виявлення вторгнень на основі аномалій

2.5.1 Техніки що базуються на використанні статистики

Системи виявлення вторгнень що базуються на статистиці створюють модель розподілу нормальної поведінки, під час перевірки визначається ймовірність виникнення такої події і якщо вона достатньо низька викликається тривога. Також така система може враховувати статистичні показники трафіку, наприклад середнє значення кількості надісланих пакетів від джерела, моди протоколів, стандартне відхилення та інші. Системи виявлення вторгнень що базуються на основі статистики можуть виявити будь-яку відмінність поведінки що перевіряється від нормальної. Як правило використовується одна із наведених моделей:

- Univariate: Така модель відстежує зміни лише однієї характеристики трафіку, таким чином зрозуіло що і нормальний трафік в такому випадку визначається лише однією характеристикою. Однофакторна система виявлення аномалій шукає аномалії в кожному окремому показнику [15].
- Multivariate: Така модель відстежує зміни в декількох характеристиках трафіку та розуміє відношення між характеристиками.
- Модель часових рядів (time series): Така модель відстежує зміни протягом певного інтервалу часу і визначає імовірність виникнення події в цей момент часу.

2.5.2 Техніки що базуються на знаннях

Системи виявлення вторгнень що базуються на знаннях для роботи використовують попередньо створену базу знань, що містить нормальну поведінку в мережі. Якщо подія що перевіряється не схожа на нормальну поведінку вона вважається вторгненням. Така система потребує людини яка визначить нормальну поведінку системи використовуючи певні правила.

Так як система має визначеними всі можливі профілі нормальної поведінки вона має набагато менше хибних тривог, але так як середовище

роботи систем для виявлення вторгнень зазвичай часто якщо не постійно змінюється і тому така система повинна часто обновлятись людиною, яка повинна зібрати інформацію про всі нові нормальні поведінки, і це не є тривіальною задачею.

Скінченний автомат (Finite state machine): Така модель зазвичай є абстракцією що представляється у вигляді станів, дій і переходів. Для прикладу перехід може відбуватися на основі виявленої послідовності вхідних даних [20]. Така модель може визначити нормальну поведінку системи і якщо виявить відмінність від такої поведінки вона викличе тривогу.

Description Language: Така модель встановлює синтаксис правил що використовуються для встановлення ознак нормального трафіку чи атаки. Часто такі правила встановлюються використовуючи N-грами та UML [21].

Експертна система: Така модель використовує певні правила що встановлюють нормальний трафік та атаки, як правило в такий системі правила встановлюються людиною або групою людей що мають навички для створення таких правил та знання предметної області [22].

Аналіз сигнатур: Така модель є першою технологією для систем виявлення вторгнень, в ній під час перевірки пакета його сигнатура співставляється із сигнатурами атак і якщо є збіг викликається тривога [23].

2.5.3 Техніки що базуються на машинному навчанні

Машинне навчання — це процес створення моделі яка визначає закономірності в даних. Така модель зазвичай складається із математичних відношень, наборів правил чи функцій. Після тренування моделі, іншими словами пошуку закономірностей в даних така модель може використовуватись для створення передбачень на даних які вона ніколи не бачила [24].

Існує великий перелік методів машинного навчання які використовуються для систем виявлення аномалій на основі аномалій, серед

них метод найближчих сусідів, кластеризація, дерева рішень та ліси дерев рішень та інші [25].

На рисунку 2.3 схематично показано роботу методів машинного навчання в системах виявлення вторгнень на основі аномалій. Використовуючи машинне навчання для систем виявлення вторгнень можна покращити точність та зменшити потребу в експертних знаннях, адже алгоритми машинного навчання самі визначають закономірності в даних. Тому останні роки кількість систем виявлення вторгнень що використовують машинне навчання збільшується

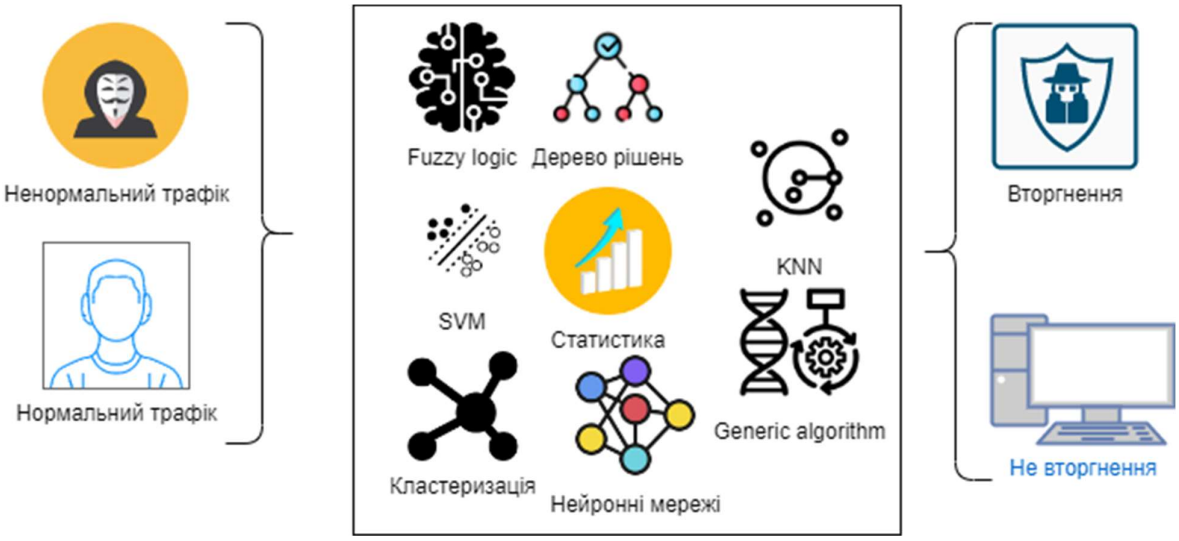

Рисунок 2.3 – Концепт роботи системи виявлення вторгнень на основі аномалій що базується на машинному навчанні

Загалом існує два типи методів машинного навчання: з учителем та без.

Методи машинного навчання з учителем використовують позначені данні для тренування та тестування моделі. На кроці тренування модель отримує дані які містять ознаки та мітку класу до якого належить кожен запис, таким чином модель визначає закономірності належності певних ознак до певного класу, на кроці тестування модель намагається визначити мітку класу для даних які вона не бачила раніше і порівнюючи справжні і передбачені

мітки класу на тестувальному наборі даних ми можемо визначити здатність моделі визначати класи. У випадку систем виявлення вторгнення міткою класу буде чи належить запис до нормального трафіку чи до атаки.

Для систем виявлення вторгнень використовуються методи класифікації, серед яких є нейронні мережі, метод опорних векторів, дерева рішень та інші. Кожен алгоритм використовує іншій підхід для виявлення закономірностей. Можливість натренованої моделі показувати хороші результати не лише на тренувальному а й на тестувальному наборі даних називають здатністю до узагальнення і вона також є важливою характеристикою моделі.

Важко уявити простіший метод, ніж k-NN, де дані класифікуються просто на основі найближчого сусіда (або сусідів) у даному навчальному наборі [33]. Для k-NN немає фази явного навчання, оскільки після того, як навчальний набір визначено, більше нічого не потрібно робити для навчання моделі. Метод k-NN використовувався в таких різноманітних програмах безпеки, як виявлення вторгнень, біометрична автентифікація на основі динаміки натискання клавіш і захист запитів у хмарі [57]. На рисунку 2.4 показано класифікатор K-найближчих сусідів, де $k = 5$. Точка в центрі представляє екземпляр непозначених даних, яку потрібно класифікувати. Серед п'яти найближчих сусідів є три схожі шаблони з класу Intrusion і два з класу Normal. Більшість голосів дозволяє призначити цю точку до класу вторгнення.

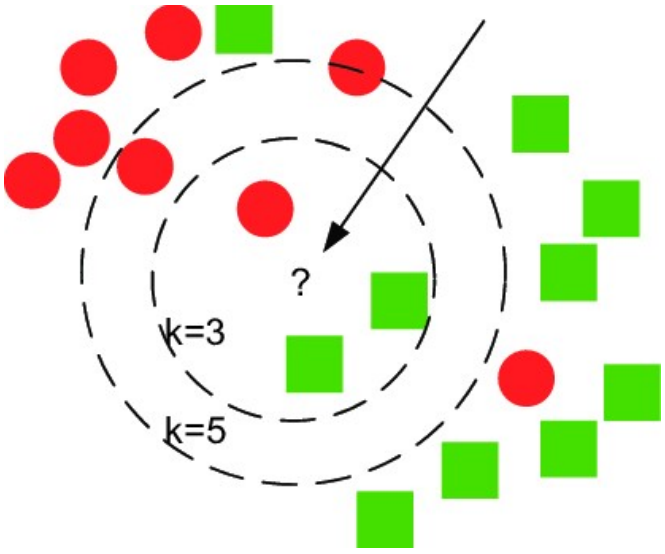

Рисунок 2.4 – Приклад роботи KNN

Модель дерева рішень складається із декількох компонентів, а саме “листя”, “гілки” та вузли прийняття рішень, вузол ідентифікує атрибут від якого залежить цільова функція, тоді як гілка вказує на наступний вузол на основі значення атрибуту, листок вказує на клас бо якого належить запис [26].

Наївний баєсів класифікатор базується на теоремі Байєса щоб визначити ймовірність належності елемента який перевіряється до наявних класів при припущенні що змінні незалежні. Така модель в системах виявлення вторгнень визначає ймовірність того що відбувається атака, або навіть ймовірність якої саме атаки найбільша. Наївний баєсів класифікатор часто використовується в системах виявлення вторгнень завдяки простоті використання а достатньо високій точності [27]. Також варто зазначити що така модель погано працює якщо змінні не є незалежними, приклад такого випадку це набір даних KDD99, що містить складні залежності характеристик. Також варто зазначити що зазвичай наївний баєсів класифікатор менш ефективний для великих наборів даних. Деякі проблеми наївного баєсового класифікатора вирішує складніша варіація алгоритму - Прихований баєсів класифікатор, який також застосовується для завдань систем виявлення вторгнень, може обробляти

велику кількість даних в швидких мережах, не погіршується в роботі із великою кількістю характеристик та взаємозвязків між атрибутами [28].

Генетичні алгоритми намагаються оптимізувати модель використовуючи симуляцію еволюції. Усі параметри рішення представляється у вигляді генів, таким чином симулюючи еволюцію з часом якість рішень покращується, для цього використовуються способи відбору та відтворення [29].

Штучні нейронні мережі це модель що складається із ваг та нейронів, та навчається за допомогою алгоритму зворотного поширення помилки який визначає градієнт помилки нейронної мережі. Такі моделі є доволі поширеним методом що використовується для виявлення шкідливих програм хоча і має деякі проблеми для використання в системах виявлення вторгнень, так як наприклад нейронні мережі часто мають проблеми із виявленням рідкісних атак, адже їх прикладів часто недостатньо для тренування такої моделі, така вразливість може завдати серйозної шкоди якщо атаки з меншою частотою не будуть виявлені, як наприклад атака User to Root буде успішною то зловмисник отримає доступ до користувача адміністратора, що дозволить подальші атаки, також це важливо тому що рідкісні атаки частіше мають змогу вийти за межі комп'ютерної мережі в якій вони почались [30]. Штучні нейронні мережі можуть бути вразливими до локальних мінімумів, це означає що тренування може тривати досить довго. Серед переваг нейронних мереж те що деякі архітектури нейронних мереж можуть визначати нелінійні зв'язки між вхідними ознаками запису та класами які потрібно визначити, тому нейронні мережі є потужним інструментом який може використовуватись в мережах виявлення вторгнень.

Нечітка логіка (Fuzzy logic): це модель що використовує замість типової істинній чи хибній логіці степені невизначеності, вона використовувалась для створення сучасних комп'ютерів. Простіше кажучи така модель дає змогу отримати висновок використовуючи нечіткі, шумні, неоднозначні, відсутні

або неточні вхідні дані. Така модель дозволяє кожному запису частково належати декільком класам одночасно, така властивість моделі може дозволити їй бути хорошим вибором для проблем виявлення вторгнень, адже і сама безпека включає невизначеність, також і різниця між нормальною поведінкою і ненормальною часто невизначена. До того ж, характеристиками для виявлення вторгнень часто є різні числові показники які добуто із даних, а також похідні статистичні показники. А так як використання жорстких порогів для числових даних в системі виявлення вторгнень часто може призвести до породження хибних тривог то модель нечіткої логіки може мати змогу зменшити кількість помилкових тривог [31]. Також така модель може розпізнати діяльність яка лише трішки відрізняється від нормальної поведінки.

Модель Маркова із прихованим профілем спочатку була розроблена для застосування в біоінформатиці, але застосовувалися ширше, хоча й не так широко, як багато інших методів машинного навчання. Оскільки Прихована модель Маркова є статистичниною за своєю природою, випадкові відсутні або сторонні спостереження не мають особливого значення. Навпаки, будь-який відсутній або сторонній елемент у моделі Маркова із прихованим профілем призведе до зміщення. Отже, ми повинні чітко враховувати вставки та видалення в межах моделі Маркова із прихованим профілем. Отже, переходи станів моделі Маркова із прихованим профілем набагато складніші, ніж переходи прихованої моделі Маркова. Навчання моделі Маркова із прихованим профілем складається з двох кроків: спочатку ми генеруємо множинне вирівнювання послідовностей, вирівнюючи колекцію навчальних послідовностей. Потім, на основі цього множинного вирівнювання послідовностей, ми, по суті, визначаємо особливо просту приховану модель Маркова у кожній позиції в межах множинного вирівнювання послідовностей. Отриманий моделі Маркова із прихованим профілем можна використовувати для підрахунку балів. Така система може використовуватись для виявлення деяких типів шкідливого програмного забезпечення [32].

Навчання без учителя або кластеризація це методологія машинного навчання що дозволяє виявляти шаблони в даних які не містять попередніх міток класу. У кластеризації наша мета полягає в тому, щоб розділити набір даних на підмножини (як правило, непересічні), де всі дані в підмножині мають подібні характеристики. Кластеризація зазвичай застосовується в режимі «дослідження даних», тобто ми використовуємо кластеризацію, щоб спробувати отримати деяке розуміння таємничих даних. Найпопулярнішим методом кластеризації є K-means, який базується на надзвичайно простому ітераційному процесі, де ми чергуємо між обчисленням центроїдів кластера (тобто центру маси кожного кластера) і призначенням точок даних кластерам. Кластеризація за максимізацією очікувань (EM) виконується за подібною процедурою, але використовує ймовірнісні розподіли для міри «відстані» [57].

На рисунку 2.5 показано приклад кластеризації, в якому було створено 2 кластери, якщо провести аналогію із система виявлення вторгнень це можуть бути записи що було поділено на два типи поведінки які прийнято вважати нормальними адже нормальні випадки повинні створювати великі кластери, також на рисунку можна побачити три записи які не потрапили в жоден кластер, такі записи вважаються викидами і є швидше за все атаками.

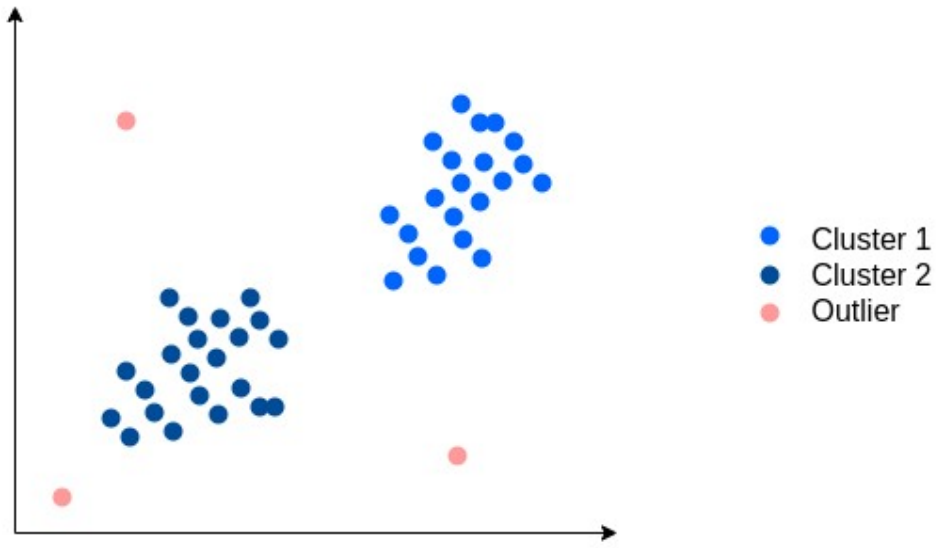

Рисунок 2.5 – Приклад кластеризації даних

Кластеризація використовувалася у великій кількості досліджень, спрямованих на виявлення шкідливих програм, аналіз шкідливих програм і класифікацію шкідливих програм. Різні форми кластеризації застосовувалися для широкого спектру інших проблем інформаційної безпеки, включаючи аналіз спаму і мережеві атакувальні дії. Крім того, кластерний аналіз виявився корисним у виявленні вторгнень, виявленні ботнет-трафіку, а також у вирішенні різних проблем конфіденційності, та серед багатьох інших програм [57]. Також виявлення і реактивне пом'якшення атак за допомогою навчання без учителя широко досліджено в області систем кіберфізичного контролю [34].

Напівавтоматичне навчання знаходиться між навчанням з учителем (з повністю позначеними мітками класів у тренувальних даних) і навчанням без учителя (без будь-яких класифікованих навчальних даних). Напівавтоматичне навчання можна використовувати в поєднанні з продуктивністю класифікатора невеликої кількості мічених даних для систем виявлення вторгнень для зменшення витрат часу та грошей. Такий підхід є достатньо важливим адже у сфері виявлення вторгнень існує мала кількість загальнодоступних даних а реальні дані досліджуваної системи можуть бути досить рідкісними [35].

Було запропоновано низку різних методів напівавтоматичного навчання, наприклад алгоритми що використовують напівавтоматичне SVM, максимізацію очікувань, методи на основі графів, самонавчання, спів-навчання, та методи напівавтоматичного навчання на основі посилення.

Деякі підходи перенавчають модель із поступовим додаванням кожної нової категорії до початкового тренувального набору даних. Експериментальні результати з використанням цього напівавтоматичного виявлення вторгнень на наборі даних NSL-KDD показують, що можливо

значною мірою підвищити точність системи виявлення вторгнень на відміну від традиційних підходів до навчання. [36]

Для підвищення точності моделі можна також об’єднати її у ансамбль із декількома іншими моделями, такий підхід теоретично повинен мати кращу точність, існує декілька способів створення ансамблю: підсилення, беггінг та пакінг.

Підсилення — це загальна техніка, за допомогою якої ми об’єднуємо кілька (слабких) класифікаторів в один (набагато сильніший) класифікатор [57]. Багато методів машинного навчання можна застосувати у спосіб, подібний до посилення.

Для використання пакінгу потрібно навчати одну і ту ж модель але використовуючи різні підмножини набору даних.

Беггінг поєднує різні класифікації за допомогою метакласифікатора [37]. Моделі базового рівня будуються на основі цілого навчального набору, потім мета-модель навчається на виходах моделі базового рівня як атрибутах.

2.6 Гібридні техніки в системі виявлення вторгнень

Розповсюджені системи виявлення вторгнень мають декілька обмежень: їх неможливо легко модифікувати, нездатність ідентифікувати нові зловмисні атаки, низька точність і висока кількість помилкових тривог. Де системи виявлення вторгнень на основі аномалій має обмеження, такі як високий рівень хибних позитивних результатів гібридна система виявлення вторгнень заснована на поєднанні систем на основі сигнатур та на основі аномалій долає недоліки систем на основі сигнатур та на основі аномалій. В [38] запропоновано гібридну систему виявлення вторгнень за допомогою наївного Баєсівського класифікатора та дерева рішень і досягли рівня виявлення 99,63% на наборі даних KDD’99.

РОЗДІЛ 3. РОЗРОБКА СИСТЕМИ ВИЯВЛЕННЯ МЕРЕЖЕВИХ АТАК ТА ЇЇ РЕЗУЛЬТАТИ

3.1 Попередня обробка тренувального набору даних

Першим кроком у використанні методів машинного навчання є попередня обробка даних, яка дозволить покращити результати для деяких алгоритмів та потрібна для перетворення даних у потрібний формат.

Набір даних Bot-IoT містить дані у трьох форматах:

- .pcap – записаний мережевий трафік
- .argus – файли створенні програмою Argus Project, що містять замість індивідуальних пакетів інформацію про статус потоків даних в мережі. [45]
- .csv – файловий формат, який використовується для зберігання табличних даних, в ньому колонки розділяються комами а рядки символом переходу на новий рядок. [46]

Для найбільшої зручності було використано формат даних csv, так як він містить усі дані та є більш зручним для обробки мовою програмування Python. Також дані у csv форматі були попередньо оброблені при створенні датасету. Зважаючи на те, що згенерований набір даних дуже великий (понад 72 000 000 записів і 16,7 ГБ для CSV, з 69,3 ГБ pcap), це ускладнило обробку даних. Таким чином, було вилучено 5% вихідного набору даних за допомогою запитів MySQL. Витягнуті 5%, які ми будемо називати наборами для навчання та тестування, складаються з 4 файлів загальним розміром приблизно 0,78 ГБ і близько 3 мільйонів записів.

Крім того, через наявність певних протоколів (ARP) значення номера порту джерела та призначення були відсутні, ці значення було встановлено на -1, що є недійсним номером порту, знову ж таки з метою оцінки набору даних. Було перетворено категоричні значення ознак у наборі даних у послідовні

числові значення для легкого застосування статистичних методів. Наприклад, атрибут стану має деякі категоричні значення, такі як «RST», «CON» і «REQ», які були зіставлені в «1», «2» і «3».

Крім того, нормалізацію було застосовано для масштабування даних у певному діапазоні, наприклад [0,1], без зміни нормальної поведінки даних. Цей крок допомагає статистичним моделям і методам машинного навчання краще тренуватись, вирішуючи локальні оптимальні проблеми. Було виконано перетворення Min-Max для даних.

Пряме порівняння показників ентропії та кореляції було використано щоб зменшити кількість ознак в наборі даних. Ознака вважатиметься ідеальною для набору даних, якщо її оцінка ентропії достатньо висока, а оцінка кореляції досить низька. Це означало б, що ця функція не містить надлишкової інформації, яка використовується спільно з іншими функціями, і що вони якомога не пов’язані одна з одною. Щоб порівняти середні значення цих різних статистичних показників, значення балів були нормалізовані в діапазоні [0,1]. Враховуючи те, що вищі значення коефіцієнта кореляції вказують на висококорельовані характеристики, які повинні бути видалені з набору даних, після виконання перетворення Min-Max було інвертовано результати, щоб привести середні бали крос-ентропії у той самий формат, що й оцінка спільної ентропії (де вищі значення означає більшу випадковість між функціями). Було порівняно нові зіставлені значення коефіцієнта кореляції та спільної ентропії, щоб виділити підмножину з 10 ознак, які, загалом, мали найкращі оцінки в обох статистичних показниках. Таким чином, було визначено наступні 10 найкращих ознак: srate, drate, rate, max, state number, mean, min, stddev, flgs number, seq. [41]

Також кінцевий набір даних було розділено на тренувальний та тестувальний набір даних, на рисунку 3.1 наведено розподіл категорій трафіку в тренувальному наборі та на рисунку 3.2 наведено розподіл категорій трафіку в тренувальному наборі даних.

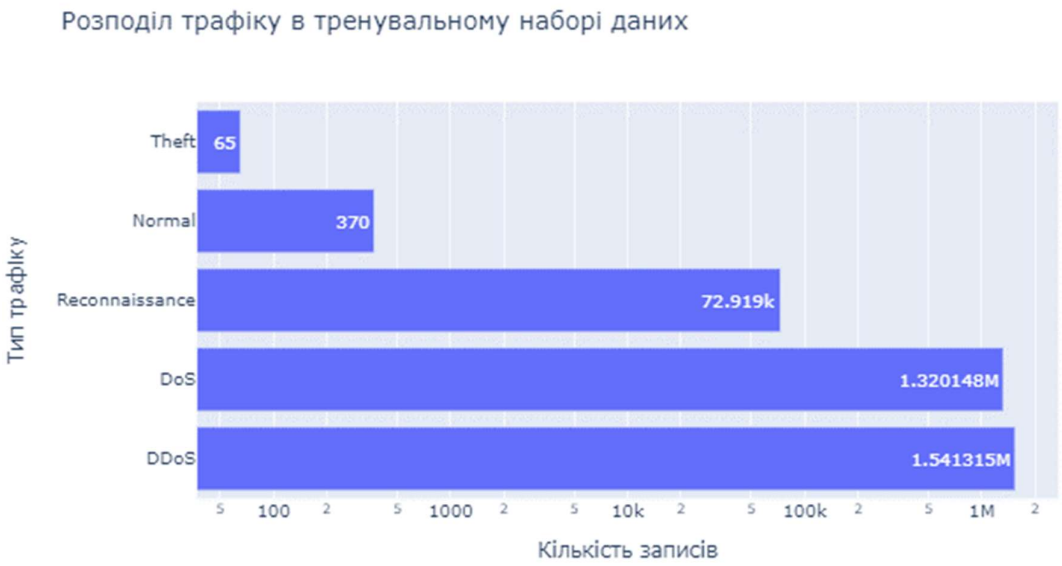

Рисунок 3.1 – Розподіл трафіку в тренувальному наборі даних

Рисунок 3.2 – Розподіл трафіку в тестувальному наборі даних

Так як ми будуємо систему виявлення аномалій наше завдання саме виявити аномальний трафік а не визначити яка саме атака використовується,

тому в для тренування моделі в якості ознаки яку модель буде визначати буде ознака ‘attack’ яка буде мати значення 0 – для нормального трафіку і 1 – для атаки.

Також так як в наборі даних присутній значний дисбаланс даних, а саме записів одного класу набагато більше або менше ніж інших потрібно використати техніки які дозволяють штучно згенерувати додаткові записи для класів в яких таких записів мало та вибрати вибірку даних для класів в яких велика кількість записів.

Для вибірки використовується RandomUnderSampler [49] який випадковим чином вибирає потрібну кількість записів для кожного класу.

Також для штучної генерації використовується алгоритм SMOTE [50] який генерує синтетичні данні які відображають реальні ознаки даних.

Отриманий набір даних після балансування наведений на рисунку 3.3. Так як для тренування моделі ми використовуємо лише мітку чи певний запис атака чи ні то сума кількості записів усіх атак після балансування дорівнює кількості записів нормального трафіку, результат наведено на рисунку 3.4.

Рисунок 3.3 – Розподіл трафіку в тренувальному наборі даних після балансування

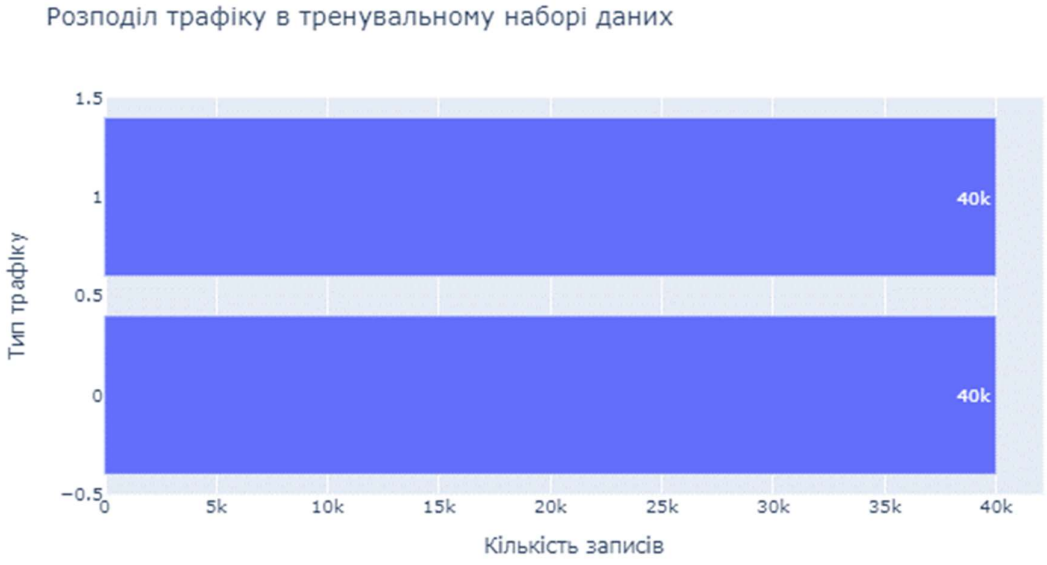

Рисунок 3.4 – Розподіл трафіку в тренувальному наборі даних

Таким чином в наступному розділі будуть описані результати тренування моделі на тренувальному наборі даних та на його збалансованій версії.

Також для вибору параметрів для тренування моделі використовувались RandomizedSearchCV [52] та GridSearchCV [53]. RandomizedSearchCV дозволяє задавши великий діапазон для кожного параметру випадковим чином знайти діапазон менший за початковий в якому модель показує найкращі результати тренування і далі використовується GridSearchCV для пошуку в меншому діапазоні найкращих параметрів.

Лістинг програми що використовується для тренування моделі наведено в додатку (див. Додаток Б).

3.2 Результати класифікації

Існує багато класифікаційних показників для систем виявлення вторгнень, деякі з яких відомі під різними назвами. Таблиця 3.1 показує матрицю невідповідностей для двокласового класифікатора, яку можна використовувати для оцінки продуктивності системи виявлення вторгнень. Кожен стовпець матриці представляє класи в передбаченому класі, тоді як кожен рядок представляє класи в реальному класі.

IDS зазвичай оцінюють на основі таких метрик:

- Справжній позитивний коефіцієнт (TPR): це відношення правильно визначених атак або записів відповідного класу до загальної кількості атак або записів відповідного класу, наприклад якщо модель вірно визначила усі атаки то TPR дорівнюватиме 1, однак це дуже рідкісний показник, також цю метрику інколи називають чутливістю.
- Рівень хибнопозитивних результатів (FPR): це відношення записів нормального трафіку які були визначені системою як атака до кількості записів нормального трафіку
- Частота хибнонегативних результатів (FNR): хибнонегативний результат означає, що система ну в змозі визначити аномалію та визначає її як запис нормального трафіку.
- Точність: точність вимірює, наскільки точна IDS у виявленні нормальної чи аномальної поведінки трафіку. Він описується як відсоток усіх цих правильно передбачених випадків до всіх випадків.
- В класифікації влучність є відношенням атак які було визначено системою до записів які були визначені системою як атаки. Повнота є відношенням атак які було вірно визначено системою до кількості усіх атак які були в наборі даних. Влучність відрізняється від точності, яка є відношенням правильно визначених записів, як позитивних, так і негативних до усіх записів. Влучність стосується лише позитивних результатів. [51]

Далі надані результати моделі натренованої на збалансованому та незбалансованому наборі даних.

В таблиці 3.1 надано матрицю невідповідностей класифікації тестувального набору для моделі натренованої на незбалансованому наборі даних, під час такого тренування модель віддала перевагу більш численному класу аномалій і тому неправильно визначила декілька записів нормального трафіку як атака. У таблиці 3.2 наведено метрики тестувального набору даних для моделі натренованої на незбалансованому наборі даних.

Таблиця 3.1 – Матриця невідповідностей для моделі натренованої на незбалансованому наборі даних

		Передбачене значення	
		Нормальний трафік	Аномальний трафік
Справжнє значення	Нормальний трафік	102	5
	Аномальний трафік	0	733598

Таблиця 3.2 – Результати класифікації для моделі натренованої на незбалансованому наборі даних

Метрика	Значення
Accuracy	0.999
Precision	0.999
Recall	1.0
ROC AUC	0.999

В таблиці 3.3 надано матрицю невідповідностей класифікації тестувального набору для моделі натренованої на збалансованому наборі даних, під час такого тренування під час такого тренування модель краще змогла визначити нормальний трафік і визначила лише один запис

нормального трафіку як атаку, але також модель схоже пропустила 20 записів атак які були схожими на нормальну поведінку. У таблиці 3.4 наведено метрики тестувального набору даних для моделі натренованої на збалансованому наборі даних.

Таблиця 3.3 – Матриця невідповідностей для моделі натренованої на збалансованому наборі даних

		Передбачене значення	
		Нормальний трафік	Аномальний трафік
Справжнє значення	Нормальний трафік	106	1
	Аномальний трафік	20	733578

Таблиця 3.4 – Результати класифікації для моделі натренованої на збалансованому наборі даних

Метрика	Значення
Accuracy	0.999
Precision	0.999
Recall	1.0
ROC AUC	0.999

Однією із переваг моделі RandomForest є те що вона після тренування дозволяє отримати результати важливості ознак на який вона була натренована, на рисунках 3.5 та 3.6 наведено важливість ознак для незбалансованого та збалансованого наборів даних.

Рисунок 3.5 – Важливість ознак моделі натренованої на незбалансованому наборі даних

Рисунок 3.6 – Важливість ознак моделі натренованої на збалансованому наборі даних

Так як обидві моделі отримали доволі високий результат на тестовому наборі даних, ми можемо вважати що обидва підходи до створення набору даних для тренування моделі є непоганими, також ми можемо спостерігати що обидві моделі визначили схожі ознаки як найважливіші, наприклад ознака N_IN_Conn_P_DstIP визначена найважливішою в обох моделях. Також так як обидві моделі отримали такий високий результат ми можемо зробити висновок що цих даних достатньо для тренування моделі для виявлення наведених у прикладі атак.

Надалі використовується модель натренована на збалансованому наборі даних, так як вона змогла краще визначити нормальний трафік, а це є важливішим для нашого експерименту.

3.3 Реалізація рішення для перевірки трафіку в реальному часі

Щоб модель натренована в попередньому розділі могла працювати в реальній системі дані які обробляє така система повинні бути приведені в такий ж формат як і дані в тренувальному та тестувальному наборах. На рисунку 3.5 наведено схему роботи такої системи.

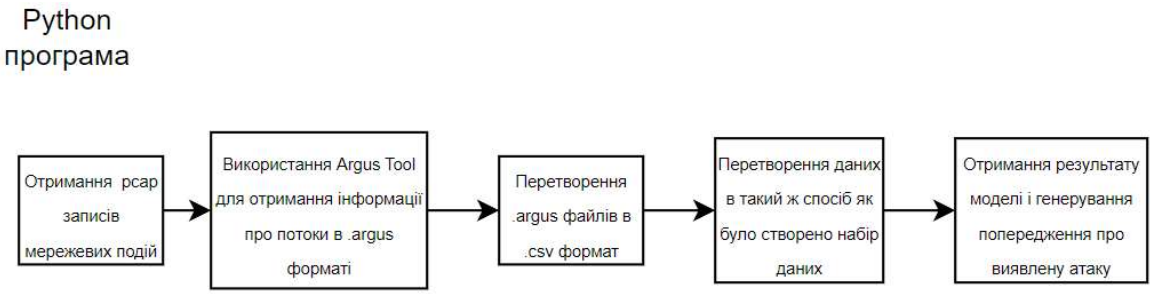

Рисунок 3.7 – Схема роботи IDS системи яка використовуватиме натреновану модель

Програма створена використовуючи мову програмування Python.

Для захоплення записів мережевих подій використовується Python модуль Pyshark [58] що є обгорткою для використання tshark [59] і дозволяє використовувати всі його функції в Python програмі. Сам tshark (Terminal wierSHARK) це утиліта командного рядку (CLI) яка має більшість, якщо не усі функції Wireshark. До того ж tshark дозволяє використовувати функції із запису мережевих пакетів всередині скриптів.

Для перетворення мережевих записів (pcap) в інформацію про мережеві потоки використовується програма Argus. Argus — це перша система мережевого потоку, розроблена Картером Буллардом на початку 1980-х років у Georgia Tech і адаптована для реагування на інциденти кібербезпеки в першій Групі реагування на надзвичайні ситуації з комп'ютером (CERT) в Інституті розробки програмного забезпечення Карнегі-Меллона наприкінці 1980-х років. Відтоді технологія мережевого потоку стала критично важливою частиною сучасної мережі та кібербезпеки, і Argus був активною частиною цієї еволюції.

Argus — це технологія мережевого аудиту, що забезпечує механізм аудиту мережевої активності для всього мережевого трафіку, а не лише для IP. Його було створено за моделлю телефонних мереж загального користування (PSTN) Call Detail Record (CDR) і розроблено для обліку всіх мережевих дій таким чином, щоб підтримувати всі типи функцій керування мережею, включаючи керування безпекою. Аудит є основним контролем безпеки NIST.

Проект Argus — це проект із відкритим вихідним кодом, зосереджений на демонстрації підтвердження концепції всіх аспектів широкомасштабної обізнаності про мережу, отриманої з даних мережевого потоку. Argus намагається бути «передовим краєм» технології мережевого потоку, дуже швидко обробляючи пакети, як по дроту, так і під час захоплення, у найбагатші доступні дані мережевого потоку. Система Argus намагається вирішити велику кількість проблем обробки даних мережевого потоку; масштаб, продуктивність, застосовність, конфіденційність і корисність.

Незважаючи на те, що Argus є концептуальним проектом, він оперативно використовується в Уряді США, Міністерстві оборони США, Міністерстві охорони здоров'я, Міністерстві економіки США, великих корпораціях та університетських мережах по всьому світу. Він широко використовується в дослідженні мереж, підтримуючи різноманітні проекти з аналізу продуктивності мережі, ситуаційної обізнаності, кібербезпеки, машинного навчання та навіть проектування чіпів програмно-визначених мереж (SDN).

Архітектура Argus розроблена для підтримки малих і дуже великих мережевих аудитів. Дані в реальному часі надають багато інформації, яку можна зберігати у файлах для подальшої обробки, або клієнтські програми можна об'єднати, щоб забезпечити потоки мережевих даних у реальному часі для простої обізнаності про мережу, широкомасштабної розподіленої видимості, навіть активного кібернетичного режиму. захист. [60]

Для встановлення програми Argus достатньо ввести команду наведену в рисунку 3.8 в командний рядок ОС Debian.

Рисунок 3.8 – Встановлення Argus

Щоб перетворити запис мережевих пакетів використовуються команди Argus наведені на рисунку 3.9, перша команда обробляє створений за допомогою Pyshark .pcap файл і створює .argus файл який містить усю інформацію про мережеві потоки, наступна команда записує ці данні у .csv формат, при цьому ми також вказуємо потрібні нам ознаки.

```
> sudo argus -r capture-output.pcap -w capture.argus
> ra -r capture.argus -c , -s +stddev -s +min -s +max -s +mean -s +drate -s +srate > capture1.csv
```

Рисунок 3.9 – Використання Argus для отримання інформації про мережеві потоки

Перед тим як використати натреновану модель на мережевих потоках нам потрібно обробити дані, для цього використовуються наступні кроки:

- Забираємо непотрібні ознаки
- Створюємо поля N_IN_Conn_P_SrcIP та N_IN_Conn_P_DstIP агрегуючи дані з використанням ковзаючого вікна розміром в 100 записів.
- Замінюємо текстові значення портів, протоколів, та мітки стану на відповідні числові значення

Далі використовуючи натреновану модель ми отримуємо результати передбачення і виводимо їх в командний рядок, приклад наведено на рисунку 3.10, код програми наведено в додатку (див. Додаток В).

```
(dyplomna_code) root@DESKTOP-5RCLQMI:/home/fastik/Projects/dyplomna_code# python3 realtime_IDS.py
1671308044.5677543 - start traffic recording
Loaded model from rf_clf_balanced.joblib
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
1671308104.7385347 - finish traffic recording
1671308104.7385733 - start traffic recording
Checking for new files in raw_records
1671308104.7551806 - started analyzing ./raw_records/1671308044.5677543-record.pcap

1671308105.8257582 - finished analyzing ./raw_records/1671308044.5677543-record.pcap
Analyzed 19 network flows
Found 0 possible attacks
```

Рисунок 3.10 – Результат виконання програми в CLI

В наведеному прикладі програма створила файл який містить одну хвилину мережевого трафіку і проаналізувала його. Для того щоб ми не втратили жодного мережевого пакету ці дії виконуються паралельно за допомогою Python ProcessPoolExecutor [61], схема виконання програми наведена на рисунку 3.11.

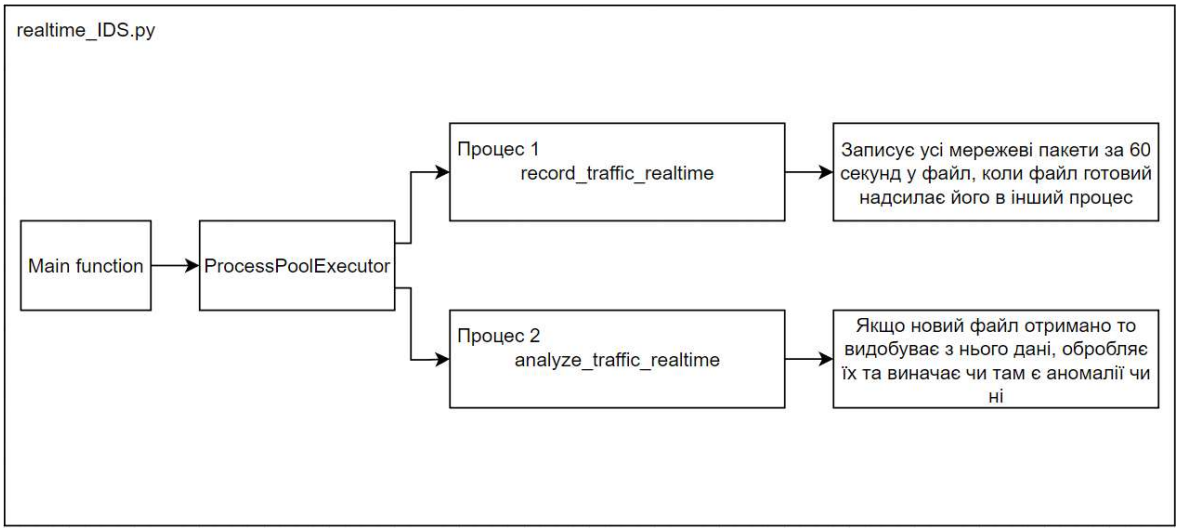

Рисунок 3.11 – Схема виконання програми

3.4 Тестування рішення для перевірки трафіку в реальному часі

На рисунку 3.10 вже наведено приклад виконання програми коли вона не визначила жодної аномалії, однак нам також потрібно перевірити її можливість визначати аномалії.

Для цього нам потрібно відхилитись від звичайної поведінки користувача яку вона вивчила, що насправді простіше ніж здається, на рисунку 3.13 наведено приклад коли програма визначає як аномалію безперервне надсилання ping [62] пакетів. ping — це утиліта для адміністрування комп'ютерної мережі, яка використовується для перевірки доступності хоста в мережі Інтернет-протоколу (IP). Вона доступна практично для всіх операційних систем, які мають мережеві можливості, включаючи більшість вбудованих програм для адміністрування мережі.

Ping вимірює час зворотного зв'язку для повідомлень, надісланих із вихідного хоста на комп'ютер призначення, які повертаються до джерела. Ping працює за допомогою пакетів ICMP (Internet Control Message Protocol). Пінгування передбачає надсилання ехо-запиту ICMP на цільовий хост і очікування ехо-відповіді ICMP. Програма повідомляє про помилки, втрату

пакетів і статистичне зведення результатів, як правило, включаючи мінімальний, максимальний, середній час зворотного зв’язку та стандартне відхилення середнього значення.

На рисунку 3.12 наведено приклад виявлення програмою великої кількості ping трафіку.

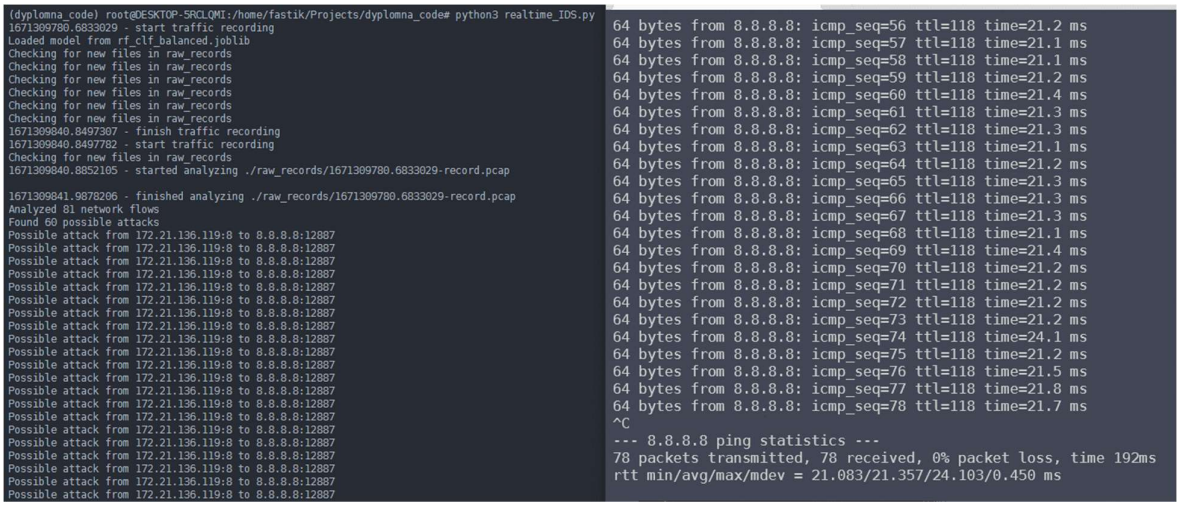

Рисунок 3.12 – Використання ping для створення мережевої аномалії

Трафік який генерує ping є аномальний для моделі, так як вона навчалась на даних в яких нормальний трафік не включав ping. Також безперервне надсилання ping могло схожим до атак DoS або Reconnaisance які були в тренувальному наборі даних.

Для подальшого тестування я використаю DoS програму Goldeneye яка також використовувалась для створення тренувального набору даних. Goldeneye — це безкоштовний інструмент із відкритим кодом, доступний на GitHub. За допомогою цього інструменту ми можемо здійснити атаку на відмову в обслуговуванні. Це фреймворк, написаний на .NET Core. Цей інструмент надає багато базових класів і розширень для використання у щоденній роботі. Цей інструмент дозволяє одному комп’ютеру знищити веб-сервер іншого комп’ютера, який використовує абсолютно законний трафік HTTP. Він створює повне TCP-з'єднання, а потім вимагає лише кілька сотень

запитів через тривалі та регулярні проміжки часу. Як результат, інструменту не потрібно використовувати багато трафіку, щоб вичерпати доступні з'єднання на сервері.

В якості серверу до якого буде встановлюватись з'єднання я використаю вбудований у Python HTTP файловий сервер, приклад створення такого сервера наведено на рисунку 3.13.

```
C:\Users\boris>python -m http.server 80
Serving HTTP on :: port 80 (http://[::]:80/) ...
```

Рисунок 3.13 – Запуск сервера

Використовуючи команду наведену на рисунку 3.14 можна запустити Goldeneye для атаки на запущений сервер. На рисунку 3.15 наведено результати створеної системи IDS.

```
❯ goldeneye http://192.168.1.11:80

GoldenEye v2.1 by Jan Seidl <jseidl@wroot.org>

Hitting webserver in mode 'get' with 10 workers running 500 connections each. Hit CTRL+C to cancel.
0 GoldenEye strikes deferred. (1665 Failed)
^CCTRL+C received. Killing all workers
Shutting down GoldenEye
```

Рисунок 3.14 – Запуск утиліти Goldeneye

```
1671312936.4165568 - started analyzing ./raw_records/1671312873.419464-record.pcap

1671312937.4931042 - finished analyzing ./raw_records/1671312873.419464-record.pcap
Analyzed 22 network flows
Found 0 possible attacks

Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
Checking for new files in raw_records
1671312994.00371 - finish traffic recording
1671312994.0044456 - start traffic recording
Checking for new files in raw_records
1671312998.1724837 - started analyzing ./raw_records/1671312933.6384268-record.pcap

1671312999.8464031 - finished analyzing ./raw_records/1671312933.6384268-record.pcap
Analyzed 31 network flows
Found 10 possible attacks
Possible attack from 172.21.136.119:223380 to 192.168.1.11:80
Possible attack from 172.21.136.119:223488 to 192.168.1.11:80
Possible attack from 172.21.136.119:223490 to 192.168.1.11:80
Possible attack from 172.21.136.119:223492 to 192.168.1.11:80
Possible attack from 172.21.136.119:223496 to 192.168.1.11:80
Possible attack from 172.21.136.119:223504 to 192.168.1.11:80
Possible attack from 172.21.136.119:223506 to 192.168.1.11:80
Possible attack from 172.21.136.119:223508 to 192.168.1.11:80
Possible attack from 172.21.136.119:223382 to 192.168.1.11:80
Possible attack from 172.21.136.119:223494 to 192.168.1.11:80
```

Рисунок 3.15 – Результат виконання IDS

Як результат можна зазначити що створена система може впоратись із виявленням аномального шкідливого трафіку якщо для її тренування використовувались достатньо вичерпні набори даних.

РОЗДІЛ 4. ОХОРОНА ПРАЦІ ТА БЕЗПЕКА В НАДЗВИЧАЙНИХ СИТУАЦІЯХ

4.1 Охорона праці

Охорона праці за своєю сутністю є турботою про людину у процесі використання її праці і розглядається як охорона працездатності людини. З іншого боку, відносини щодо охорони праці — невід'ємна складова організації процесу праці, що створює умови для стабільної та успішної трудової діяльності громадян.

Охорону праці і здоров'я громадян віднесено до пріоритетних напрямків соціальної політики України. Так, Конституція України одним з основних соціальних прав громадян визначає право кожного на належні, безпечні й здорові умови праці, встановлює, що використання праці жінок і неповнолітніх на небезпечних для їхнього здоров'я роботах забороняється. Право на охорону здоров'я закріплено і в Основах законодавства України про охорону здоров'я.

Нормативно-правові акти з охорони праці охоплюють правила, норми, регламенти, положення, стандарти, інструкції та інші документи, обов'язкові для виконання. В нормативно-правових актах представлені різні види норм права (охоронні, регулятивні, захисні, дефінітивні та ін.).

З метою усунення причин, які можуть викликати небезпечні для життя і здоров'я працівника ситуації, нормативно-правові акти з охорони праці визначають, що саме роботодавець має виконувати у сфері безпеки праці. Положення цих вимог стосуються усіх компонентів виробничого процесу: якості обладнання, оснащення робочих місць засобами колективного та індивідуального захисту, прийомів безпечного ведення робіт, методів нейтралізації факторів небезпечного та шкідливого впливу на працівника, порядку та розмірів компенсації за несприятливі умови праці та заподіяну здоров'ю шкоду.

Існує набір нормативно-правових актів з охорони праці що регулює роботу із комп'ютерними системами, яких потрібно дотримуватись при роботи із машинним навчанням.

Умови й організацію праці при роботі з візуальними дисплейними терміналами усіх типів вітчизняного та зарубіжного виробництва на основі електронно-променевих трубок, що використовуються в електронно-обчислювальних машинах (ЕОМ*) колективного використання та персональних, визначають Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин (ДСанПІН 3.3.2.007-98), затверджені постановою головного держаного санітарного лікаря України від 10 грудня 1998 р. № 7.

Мінімальні вимоги безпеки та захисту здоров'я під час роботи, пов'язаної з використанням екранних пристроїв незалежно від їхнього типу та моделі, визначають Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями, затверджені наказом Міністерства соціальної політики України від 14 лютого 2018 р. № 207 (далі — НПАОП 0.00-7.15-18).

Роботодавець повинен забезпечити навчання і перевірку знань з питань охорони праці та безпечного використання екранних пристроїв до початку роботи з ними та проведення медичних оглядів працівників (п. 2, 6 розділу II НПАОП 0.00-7.15-18).

Вимоги безпеки щодо використання комп'ютерної техніки визначають:

- ДСТУ EN 41003:2014 «Обладнання, яке підключають до телекомунікаційних мереж та/або кабельних розподільчих систем. Додаткові вимоги щодо безпеки».
- ДСТУ EN 60335-1:2015 «Прилади побутові та аналогічні електричні. Безпека. Частина 1. Загальні вимоги».
- ДСТУ EN 60950-1:2015 «Обладнання інформаційних технологій. Безпека. Частина 1. Загальні вимоги» (далі — ДСТУ EN 60950-1:2015).

- ДСТУ EN 61140:2015 «Захист проти ураження електричним струмом. Загальні аспекти щодо установок та обладнання» (далі — ДСТУ EN 61140:2015).
- ДСТУ EN 62368-1:2017 «Обладнання аудіо-, відео-, інформаційних та комунікаційних технологій. Частина 1. Вимоги щодо безпеки».

Приміщення, в яких планується установка та подальша робота з комп'ютером, повинні відповідати проектній документації будинку, погодженій з уповноваженими державними органами. Крім того, роботодавець повинен враховувати чинні санітарні нормативи освітлення, вимоги до параметрів мікроклімату (температура, відносна вологість), ступеня і сили вібрації, звукового шуму і вогнестійкості приміщення, а також характеристики електромагнітного, ультрафіолетового та інфрачервоного полів.

Роботодавцю заборонено установлювати комп’ютери в приміщеннях, розташованих у підвалах будинків. Для уникнення можливих аварій та замикань, поряд з приміщеннями, де вестиметься робота з комп’ютером (над чи під ними), також не дозволяється проведення робіт, що потребують здійснення надмірно вологих технологічних процесів.

Відповідне приміщення повинно бути укомплектоване системами центрального або індивідуального опалення, кондиціонування чи вентиляції повітря. Але при установці зазначених систем, необхідно переконатись, що батареї опалення, водопровідні труби, вентиляційні кабелі тощо, надійно сховані під захисними щитками, які перешкоджатимуть можливому потраплянню робітника під напругу.

У кожній кімнаті, де обладнуватимуться робочі місця співробітників, що працюватимуть на комп’ютері, повинні бути наявні елементи природного та штучного освітлення. При цьому, на вікнах слід встановити легко регульовані жалюзі чи штори, які дозволять працівникам коригувати рівень освітлення в

приміщенні. Бажано розмістити комп’ютери в кімнаті таким чином, щоб світло потрапляло на екрани моніторів з півдня чи північного сходу.

З метою досягнення максимального рівня безпечності і охорони праці при роботі з комп’ютером, виробничі приміщення необхідно обладнати аптечками першої медичної допомоги, системами автоматичної пожежної сигналізації і вогнегасниками. В приміщенні, в якому разом працюють 5 або більше комп'ютерів, на видимому місці установлюється службовий вимикач, який у разі потреби дозволить повністю відключити електричне живлення кімнати.

Роботодавець, який використовує найману працю робітників, повинен забезпечити відповідність їхніх робочих місць комфортним та безпечним умовам.

Розмір одного робочого місця має становити не менше 6 квадратних метрів. При необхідності, суміжні робочі місця співробітників, що працюють з комп’ютером, слід розділити перегородками висотою до 2 метрів. При визначенні достатнього розміру приміщення і робочого місця на одну особу необхідно додатково враховувати шафи, сейфи, тумби або інші предмети меблів чи обладнання, які знаходяться в кімнаті.

На столі працівника можливо розмістити допоміжні для роботи пристрої (принтери, колонки, сканери), а також місця для зберігання документів, за умови, що це не обмежуватиме видимість екрану і не заважатиме працівнику. У разі надмірного шуму чи вібрації технічного обладнання, роботодавець повинен забезпечити працівників антивібраційними килимками.

Робочий стілець співробітника має бути підйомно-поворотним, легко регульованим за висотою та забезпечувати належну підтримку та зручне положення спини і хребта особи.

Щодня необхідно проводити вологе прибирання приміщення, та очищати робоче місце та безпосередньо монітор комп’ютера від запиленості.

На користувачів під час роботи з комп’ютерною технікою можуть діяти такі види небезпек:

- ураження електричним струмом;
- енергетична небезпека (виникає через коротке замикання: опіки, електрична дуга, викид розплавленого металу);
- небезпека загоряння;
- термонебезпека (дія високих температур через нагрівання конструктивних елементів);
- механічна небезпека (травми через падіння, дію рухомих частин, поріз за гострі частини конструктивних елементів);
- небезпека випромінювання (дія звукового (акустичного), високочастотного, інфрачервоного, ультрафіолетового й іонізуючого випромінювання, а також видимого світла когерентної високої інтенсивності (лазерного випромінювання);
- хімічна небезпека (контакт із деякими хімікатами, які використовують для того, щоб обслуговувати обладнання, або від вдихання їх парів).

ДСанПіН 3.3.2.007-98 визначає, що ті, хто працює з ВДТ ЕОМ і ПЕОМ, мають пройти обов’язкові медичні огляди: попередній — під час влаштування на роботу та періодичні — протягом трудової діяльності. Обов’язковим є проходження раз на два роки таких лікарів, як терапевта, невропатолога та офтальмолога.

У компанії мають бути чітко встановлені перерви для відпочинку працівників (окрім обідньої), як правило, тривалістю 10-15 хвилин раз на годину або дві залежно від складності роботи. У будь-якому випадку роботодавець повинен передбачити такий розпорядок роботи на підприємстві, щоб час неперервної роботи з комп’ютером був не більше ніж 4 години.

Додатково, для збереження належного рівня здоров’я та професійної придатності робітників, рекомендується виділити на підприємстві окреме

побутове приміщення для перепочинку працівників і зняття ними нервово-емоційного напруження, що виникає при роботі з комп'ютером.

4.2 Характеристика стихійних лих, аварій (катастроф) та їх наслідків

Стихійні лиха – небезпечні природні явища, як правило раптового походження, хоча іноді і прогнозовані за допомогою метеорогогії, але на інтенсивність яких люди впливати не можуть. Їх можна класифікувати: за швидкістю переміщення – землетруси, зсуви, цунамі, снігопади, ожеледі – швидкі; підвищення рівня води в ріках через інтенсивні опади або танення снігу, льоду (повіні), звільнення внутрішньої енергії Землі, виверження вулканів – повільні. Часто виникають потужні, високошвидкісні потоки повітря через швидкий перепад значень атмосферного тиску (урагани, смерчі, циклони). Стихійні лиха речовинного характеру можуть ініціювати виникнення різноманітних полів, які негативно впливають на здоров'я, самопочуття людини.

Стихійні явища часто виникають в комплексі, що значно посилює їх негативний вплив. Небезпечні природні явища визначаються трьома основними групами процесів – ендогенні, екзогенні та гідрометеорологічні.

Стихійні лиха, які характерні для України, за структурою можна поділити на прості, що включають один елемент – наприклад, сильний вітер, зсув або землетрус та складні. Вони складаються з декількох процесів однієї групи або кількох груп. Найбільші збитки спричиняють повені – 40%, на другому місці – циклони (20%), на третьому – посухи та землетруси (15%).

Деякі стихійні лиха (пожежі, обвали, зсуви і навіть землетруси) можуть виникати в результаті дій самих людей, тобто мають антропогенне походження, але наслідки їх завжди є діями сил природи. Для кожного стихійного лиха характерна наявність властивих йому вражаючих чинників, що несприятливо впливають на стан здоров'я, життя людини.

Причинами стихійних лих можуть бути:

- швидке переміщення речовини (землетрусу, зсуви);
- вивільнення внутриземної енергії (вулканічна діяльність, землетруси);
- підвищення рівня вод річок, ставків і морів (повені, цунамі);
- вплив надзвичайно сильного вітру (урагани, торнадо, циклони);

Важливо своєчасно провести роботи, спрямовані на локалізацію природного лиха, щоб зменшити зони руйнувань, звести до мінімуму кількість загиблих та постраждалих.

4.3 Види стихійних лих

Землетрус – це природне явище, що супроводжується підземними поштовхами і коливаннями земної поверхні, появою тріщин, зсувів у ґрунті, грязьових потоків, сніжних лавин, цунамі тощо. Землетруси зазвичай охоплюють великі території. При сильних землетрусах порушується цілісність грунту, руйнуються будівлі і споруди, виводяться з експлуатації комунально-енергетичні мережі, можливі великі людські жертви. Осередки землетрусів знаходяться на глибині 30-60 км, а інколи на глибині до 700 км.

Інтенсивність землетрусів вимірюють в балах. У нашій країні прийнята міжнародна шкала Ріхтера, відповідно до якої землетрусу поділяються за силою поштовхів лежить на поверхні землі на 12 балів. Умовно їх можна розділити на слабкі (1-4 бала), сильні (5-8 балів) і традиційно сильні, чи руйнівні (8 балів і від).

При 3-бальному землетрусі коливання невеликі і лише у приміщенні; при 5-бальному – гойдаються висячі предмети і всі в приміщенні відчувають поштовхи; при 6-бальному – з'являються ушкодження в будинках, при 8-бальному з'являються тріщини у будівлях. 10-бальний землетрус

супроводжується загальною руйнацією будинків та порушенням землі, 12-бальний призводить до зміни ландшафту.

Залежно від причини виникнення, землетрусу бувають:

- тектонічні – творяться у результаті переміщення мас земної кори під впливом внутрішніх напруг;
- вулканічні – виникають при виверженні вулканів. Зазвичай охоплюють невеликі райони і супроводжуються потоками лави, викидами попелу і газів. При виверженні підводних вулканів можуть утворюватися величезні волни-цунамі і утворюються нові острови;
- обвальні – спостерігаються при обрушении склепінь підземних карстових порожнин. Зазвичай мають локальний характері і здебільшого суттєвих руйнацій не приносять;
- моретрясіння – різке коливанням води в морях і океанах під час землетрусів, осередок яких міститься під дном моря (океану) чи прибережних районах.

Сейсмоактивні зони оточують Україну на південному заході і півдні: Закарпатська, Вранча, Кримсько-чорноморська та Південно-Азовська. В сейсмічному відношенні найбільш небезпечними областями в Україні є Закарпатська, Івано-Франківська, Чернівецька, Одеська та Автономна Республіка Крим.

Серед всіх стихійних лих за даними ЮНЕСКО землетруси займають перше місце в світі за результатами економічної шкоди та кількістю загиблих. Найменш вірогідні землетруси на древніх платформах, наприклад на Українському кристалічному масиві (майже вся територія України), бо тут вже закінчилися природні тектонічні процеси і товщина земної кори складає біля 90 км.

Тільки за останні 100 років від землетрусів загинуло більше двох мільйонів людей. Тепер все частіше виникають антропогенні землетруси, тобто землетруси, що виникли з вини людства.

Провокують початок землетрусів, навіть там де вони не повинні бути, так звані штучні "моря", особливо об'ємом більше 1 куб. км, пустоти після видобутку нафти, газу, вугілля. В Україні - це шельф Чорного та Азовського морів (активізувалась діяльність грязьових вулканів біля Керчі), Донбас.

Основним способом зниження втрат і шкоди при землетрусах є будівництво сейсмостойких будинків та споруд.

Найкраща міра захисту – це швидко (протягом 15-20 секунд після першого поштовху) залишити приміщення, від нього з боку відкрите місце. Якщо це зробити неможливо – сховатися в заздалегідь обраному місці: дверному отворі, в прорізах внутрішніх вертикальних стін, кутках, освічених капітальними стінами, місцях у колон й під балками каркаса.

Повінь – це значне затоплення місцевості внаслідок підйому рівня води у річці, озері, водосховищі, викликане припливом води під час сніготанення чи злив, вітрових нагонів води, при заторах льоду на річках, прориві гребель і огороджуючих дамб, завалах річок при землетрусах, гірських обвалах чи селевих потоках. Повені часто супроводжуються людськими жертвами і завдають величезні матеріальні збитки: пошкоджуються і руйнуються житлові і виробничі будинки, автомобільні і залізничні шляхи, лінії електропередач, зв'язку, загибель худоби і врожаю сільськогосподарських культур, псування і знищення сировини, палива, продуктів, кормів і добрив та інших.

Повені можна прогнозувати: встановити час, характер, очікувані його розміри і організувати застережні заходи, які значно знижуватимуть збитки, створити сприятливі умови для рятувальних і невідкладних аварійно-відбудовних робіт. При прогнозованому затопленні населення оповіщається заздалегідь. У повідомленні про загрозу повені даються гідрометеодані, вказується порядок дій населення і порядок евакуації.

Річки Карпат і Криму в середньому дають 6—7 повеней на рік у будь-який сезон року, що часто спричиняє катастрофічні наслідки із загибеллю людей і масовими руйнуваннями. Небезпечним є й те, що повені на гірських

річках формуються дуже швидко, від кількох годин до 2—3 діб. У таких ситуаціях ставляться високі вимоги до оперативності прогнозування та оповіщення.

Катастрофічні повені в Криму і Карпатах у період з 1960 по 2003 р. були 14 разів. За післявоєнний час на Закарпатті сталося багато високих паводків, які завдали значних збитків господарству. Це паводки 1947,1957,1970,1980,1992 (два), 1993,1995 (два), 1997 рр. і катастрофічний дощовий паводок 4—8 листопада 1998 р., коли рівні води в річках на 1,8—2,6 м перевищили передпаводкові показники. Під час цього паводку загинуло 17 чоловік, зруйновано або стали непридатними 2695 житлових будинків, 2877 потребували ремонту.

Повені Дніпра, Дністра, Дунаю та Сіверського Донця супроводжуються затопленням значних територій, у тому числі сільськогосподарських угідь, де гинуть посіви культур. Це вимагає проведення евакуації населення, сільськогосподарських тварин і машин, посівного матеріалу і кормів. При таких затопленнях небезпечною є загроза затоплення хімічно небезпечних об'єктів.

Головна причина підтоплення — це незадовільний стан дренажних систем водовідведення.

Перед евакуацією необхідно відключити газ, воду, електрику, загасити палаючі печі, перенести на верхні поверхи будинків (горища) цінні речі й предмети, закрити вікна і двері перших поверхів, і оббити їх дошками. З отриманням попередження про евакуацію необхідно зібрати необхідні документи, гроші й цінності, медичну аптечку, комплект одягу по сезону, запас продуктів кілька днів і прибути на збірний пункт відправлення безпечний район.

Основний напрям боротьби з повенями полягає у зменшенні максимального рівня витрати води у річках, шляхом перерозподілу стоку води

за допомогою водоймищ, будівництва дамб і відводу води в русла інших рік і водосховища.

Ці явища природи є надзвичайно швидкими переміщеннями повітряних мас, що найчастіше завдають катастрофічних наслідків. Градація швидкостей вітру подається за шкалою Бофорта. У ньому прийнята 17-бальна система розподілу швидкостей вітру і подані приблизні руйнації, які можуть виникнути під час різної силі вітру. До метеорологічних небезпечних явищ, що бувають в Україні, належать: сильні зливи (Карпатські та Кримські гори), град (на всій території України); сильна спека (Степова зона); посуха, суховії (Степова та східна Лісостепова зони); урагани, шквали, смерчі (більша частина території); пилові бурі (південний схід Степової зони); снігові заноси (Карпати); значні ожеледі (Степова зона); сильний мороз (північ Полісся та схід Лісостепової зони); сильні тумани (південний схід Степової зони); шторми, урагани, ураганні вітри, смерчі, зливи, ожеледі й заметілі, сильні тумани (узбережжя й акваторія Чорного і Азовського морів).

Щорічно в Україні буває до 150 випадків стихійних метеорологічних явищ: снігопади, сильні дощі, ожеледі, тумани, рідше пилові бурі, крижані обмерзання. В 2004 р. в Україні виникло 2516 НС метеорологічного походження.

Від стихійних метеорологічних явищ зимою і літом частіше потерпають Степова зона, Карпати — від сильних злив, селевих потоків, граду, сильних вітрів, туманів, сильних снігопадів і заметілей. Тільки за останнє десятиріччя XX ст. в Україні зафіксовано 240 випадків катастрофічних природних явищ метеорологічного походження.

Урагани і тайфуни часто виникають під час проходження глибоких циклонів – гігантських атмосферних вихорів з спадним до центра тиском повітря. Це вітри силою 12 і більше балів (швидкість більш 29 м/с), що спричиняють найсильніші руйнації. Тривалість існування урагану (тайфуну) сягає 9-12 діб. Вони супроводжуються зливами, снігопадами, градом,

електричними розрядами дають великі руйнації народному господарству: зносять легкі будівлі та ушкоджують міцні, обривають дроти ліній електропередач, зв'язку, спустошують поля, ламають і вивертають з корінням дерева. У результаті люди гинуть чи отримують травми різної тяжкості, контузії.

Шторм під час руху повітряних мас від поверхні моря (океану) викликає сильне хвилювання та хвилі. Висота хвиль сягає 10-12 метрів і більш, що зумовлює пошкодження і загибель судів.

Буря – це теж сильний вітер, що спостерігається зазвичай під час проходження циклону і що супроводжується руйнаціями суші. Швидкість вітру сягає 16-27 м/с (60-100 км/год), а тривалість – від кількох годин до кількох діб. Пилові бурі виникають щорічно в Україні в різних областях, але частіше в Степовій зоні. У зимово-весняний період у центральних та східних областях України бувають сніжно-пилові бурі.

Особливо небезпечні пилові бурі для сільського господарства: знищується орний шар ґрунту, зносяться і руйнуються посіви, засипаються шаром пилу, піску великі території сільськогосподарських посівів, засипаються піском сільськогосподарські рослини.

Смерч (торнадо) – вихровий рух повітря, що виникає у грозовій хмарі, та розповсюджується у вигляді чорного рукави до самої землі. Коли смерч опускається до землі, основа його нагадує вирву, діаметром кілька десятків метрів. Рух повітря – проти годинникової стрілки зі швидкістю до 100 м/с (360 км/год). Тиск повітря всередині воронки різко знижений, тому туди засмоктується усе, що вихор може відірвати від землі і різко підняти спіраллю вгору, переносячи на значні відстані. Рухаючись над місцевістю, смерч руйнує будівлі, лінії передач, мости тощо.

Виникають смерчі майже щорічно то в одній, то в іншій області (1—2 рази на рік), переважно в серпні, мають невелику тривалість (до десяти хвилин). Частіше вони виникають у Центральному Поліссі і Степовій зоні,

особливо в Запорізькій, Херсонській областях і в Криму. За останні 20 років XX ст. в Україні зареєстровано 34 випадки смерчів з людськими жертвами і значними збитками, особливо в сільському і лісовому господарствах. Тому руйнівну силу смерчів можна порівняти з ударною хвилею осередку ядерного ураження. Ураганні вітри руйнують будівлі, лінії електропередачі та зв'язку, розкидають скирти сіна і соломи, спустошують посіви, пошкоджують транспортні магістралі й мости, призводять до аварій на комунально-енергетичних мережах, а головне — до людських жертв.

Кращий засіб порятунку з наближенням торнадо – сховатися у звичному притулку. Якщо смерч застав Вас у дорозі, на відкритій місцевості, найкраще сховатися в кюветі дороги, ямі, рові, яру і добре притиснутися до землі. У місті треба негайно залишити автомобіль, автобус, трамвай й затаїтися у найближчому підвалі, метро, підземному переході.

Град — це частинки льоду, різні за розмірами, формою, структурно неоднорідні, випадають із шарувато-дощових хмар у теплий період року. Град завдає великих збитків сільському господарству, особливо від червня до середини вересня, у Криму, Полтавській, Тернопільській, Чернівецькій, Луганській, Сумській, Запорізькій, Херсонській, Миколаївській і Одеській областях, на Волині, Поділлі й Приазов'ї.

Тумани. З'являються в основному в холодну пору року — у жовтні — квітні. Особливо поширені у гірських районах Карпат і Криму, інколи і на Південному березі Криму. В цих районах близько 100 днів бувають з туманами, а з сильними — до 80. На Приазовській, Придніпровській, Волинській, Подільській височині й Донецькому кряжі з туманами бувають близько 80 днів, а з сильними до 30. У Степовій зоні, на рівнині південної частини тумани бувають ЗО днів на рік, а сильні — до 20 днів.

Сильні снігопади і заметілі — це інтенсивне випадання снігу більше 20 мм за півдоби (визначається шаром талої води), що призводить до погіршення видимості та припинення руху транспорту.

Снігові замети утворюються під час інтенсивного випадання снігу при буранах, заметілях. При низових заметілях багато снігу нагромаджується в населених пунктах, на території тваринницьких ферм. Снігом заносяться залізничні й автомобільні шляхи. Порушується нормальне життя населених пунктів. У багатьох районах через великі замети може тимчасово припинитися доставка продуктів харчування і кормів.

Великі снігопади один раз на три роки спостерігаються в Черкаській, Київській, Вінницькій, Чернівецькій областях і в Криму, а один раз на п'ять років у Чернігівській, Сумській, Дніпропетровській, Рівненській, Тернопільській, Миколаївській і Запорізькій областях.

Майже щорічно виникають заметілі в різних регіонах України, особливо в Донбасі, Криму і Карпатах.

При наближенні снігопадів, буранів, заметілей, важливо, щоб система повідомлення своєчасно попередила підприємства, сільськогосподарські об'єкти та населення.

При загрозі виникнення снігової бурі запобіжні заходи в основному такі самі, що й при наближенні урагану. Снігова буря може тривати кілька днів, тому необхідно створити запаси продуктів харчування, води, предметів першої необхідності, кормів для сільськогосподарських тварин, обмежити пересування, закрити школи, дитячі садки і ясла.

Ландшафтні пожежі мають причинами виникнення необережне поводження з вогнем, порушення правил пожежної безпеки, удари блискавок, і навіть самозаймання торфу і сухий рослинності. Основними видами пожеж як стихійних лих, що охоплюють великі території, є:

1) лісові пожежі – некероване горіння рослинності, розповсюджується площею лісу у посушливе сезон:

– низові лісові пожежі характеризуються горінням лісової підстилки і підліска без захоплення крон дерев;

– верхові пожежі розвиваються, зазвичай, з низових і характеризуються горінням крон дерев;

– підземні (грунтові) пожежі виникають іноді як продовження лісових. Вони виникають у ділянках із торф'яними ґрунтами або у тих, які мають потужний шар підстилки. Горіння відбувається повільно, без пламені. Підгорають коріння дерев, які падають, створюючи завали.

2) торф'яні пожежі найчастіше бувають у місцях видобутку торфу, виникають зазвичай через неправильну роботу з вогнем, від розрядів блискавки чи самозагорания. Торф горить повільно протягом усієї глибину його залягання. Торф’яні пожежі охоплюють великі площі й важко піддаються гасінню.

3) степові (польові) пожежі виникають на відкритій місцевості за наявності сухої трави чи дозрілих хлібів. Вони мають сезонний характері і частіше бувають влітку, рідше – навесні та практично відсутні взимку.

З метою запобігання пожеж проводиться роз'яснювальна роботу з населенням про недопущення розведення багать у лісі і дотриманні запобіжних заходів при куріння тощо. Потрапивши до зони лісової пожежі необхідно з'ясувати напрям вітру, щоб визначити напрямок руху вогню й напрям маршруту виходу з лісу. В пожежонебезпечний сезон в лісі забороняється розпалювати вогонь, курити дозволяється тільки на спеціально обладнаних майданчиках. Забороняється спалювати сміття поблизу лісу. Якщо людина опинилася в палаючому лісі або на полі, то переходити лінію вогню необхідно проти вітру, рухатись ліпше по річці, струмках, просіках, шляхах.

При перебування у зоні пожежі рекомендується, якщо можливо, поринути у одязі у найближчу водойму. Виринувши з неї, обгорнути голову мокрою сорочкою чи чимось іншим. Щоб уникнути вдихання гарячого повітря або диму потрібно дихати через мокру тканину повітрям, прилеглим до землі, і рухатися під прямим кутом до подальшого поширення вогню.

Основними способами боротьби з лісовими і степовими пожежами є: захльостування крайки вогню, засипання його землею, заливання водою (хімікатами), створення загороджувальних і мінеральних смуг, пуск зустрічного вогню (відпал).

4.4 Висновки до 4 розділу

Таким чином, проаналізувавши причини виникнення та наслідки стихійних лих, можна зробити певні висновки.

В останні роки кількість стихійних лих в Україні та в світі в цілому значно збільшилася. Найчастіше в Україні виникають такі природні катастрофи як землетруси, повені, посухи (на Півдні України), лісові пожежі в літню пору року, снігові замети, зсуви поверхні.

Є серйозні підстави вважати, що масштабність впливу лиха й катастроф на соціальні, економічні, політичні та інших процесів сучасного нашого суспільства та їх драматизм вже перевищили такий рівень, який дозволяв ставитися до них як до локальних збоїв у розміреному функціонуванні державних та громадських структур.

Отже, перед людиною та громадськістю в XXI в. вимальовується нова мета - глобальна безпека. Досягти цього можна, в першу чергу, за допомогою зміни світогляду людини, а також покращення системи профілактичних заходів у боротьбі зі стихійними лихами, а саме: вдосконалення рятувальних служб та рятувальної техніки, проведення попереджувальних заходів та пропагандистської роботи з громадянами щодо правил поведінки та дій під час стихійних лих. Це допоможе в майбутньому зменшити кількість загиблих та постраждалих від природних катастроф, а також зменшить матеріальні збитки, що були завдані стихійним лихом.

Природні лиха з часом нікуди не зникнуть. Будуть виникати землетруси в геологічно активних районах, будуть виникати повені, а штормові припливи

стануть, раз у раз затопляти морські узбережжя, не обійдеться і пожеж. Людина безсила запобігти природним процесам, але тільки в наших силах зменшити кількість жертв і матеріальних втрат.

ВИСНОВКИ

Сучасні системи виявлення та запобігання вторгненням потрібно покращувати щоб краще виявляти невідомі та нові атаки, одним із найкращих способів для цього є використання машинного навчання, вона має ряд переваг:

- може виявляти раніше невідомі атаки;
- після фази тренування не потребує людського обслуговування;
- легко імплементується;
- може точно визначати атаки.

В ході виконання кваліфікаційної роботи було розглянуто найпопулярніші набори даних для створення чи тестування системи виявлення та запобігання вторгненням, головні підходи до створення такої системи а також типи таких систем та розглянуто вимоги до такої системи.

Було досягнуто завдання із вибору засобів розробки та побудовано архітектуру системи виявлення вторгнень використовуючи набір даних Bot-IoT та модель машинного навчання RandomForestClassifier для визначення аномального трафіку, та використання програм tshark та Argus для збору та обробки мережевого трафіку.

Було створено тестувальний набір даних який використовувався для оцінки якості та тестування створеної системи. Систему було протестовано за допомогою доступних даних.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Symantec, "Internet security threat report 2017," April, 7017 2017, vol. 22/ [Електронний ресурс]. URL: https://www.symantec.com/content/dam/symantec/docs/reports/istr-22-2017-en.pdf

2. Breach_LeveL_Index. (2017, November). Data breach statistics. [Електронний ресурс]. URL: http://breachlevelindex.com/

3. Australian. (2017, November). Australian cyber security center threat report 2017. [Електронний ресурс]. URL: https://www.cyber.gov.au/acsc/view-all-content/reports-and-statistics/acsc-threat-report-2017

4. R. Singh, P.-K. Mannepalli, “Survey on Feature Reduction Techniques of Intrusion Detection System”. International Journal of Engineering Research in Current Trends (IJERCT) ISSN: 2582-5488, Volume-2 Issue-3, Jun 2020

5. H.-J. Liao, C.-H. Richard Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: a comprehensive review," J Netw Comput Appl, vol. 36, no. 1, pp. 16–24, 2013a/01/01/ 2013

6. Khraisat A, Gondal I, Vamplew P (2018) An anomaly intrusion detection system using C5 decision tree classifier. In: Trends and applications in knowledge discovery and data mining. Springer International Publishing, Cham, pp 149–155

7. Kreibich C, Crowcroft J (2004) Honeycomb: creating intrusion detection signatures using honeypots. SIGCOMM Comput Commun Rev 34(1):51–56

8. Roesch M (1999) Snort-lightweight intrusion detection for networks. In: Proceedings of the 13th USENIX conference on system administration. Seattle, Washington, pp 229–238

9. Vigna G, Kemmerer RA (1999) NetSTAT: a network-based intrusion detection system. J Comput Secur 7:37–72

10. C. R. Meiners, J. Patel, E. Norige, E. Torng, and A. X. Liu, "Fast regular expression matching using small TCAMs for network intrusion detection and prevention systems," presented at the Proceedings of the 19th USENIX conference on security, Washington, DC, 2010

11. Lin C, Lin Y-D, Lai Y-C (2011) A hybrid algorithm of backward hashing and automaton tracking for virus scanning. IEEE Trans Comput 60(4):594–601

12. Butun I, Morgera SD, Sankar R (2014) A survey of intrusion detection systems in wireless sensor networks. IEEE Communications Surveys & Tutorials 16(1):266–282

13. A. Alazab, M. Hobbs, J. Abawajy, and M. Alazab, "Using feature selection for intrusion detection system," in 2012 international symposium on communications and information technologies (ISCIT), 2012, pp. 296–301

14. Creech G, Hu J (2014a) A semantic approach to host-based intrusion detection systems using Contiguousand Discontiguous system call patterns. IEEE Trans Comput 63(4):807–819

15. Ye N, Emran SM, Chen Q, Vilbert S (2002) Multivariate statistical analysis of audit trails for host-based intrusion detection. IEEE Trans Comput 51(7):810–820

16. Bhuyan MH, Bhattacharyya DK, Kalita JK (2014) Network anomaly detection: methods, systems and tools. IEEE Communications Surveys & Tutorials 16(1):303–336

17. L. Chao, S. Wen, and C. Fong, "CANN: an intrusion detection system based on combining cluster centers and nearest neighbors," Knowl-Based Syst, vol. 78, pp. 13–21, 4// 2015

18. S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, and F. Herrera, "On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems," Expert Syst Appl, vol. 42, no. 1, pp. 193–202, 1// 2015

19. Buczak AL, Guven E (2016) A survey of data mining and machine learning methods for cyber security intrusion detection. IEEE Communications Surveys & Tutorials 18(2):1153–1176

20. N. Walkinshaw, R. Taylor, and J. Derrick, "Inferring extended finite state machine models from software executions," Empirical Software Engineering, journal article vol. 21, no. 3, pp. 811–853, June 01 2016

21. Studnia I, Alata E, Nicomette V, Kaâniche M, Laarouchi Y (2018) A language-based intrusion detection approach for automotive embedded networks. Int J Embed Syst 10(1):1–12

22. G. Kim, S. Lee, and S. Kim, "A novel hybrid intrusion detection method integrating anomaly detection with misuse detection," Expert Syst Appl, vol. 41, no. 4, Part 2, pp. 1690–1700, 2014/03/01/ 2014

23. Kenkre PS, Pai A, Colaco L (2015b) Real Time Intrusion Detection and Prevention System. Springer International Publishing, Cham, pp 405–411

24. S. Dua and X. Du, Data mining and machine learning in cybersecurity. CRC press, 2016

25. L. Xiao, X. Wan, X. Lu, Y. Zhang, and D. Wu, "IoT security techniques based on machine learning," arXiv preprint arXiv:1801.06275 , 2018

26. Rutkowski L, Jaworski M, Pietruczuk L, Duda P (2014) Decision trees for mining data streams based on the Gaussian approximation. IEEE Trans Knowl Data Eng 26(1):108–119

27. X. Yang and Y. L. Tian, "EigenJoints-based action recognition using Naïve-Bayes-nearest-neighbor," in 2012 IEEE computer society conference on computer vision and pattern recognition workshops, 2012, pp. 14–19

28. L. Koc, T. A. Mazzuchi, and S. Sarkani, "A network intrusion detection system based on a hidden Naïve Bayes multiclass classifier," Expert Syst Appl, vol. 39, no. 18, pp. 13492–13500, 2012/12/15/ 2012

29. Hoque MAM, Bikas MAN (2012) An implementation of intrusion detection system using genetic algorithm. International Journal of Network Security & Its Applications 4:2

30. G. Wang, J. Hao, J. Ma, and L. Huang, "A new approach to intrusion detection using artificial neural networks and fuzzy clustering," Expert Syst Appl, vol. 37, no. 9, pp. 6225–6232, 2010/09/01/ 2010

31. S. Elhag, A. Fernández, A. Bawakid, S. Alshomrani, and F. Herrera, "On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on intrusion detection systems," Expert Syst Appl, vol. 42, no. 1, pp. 193–202, 1// 2015

32. C. Annachhatre, T. H. Austin, and M. Stamp, "Hidden Markov models for malware classification," Journal of Computer Virology and Hacking Techniques, vol. 11, no. 2, pp. 59–73, 2015/05/01 2015

33. W.-C. Lin, S.-W. Ke, and C.-F. Tsai, "CANN: an intrusion detection system based on combining cluster centers and nearest neighbors," Knowl-Based Syst, vol. 78, no. Supplement C, pp. 13–21, 2015/04/01/ 2015

34. Alcaraz C (2018) Cloud-assisted dynamic resilience for cyber-physical control systems. IEEE Wirel Commun 25(1):76–82

35. Ashfaq RAR, Wang X-Z, Huang JZ, Abbas H, He Y-L (2017) Fuzziness based semi-supervised learning approach for intrusion detection system. Inf Sci 378:484–497

36. Ashfaq RAR, Wang X-Z, Huang JZ, Abbas H, He Y-L (2017) Fuzziness based semi-supervised learning approach for intrusion detection system. Inf Sci 378:484–497

37. A. A. Aburomman and M. B. Ibne Reaz, "A novel SVM-kNN-PSO ensemble method for intrusion detection system," Appl Soft Comput, vol. 38, pp. 360–372, 2016/01/01/ 2016

38. D. M. Farid, N. Harbi, and M. Z. Rahman, "Combining naive bayes and decision tree for adaptive intrusion detection," arXiv preprint arXiv:1005.4496, 2010

39. MIT Lincoln Laboratory. (1999, June). DARPA Intrusion Detection Data Sets. [Електронний ресурс]. URL: https://www.ll.mit.edu/ideval/data/

40. Creech G, Hu J (2014b) A semantic approach to host-based intrusion detection systems using contiguous and Discontiguous system call patterns. IEEE Trans Comput 63(4):807–819

41. I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in ICISSP, 2018, pp. 108–116

42. Node-red tool. [Електронний ресурс]. URL: https://nodered.org/

43. Argus tool. [Електронний ресурс]. URL: https://qosient.com/argus/index.shtml

44. D. Soni, A. Makwana, A survey on mqtt: a protocol of internet of things (iot), in: Proceeding of the International Conference on Telecommunication, Power Analysis and Computing Techniques, Chennai: IN, 2017.

45. Using Argus. [Електронний ресурс]. URL: https://openargus.org/using-argus

46. CSV. [Електронний ресурс]. URL: https://uk.wikipedia.org/wiki/CSV

47. S. Paliwal, R. Gupta, Denial-of-service, probing & remote to user (r2l) attack detection using genetic algorithm, International Journal of Computer Applications 60 (19) (2012) 57–62

48. N. Hoque, M. H. Bhuyan, R. C. Baishya, D. K. Bhattacharyya, J. K. Kalita, Network attacks: Taxonomy, tools and systems, Journal of Network and Computer Applications 40 (2014) 307–324.

49. Imbdalanced learn RandomUnderSampling. [Електронний ресурс]. URL: https://imbalanced-

learn.org/stable/references/generated/imblearn.under_sampling.RandomUnderSampler.html

50. Imbdalanced learn SMOTE. [Електронний ресурс]. URL: https://imbalanced-learn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html

51. Влучність та повнота. [Електронний ресурс]. URL: https://uk.wikipedia.org/wiki/%D0%92%D0%BB%D1%83%D1%87%D0%BD%D1%96%D1%81%D1%82%D1%8C_%D1%82%D0%B0_%D0%BF%D0%BE%D0%B2%D0%BD%D0%BE%D1%82%D0%B0

52. Scikit-learn RandomizedSearchCV. [Електронний ресурс]. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html

53. Scikit-learn GridSearchCV. [Електронний ресурс]. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html

54. Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. Artificial Intelligence, 97(1–2), 273–324.

55. Zimek, A., Schubert, E., & Kriegel, H. P. (2012). A survey on unsupervised outlier detection in high-dimensional numerical data. Statistical Analysis and Data Mining, 5(5), 363–387.

56. Efron, B., Hastie, T., Johnstone, I., & Tibshirani, R. (2004). Least angle regression. Annals of Statistics, 32, 407–499.

57. M. Stamp (2018), “A Survey of Machine Learning Algorithms and Their Application in Information Security: An Artificial Intelligence Approach”. Guide to Vulnerability Analysis for Computer Networks and Systems (pp.33-55)

58. Pyshark Github page. [Електронний ресурс]. URL: https://github.com/KimiNewt/pyshark

59. tshark about page. [Електронний ресурс]. URL: https://tshark.dev/setup/about/

60. Argus tool. [Електронний ресурс]. URL: https://openargus.org/

61. Python documentation ProcessPoolExecutor. [Електронний ресурс]. URL: https://docs.python.org/3/library/concurrent.futures.html#concurrent.futures.ProcessPoolExecutor

62. Wikipedia Ping. [Електронний ресурс]. URL: https://uk.wikipedia.org/wiki/Ping

Borys LYPA[1], Oleh IVER [2], Viktor KIFER[3]

Supervisor: Nataliya ZAGORODNA[4]

APPLICATION OF MACHINE LEARNING METHODS FOR NETWORK INTRUSION DETECTION SYSTEM

Summary: The article is devoted to the application of the machine learning algorithms for intrusion detection in networks. Creating and training intrusion detection system (IDS) using machine learning is mainly limited by the out-of-date open available datasets. The most popular machine learning classification models like decision tree, random forest, and linear support vector classification will be researched based on the CSE-CIC-IDS2018 dataset.

Keywords: Intrusion detection, cybersecurity, machine learning, CSE-CIC-IDS2018 dataset

ZASTOSOWANIE METOD UCZENIA MASZYNOWEGO DO BUDOWY SYSTEMU WYKRYWANIA CYBERWŁAMAŃ

Streszczenie: W artykule omówiono zastosowanie algorytmów uczenia maszynowego do detekcji cyberwłamań w sieciach. Tworzenie oraz uczenie systemów detekcji cyberwłamań (IDS) z zastosowaniem uczenia maszynowego jest ograniczone głównie poprzez dostępność do zbiorów/zestawów danych, które są zdezaktualizowane (out-of-date). W pracy badano najbardziej popularne modele klasyfikacji oparte o uczenie maszynowe – takie jak: drzewa decyzyjne, losowe lasy (w sensie teorii grafów), a także liniową klasyfikacje wektorową. Badania te przeprowadzono na specjalnych zbiorach danych CSE-CIC-IDS2018.

Słowa kluczowe: :detekcja cyberwłamań, cyberbezpieczeństwa, uczenie maszynowe, zbiór danych CSE-CIC-IDS2018

[1] Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity; boryslypa1@gmail.com
[2] Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Computer Engineering; olehiwer@gmail.com
[3] PhD student, Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity department; kifervictor@gmail.com
[4] PhD, Ternopil Ivan Puluj National Technical University, Ukraine; Faculty of Computer Information Systems and Software Engineering; Cybersecurity department; zagorodna.n@gmail.com

1. Introduction

Internet and social networks are increasingly changing our life, but they also expose us to serious security threats.

Due to the fact that in the past years computer networks, services and systems faced more and more threats security in cyberspace has become a very important problem. Identification of various network attacks, especially not previously seen attacks is a key issue to be solved urgently.

IDS are an important part of cybersecurity. They detect intrusions and abnormal behavior in networks or other information systems. IDS usually apply one of two detection principles: signature-based or anomaly-based algorithms [2]. Signature-based approach means usage of manually created rules that detect intrusions, whereas anomaly-based systems try to profile normal behavior and detect abnormal situation dynamically. It is also possible to combine these approaches to form a hybrid IDS. In this case signatures are created automatically and can be periodically updated [3].

Nowadays almost all computer systems generate big data. Classical IDS cannot effectively process huge datasets and response to new threats. So it was a need to find a way of fast data analysis and effective search of anomalies. Machine learning (ML) and Data Mining algorithms came in handy and offers many benefits for intrusion detection. In general machine learning is an approach of artificial intelligence (AI) that uses a system which capable to learn from experience. In other words, ML is a system that can recognize patterns by using examples rather than by programming them. However, it has some restriction of usage which should be considered.

In the research, we use decision trees, random forest and support vector machine algorithms as they belong to the popular classification methods and work fast enough.

The performance of a machine learning algorithm largely depends on the dataset it is trained on. Since intrusion methods are improving rapidly, a valid dataset is required to construct adequate defense model [4].

2. Dataset

We built a classification model on a realistic cyber defense dataset provided by Canadian Institute for Cybersecurity (CIC) on AWS (Amazon Web Services) [5]. The final dataset includes seven different attack scenarios: Brute-force, Heartbleed, Botnet, DoS, DDoS, Web attacks, and infiltration of the network from inside. The attacking infrastructure includes 50 machines; the victim organization has 5 departments and includes 420 machines and 30 servers. The dataset includes the captures network traffic and system logs of each machine, along with 80 features extracted from the captured traffic using CICFlowMeter-V3 [6,7].

The description attacks are given below according to information provided on CSE-CIC-IDS2018 dataset`s page [9].

Brute force attacks consist of submitting many passwords or passphrases with the hope of eventually correct guess. They are very common against networks, as they tend to break the accounts with weak username and password combinations. There were two modules, FTP and SSH on the Kali Linux machine as the attacker and an Ubuntu 14.0 system as the victim machine. A large dictionary that contains 90 million words was used as a list of passwords.

The Heartbleed Bug is a serious vulnerability in the popular OpenSSL cryptographic software library. This weakness allows stealing the information protected under normal conditions by the SSL/TLS encryption which used to secure the Internet. SSL/TLS provides communication security and privacy over the Internet for applications such as web, email, instant messaging (IM) and some virtual private networks (VPNs) [8]. The Heartleech is one of the most famous tools to exploit Heartbleed which was used in this dataset.

In botnet scenario machines are infected with two different botnets (Zeus and Ares), every 400 seconds screenshots are also requested from the zombies. Zeus is a Trojan horse malware package that runs on some versions of Microsoft Windows. While it can be used to carry out many malicious and criminal tasks, it is often used to steal banking information by man-in-the-browser keystroke logging and form grabbing. It is also used to install the Crypto-Locker ransomware. Zeus is spread mainly through drive-by downloads and phishing schemes. Also, it is used as a complement Ares botnet which is an open-source botnet.

In the Denial-of-Service (DoS) a Slowloris Perl-based tool is used to take down the web server. Slowloris is a type of denial of service attack tool invented by Robert Hansen which allows a single machine to take down another machine's web server with minimal bandwidth and side effects on unrelated services and ports.

In the Distributed Denial-of-Service scenario the High Orbit Ion Cannon (HOIC) tool was used to conduct DDoS attack by using 4 different computers. The High Orbit Ion Cannon is an open source network designed as stress testing and denial-of-service attack application to attack as many as 256 URLs at the same time.

In the web application attacks scenario, Damn Vulnerable Web App (DVWA) was used which was developed to be an aid for security professionals to test their skills, as victim web application. In the first step, the website is scanned through a web application vulnerability scanner and then different types of web attacks are conducted on the vulnerable website, including SQL injection, command injection, and unrestricted file upload.

In the infiltration of the network from inside scenario, a vulnerable application (such as Adobe Acrobat Reader 9) should be exploited. First the victim receives a malicious document through the email. Then, after successful exploitation using Metasploit framework, a backdoor will be executed on the victim's computer. Now different attacks can be conducted on the victim's network include IP sweep, full port scan and service enumerations using Nmap.

The brief summary of dataset is CSV file with more than 80 features is given in Table 1.

Table 1. List of executed attacks and duration [9]

Attack	Tools	Duration	Attacker	Victim
Bruteforce attack	FTP – Patator SSH – Patator	One day	Kali linux	Ubuntu 16.4 (Web Server)
DoS attack	Hulk, GoldenEye, Slowloris, Slowhttptest	One day	Kali linux	Ubuntu 16.4 (Apache)

DoS attack	Heartleech	One day	Kali linux	Ubuntu 12.04 (Open SSL)
Web attack	• Damn Vulnerable Web App (DVWA) • In-house selenium framework (XSS and Brute-force)	Two days	Kali linux	Ubuntu 16.4 (Web Server)
Infiltration attack	• First level: Dropbox download in a windows machine • Second Level: Nmap and portscan	Two days	Kali linux	Windows Vista and Macintosh
Botnet attack	• Ares (developed by Python): remote shell, file upload/download, capturing • screenshots and key logging	One day	Kali linux	Windows Vista, 7, 8.1, 10 (32-bit) and 10 (64-bit)
DDoS+PortScan	Low Orbit Ion Canon (LOIC) for UDP, TCP, or HTTP requests	Two days	Kali linux	Windows Vista, 7, 8.1, 10 (32-bit) and 10 (64-bit)

List of extracted traffic features are described in table 2.

Table 2. List of extracted traffic features [10]

Feature	Description	Type
Label	indicate whether the traffic is malicious or not, e.g., benign, SQLInjection, etc.	String
Dst Port	Destination port number	Integer
Protocol	Protocol	Integer
TimeStamp	Time Stamp of the flow	String
Flow duration	Flow duration	Integer
Tot Fwd/Bwd Pkts	Total packets in forward/backward directions	Integer
TotLen Fwd/Bwd Pkts	Total size of packets in forward/backward directions	Integer
Fwd/Bwd Pkt Len Max/Min/Mean/Std	Maxi/Mini/Average/Std. Dev. size of package in forward/backward directions	Integer
Flow Byts/s & Flow Pkts/s	Flow byte rate, i.e., number of packets per seconds	Float64
Flow IAT Mean/Std/ Max/Min	Average/Std. Deviation/Maxi/Mini time between two flows	Float64
Fwd/Bwd IAT Tot/Mean/ Std/- Max/Min	Total/Average/Std. Deviation/Maxi/Mini time between two packets in forward/backward directions	Float64
Fwd/Bwd PSH/URG Flags	Number of times the PSH/URG flag was set in packets in forward/backward direction	Integer
Fwd/Bwd Header Len	Total bytes used for header in forward/backward direction	Integer
Fwd/Bwd Pkts/s	Number of forward/backward packets	Float64

	per second	
Pkt Len Min/Max/Mean/Std	Maxi/Mini/Average/Std. Dev. length of a flow	Integer
Pkt Len Var	Mini inter-arrival time of packet	Float64
FIN/SYN/RST/PUSH/ACK/ URG/CWE/ECE Flag Cnt	Number of packets with FIN/SYN/RST/PUSH/ACK-/URG/CWE/ECE	Integer
Down/Up Ratio	Download/upload ratio	Integer
Pkt Size Avg	Average size of packets in forward/backward direction	Float64
Fwd/Bwd Seg Size/Byts/b/Blk Rate Avg	Average number of bulk rate/bytes bulk rate/packets bulk rate in forward/backward directions	Float64
Subflow Fwd/Bwd Pkts/Byts	The average number of bytes/packets in a sub flow in forward/backward direction	Integer
Init Fwd/Bwd Win Byts	Number of bytes sent in initial window in forward/backward directions	Integer
Fwd Act Data Pkts	Number of packets with at least 1 byte of TCP data payload in forward	Integer
Fwd Seg Size Min	Minimum segment size observed in forward	Integer
Active Mean/Std/Max/Min	Maxi/Mini/Average/Std. Dev. a flow was active before becoming idle	Float64
Idle Mean/Std/Max/Min	Maxi/Mini/Average/Std. Dev. a flow was idle before becoming active	Float64

Moreover, before starting training a model we preprocess data, included in the CSE-CIC-IDS2018 dataset. Some steps were done to work out missing values and reduce the size of dataset:

- Delete features which do not affect the performance of ML model, for example, "TimeStamp" column;
- Replace "Infinity" and "NaN" values with mean value for each column;
- Format data into standard datatype;

Let us consider an example where the ML model takes into account a total number of packets in forward direction per second or flow byte rate per second to detect malicious traffic. It is quite reasonable that these features can vary from network to network due to differences in networks bandwidth. While running the model on test dataset we observed reduction in precision, so to prevent such noises we normalized data. Normalization makes training less sensitive to the scale of features, so we can find better solution. Most of the numerous data was replaced with its standard deviation and then rescaled from -1 to 1 by using `sklearn.preprocessing.MinMaxScaler`.

To simulate real-life traffic we downloaded .pcap files with similar attacks logs from Stratosphere Lab [11] containing normal and malicious traffic, extracted the same features as in training dataset by using CICFlowMeter-V4 [6,7]. We took the same preprocessing steps to the test dataset as to the training one.

3. Machine Learning Methods

Our research was conducted in order to identify whether flow is benign or malicious, based on learning on a set of labeled in advance data. In this case our problem belongs to a supervised classification problem.

We have selected the popular machine learning models:
- Decision tree classifier;
- Random forest classifier;
- Linear Support Vector Classification.

Decision tree is a tree structure, used as a predictive model, in which each node is created to test one feature, and a branch is a test output with each leaf node representing a category.

Random forest (RF). A random forest is a set of decision trees which considers the output of each tree before providing a unified final response. Each decision tree is a conditional classifier: the tree is analyzed from the top. A given condition is checked against one or more features of the analyzed data at each node. These methods are efficient for large datasets and especially for multiclass problems, but deeper trees might lead to overfitting [12].

Linear Support Vector Classification (LinearSVC) is a Support Vector Classification (SVC) with linear kernel, but have differences in implementation, and works better with large numbers of samples [13]

We implemented these models using Anaconda 3 and the latest Scikit learn version 0.21.3 [14] and Pandas version 0.25.3 [15] in Jupyter Notebook. For each evaluated model we found best parameters using sklearn.model_selection.GridSearchCV.

4. Evaluation

The evaluation of experiments include cross validation of the training CIC-AWS-2018 Dataset on each of the attack types and the prediction using the model on test dataset. The result of classifiers accuracy are presented in the following tables, grouped by type of attack. True Positive (TP) means the percentage of positive samples correctly classified by the model and True Negative (TN) means the percentage of negative samples correctly classified by the model Accuracy: (TP + TN)/(all instances = TP + TN + False Positive + False Negative). Ratio of the number of correctly classified samples to the total number of samples for a given test data set. In tables, we present two values: Training which represent results achieved on training CIC-AWS-2018 Dataset and Test which represent result on the test dataset, described above.

Table 3. Evaluation of Machine Learning methods for DoS attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.99	1	0.99	0.51	0.94	0.02
TN	0.99	0.01	0.99	0.91	0.99	0.97
Accuracy	0.99	0.97	0.99	0.52	0.97	0.02

Table 4. Evaluation of Machine Learning methods for Botnet attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.99	0	0.99	0.03	0.8	0.01
TN	0.99	1	0.99	0.86	0.89	0.75
Accuracy	0.99	0.02	0.99	0.04	0.8	0.02

Table 5. Evaluation of Machine Learning methods for Brute force attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.8	-	0.8	-	1	-
TN	1	-	1	-	1	-
Accuracy	0.99	-	0.99	-	1	-

We didn`t manage to find logs with similar attack type to test the model on, so only results of train dataset are presented.

Table 6. Evaluation of Machine Learning methods for DDoS attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	1	0.31	1	0.31	1	0.32
TN	0.99	0.98	0.99	0.98	0.99	0.99
Accuracy	0.99	0.5	0.99	0.5	0.99	0.5

Table 7. Evaluation of Machine Learning methods for Web Application attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	0.8	0.06	0.8	0.1	0.46	0
TN	1	0.96	0.99	0.95	0.99	0.99
Accuracy	0.99	0.07	0.99	0.11	0.99	0.06

Table 8. Evaluation of Machine Learning methods for Infiltration attack

	Random forest		Decision Tree		LinearSVC	
	Train	Test	Train	Test	Train	Test
TP	1	0.2	1	0.2	0.99	0.1
TN	1	1	1	1	0.34	0.2
Accuracy	1	0.90	1	0.90	0.78	0.16

5. Conclusion

Our research examines three common machine learning classifications models trained on the CIC-AWS-2018 Dataset and tested on datasets that we extracted from .pcap logs created by Stratosphere Lab. During the research, we found out that logs

created with the same attack programs but with different parameters or with some gap in time can have very different values of features. That is why the results on test data is appeared to be worse comparing to train data. To improve them we normalized test dataset values of attributes. Usually, it gave from 10 to 20 percent of improvement.

Although Decision tree showed the best performance and may be used in some situations. We think that trained models are far from ready to use in real-life situations. Much is left to do in the future, for example, finding better way to preprocess and normalize dataset, improvement of developed models and testing new algorithms in order to fit better the statistical data, testing on more types of intrusions dynamically.

REFERENCE

1. AFTERGOOD S.: Cybersecurity: The cold war online, *Nature*, vol. 547, pp. 30-31, Jul. 2017.
2. SCARFONE K, MELL P.: Guide to intrusion detection and prevention systems (IDPS). NIST Special Publication 2007;800(2007):94.
3. ANTTI JUVONEN, TUOMO SIPOLA: Anomaly Detection Framework Using Rule Extraction for Efficient Intrusion Detection, 2014
4. SOMMER R., PAXSON V.: Outside the closed world: On using machine learning for network intrusion detection, in: 2010 IEEE symposium on security and privacy, IEEE, 2010, pp. 305–316.
5. Internet service Registry of Open Data on AWS: *https://registry.opendata.aws/cse-cic-ids2018/*
6. LASHKARI A. H., DRAPER-GIL G., MAMUN ,M.S.I., GHORBANI A.A.: Characterization of Tor Traffic Using Time Based Features, In the proceeding of the 3rd International Conference on Information System Security and Privacy, SCITEPRESS, Porto, Portugal, 2017
7. DRAPPER-GIL G., LASHKARI A.H., MAMUN M., GHORBANI A.A.: Characterization of Encrypted and VPN Traffic Using Time-Related Features, In Proceedings of the 2nd International Conference on Information Systems Security and Privacy(ICISSP 2016) , pages 407-414, Rome , Italy
8. Internet service The Heartbleed Bug: *http://heartbleed.com/*
9. Internet service University of New Brunswick: *https://www.unb.ca/cic/datasets/ids-2018.html*
10. QIANRU ZHOU, PEZAROS D.: Evaluation of Machine Learning Classifiers for Zero-Day Intrusion Detection – An Analysis on CIC-AWS-2018 dataset
11. Internet service Stratosphere Lab: *https://www.stratosphereips.org/datasets-overview*
12. APRUZZESE G., COLAJANNI M., FERRETTI L., GUIDO A., MARCHETTI M.: On the Effectiveness of Machine and Deep Learning for Cyber Security
13. Internet service Scikit-learn: *https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html*
14. Internet service Scikit-learn: *https://scikit-learn.org/stable/index.html*
15. Internet service Pandas 0.25.3 documentation: *https://pandas.pydata.org/pandas-docs/stable/index.html*

Додаток Б – Лістинг файлу model_train.py

```
import json
import os

import joblib
import numpy as np
import pandas as pd
import plotly.express as px
from imblearn.over_sampling import SMOTE
from imblearn.under_sampling import RandomUnderSampler
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import GridSearchCV,
RandomSearchCV

DATA_DIR = './data/Bot-IoT/CSV/Traning and Testing Tets (5%
of the entier dataset)'
BEST_FEAT = os.path.join(DATA_DIR, '10-best features', '10-
best Training-Testing split')

RANDOM_STATE = 9


def preprocess_df(df: pd.DataFrame, proto_to_id = None,
transform_state = None, drop: bool = True) -> pd.DataFrame:
    """Preprocess dataframe

    Args:
        df (pd.DataFrame): input dataframe
        proto_to_id (dict, optional): dictionary to replace
protocols with idx. Defaults to None.
        transform_state (bool, optional): Whether to
transform the state column. Defaults to None.
        drop (bool, optional): whether to drop unused
columns. Defaults to True.

    Returns:
        pd.DataFrame: Output dataframe
    """
    processed_df = df.copy()

    if not proto_to_id:
        id_to_proto =
processed_df['proto'].value_counts().reset_index()['index'].
to_dict()
        proto_to_id = {value: key for key, value in
id_to_proto.items()}

    if transform_state:
        state_to_id = {'RST': 1, 'CON': 2, 'REQ': 3}
```

```
        processed_df['state_number'] =
processed_df['state'].apply(lambda item:
state_to_id.get(item, -1))

    processed_df['proto_num'] =
processed_df['proto'].apply(lambda item:
proto_to_id.get(item, -1))

    # Fir port invalid formats
    processed_df['sport'] =
processed_df['sport'].apply(lambda item: item if
isinstance(item, int) else int(item, 16))
    processed_df['dport'] =
processed_df['dport'].apply(lambda item: item if
isinstance(item, int) else int(item, 16))

    if drop:
        processed_df = processed_df.drop(['saddr', 'daddr',
'category', 'subcategory', 'proto'], axis=1,
errors='ignore')

    return processed_df, proto_to_id

proto_to_id = {'udp': 0, 'tcp': 1, 'icmp': 2, 'arp': 3,
'ipv6-icmp': 4}

training_df = pd.read_csv(os.path.join(BEST_FEAT,
'UNSW_2018_IoT_Botnet_Final_10_best_Training.csv'),
index_col=0)
testing_df = pd.read_csv(os.path.join(BEST_FEAT,
'UNSW_2018_IoT_Botnet_Final_10_best_Testing.csv'),
index_col=0)

processed_training_df, __ = preprocess_df(training_df,
proto_to_id=proto_to_id)
processed_testing_df, __ = preprocess_df(testing_df,
proto_to_id=proto_to_id)

# under/oversampling
sampled_training_df, __ = preprocess_df(training_df,
proto_to_id=proto_to_id, drop=False)

sampled_training_df = sampled_training_df.drop(['saddr',
'daddr', 'subcategory', 'proto'], axis=1)

X, y = RandomUnderSampler(
    sampling_strategy={
        "DDoS": 10000,
        "DoS": 10000,
        "Reconnaissance": 10000,
        "Normal": 370,
        "Theft": 65
```

```
    },
    random_state=RANDOM_STATE
).fit_resample(sampled_training_df.drop('category', axis=1),
sampled_training_df['category'])

sampled_training_df, y = SMOTE(
    sampling_strategy={
        "DDoS": 10_000,
        "DoS": 10_000,
        "Reconnaissance": 10_000,
        "Normal": 40_000,
        "Theft": 10_000
    },
    random_state=RANDOM_STATE
).fit_resample(X, y)

# Train and evaluate RF for the unbalanced dataset

rf_clf = RandomForestClassifier(n_estimators=500,
min_samples_leaf=1, min_samples_split=2, max_depth=50,
random_state=0)
rf_clf.fit(processed_training_df.drop('attack', axis=1),
processed_training_df['attack'])

processed_testing_df['predicted'] =
rf_clf.predict(processed_testing_df[processed_training_df.dr
op('attack', axis=1).columns])
processed_testing_df['predicted_proba'] =
[json.dumps(list(item)) for item in
rf_clf.predict_proba(processed_testing_df[processed_training
_df.drop('attack', axis=1).columns])]

print('Unbalanced dataset training results:')

print(f'accuracy_score:
{metrics.accuracy_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'precision_score:
{metrics.precision_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'recall_score:
{metrics.recall_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'roc_auc_score:
{metrics.roc_auc_score(processed_testing_df["attack"],
processed_testing_df["predicted_proba"].apply(lambda item:
json.loads(item)[1]))}')

print(metrics.confusion_matrix(processed_testing_df["attack"
], processed_testing_df["predicted"]))

# Save model
```

```
joblib.dump(rf_clf, 'rf_clf.joblib')

# Train and evaluate RF for the sampled dataset

rf_clf_balanced = RandomForestClassifier(n_estimators=500,
min_samples_leaf=1, min_samples_split=2, max_depth=50,
random_state=0)

rf_clf_balanced.fit(sampled_training_df.drop('attack',
axis=1), sampled_training_df['attack'])

processed_testing_df['predicted'] =
rf_clf_balanced.predict(processed_testing_df[processed_train
ing_df.drop('attack', axis=1).columns])
processed_testing_df['predicted_proba'] =
[json.dumps(list(item)) for item in
rf_clf_balanced.predict_proba(processed_testing_df[processed
_training_df.drop('attack', axis=1).columns])]

print('Balanced dataset training results:')

print(f'accuracy_score:
{metrics.accuracy_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'precision_score:
{metrics.precision_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'recall_score:
{metrics.recall_score(processed_testing_df["attack"],
processed_testing_df["predicted"])}')
print(f'roc_auc_score:
{metrics.roc_auc_score(processed_testing_df["attack"],
processed_testing_df["predicted_proba"].apply(lambda item:
json.loads(item)[1]))}')

print(metrics.confusion_matrix(processed_testing_df["attack"
], processed_testing_df["predicted"]))

# Save model
   joblib.dump(rf_clf_balanced, 'rf_clf_balanced.joblib')
```

Додаток В – Лістинг файлу realtime_IDS.py

```
import glob
import os
import subprocess
import time
from concurrent.futures import ProcessPoolExecutor

import joblib
import pandas as pd
import pyshark
from sklearn.ensemble import RandomForestClassifier


def record_traffic(output_filename: str, timeout: int = 10)
-> None:
    """Record a `timeout` seconds of trafic to a
output_filename

    Args:
        output_filename (str): name of file to output to
        timeout (int, optional): time in seconds that the
recording will last. Defaults to 10.
    """
    capture = pyshark.LiveCapture(interface="eth0",
output_file=output_filename)
    capture.sniff(timeout=timeout)


def record_traffic_realtime():
    """Run a recording routine in a endless loop"""
    while True:

        start_time = time.time()
        print(f'{start_time} - start traffic recording')

        filename = f'{start_time}-record.pcap'
        record_traffic(output_filename=f'./tmp/{filename}',
timeout=60)

        # Move file for the other thread to pick up
        os.rename(os.path.join(os.getcwd(),
f'tmp/{filename}'), os.path.join(os.getcwd(),
f'raw_records/{filename}'))

        print(f'{time.time()} - finish traffic recording')


def pcap_to_argus(input_filename: str, output_filename: str)
-> None:
    """Transform pcap file into argus file

    Args:
```

```
        input_filename (str): input filename
        output_filename (str): output filename
    """
    process = subprocess.Popen(
        # ['argus', '-r', os.path.join(os.getcwd(),
input_filename).replace(' ', '\ '), '-w', output_filename],
        ['argus', '-r', input_filename, '-w',
output_filename],
        stdout=subprocess.PIPE,
        stderr=subprocess.PIPE,
        start_new_session=True,
        cwd=os.getcwd()
    )
    stdout, stderr = process.communicate()

    # Remove the input filename
    os.remove(input_filename)


def argus_to_csv(input_filename: str, output_filename: str)
-> None:
    """Tranform argus file into csv

    Args:
        input_filename (str): input filename
        output_filename (str): output filename
    """
    process = subprocess.Popen(
        [
            'ra',
            '-r', input_filename,
            '-c', ',',
            '-s', '+stddev', '-s', '+min', '-s', '+max', '-
s', '+mean', '-s', '+drate', '-s', '+srate', '-s', '+seq'
        ],
        stdout=subprocess.PIPE,
        stderr=subprocess.PIPE,
    )
    stdout, stderr = process.communicate() # get the csv
records into console output

    # write into an output filename the console output
    with open(output_filename, 'w') as file:
        file.write(stdout.decode())


def process_port(item):
    """Process port: replace port names with numbers and fix
invalid formats

    Args:
        item: port
```

```
    Returns:
        int: port number
    """
    try:
        return int(item, 16)
    except (TypeError, ValueError):
        portname_to_port = {'mdns': 5353, 'https': 443,
'http': 80}
        return portname_to_port.get(item, -1)


def preprocess_df(df: pd.DataFrame, proto_to_id = None,
transform_state = None, drop: bool = True) -> pd.DataFrame:
    """Preprocess dataframe

    Args:
        df (pd.DataFrame): input dataframe
        proto_to_id (dict, optional): dictionary to replace
protocols with idx. Defaults to None.
        transform_state (bool, optional): Whether to
transform the state column. Defaults to None.
        drop (bool, optional): whether to drop unused
columns. Defaults to True.

    Returns:
        pd.DataFrame: Output dataframe
    """
    processed_df = df.copy()

    if not proto_to_id:
        id_to_proto =
processed_df['proto'].value_counts().reset_index()['index'].
to_dict()
        proto_to_id = {value: key for key, value in
id_to_proto.items()}

    if transform_state:
        state_to_id = {'RST': 1, 'CON': 2, 'REQ': 3}
        processed_df['state_number'] =
processed_df['state'].apply(lambda item:
state_to_id.get(item, -1))

    processed_df['proto_num'] =
processed_df['proto'].apply(lambda item:
proto_to_id.get(item, -1))

    # Fir port invalid formats
    processed_df['sport'] =
processed_df['sport'].apply(lambda item: item if
isinstance(item, int) else int(item, 16))
```

```
    processed_df['dport'] =
processed_df['dport'].apply(lambda item: item if
isinstance(item, int) else int(item, 16))

    if drop:
        processed_df = processed_df.drop(['saddr', 'daddr',
'category', 'subcategory', 'proto'], axis=1,
errors='ignore')

    return processed_df, proto_to_id


def preprocess_csv_file(input_file) -> pd.DataFrame:
    """Open and preprocess csv file

    Args:
        input_file (str): path to input file

    Returns:
        pd.DataFrame: preprocessed dataframe
    """
    df = pd.read_csv(input_file, encoding='ascii')

    initial_column_list = ['Proto', 'SrcAddr', 'Sport',
'DstAddr', 'Dport', 'Seq', 'StdDev', 'Min', 'State', 'Mean',
'DstRate', 'SrcRate', 'Max']
    df = df[initial_column_list]

    df = df.rename(
        columns={
            'Proto': 'proto',
            'SrcAddr': 'saddr',
            'Sport': 'sport',
            'DstAddr': 'daddr',
            'Dport': 'dport',
            'Seq': 'seq',
            'StdDev': 'stddev',
            'Min': 'min',
            'State': 'state',
            'Mean': 'mean',
            'DstRate': 'drate',
            'SrcRate': 'srate',
            'Max': 'max'
        }
    )

    df['N_IN_Conn_P_SrcIP'] =
df['saddr'].astype('category').cat.codes.rolling(100,
min_periods=1).apply(
        lambda items: (items == items[-1]).sum(), raw=True
    )
```

```
    df['N_IN_Conn_P_DstIP'] =
df['daddr'].astype('category').cat.codes.rolling(100,
min_periods=1).apply(
        lambda items: (items == items[-1]).sum(), raw=True
    )

    df['sport'] = df['sport'].apply(process_port)
    df['dport'] = df['dport'].apply(process_port)

    df = df.fillna(-1)

    processed_df, __ = preprocess_df(df, proto_to_id={'udp':
0, 'tcp': 1, 'icmp': 2, 'arp': 3, 'ipv6-icmp': 4},
transform_state=True, drop=False)

    output_columns = [
        'sport', 'dport', 'seq', 'stddev',
'N_IN_Conn_P_SrcIP', 'min',
        'state_number', 'mean', 'N_IN_Conn_P_DstIP',
'drate', 'srate', 'max',
        'proto_num', 'saddr', 'daddr', 'proto'
    ]
    return processed_df[output_columns]

def analyze_traffic_realtime():
    """Run endless loop that analyzes traffic and prints
alerts"""

    model_file = 'rf_clf_balanced.joblib'
    rf_clf: RandomForestClassifier = joblib.load(model_file)
    print(f'Loaded model from {model_file}')

    while True:
        print('Checking for new files in raw_records')
        raw_records = glob.glob('./raw_records/*.pcap')

        if raw_records:

            for raw_file in raw_records:
                print(f'{time.time()} - started analyzing
{raw_file}')

                file_id = raw_file.split('/')[-1].split('-
')[0]
                argus_filename =
os.path.join('./argus_records', f'{file_id}-record.argus')
                csv_filename = os.path.join('./csv_records',
f'{file_id}-record.csv')

                pcap_to_argus(input_filename=raw_file,
output_filename=argus_filename)
```

```
                argus_to_csv(input_filename=argus_filename,
output_filename=csv_filename)

                df = preprocess_csv_file(csv_filename)

                df['predictions'] =
rf_clf.predict(df.drop(['saddr', 'daddr', 'proto'], axis=1,
errors='ignore'))

                possible_attacks = df[df['predictions'] ==
1]

                print()
                print(f'{time.time()} - finished analyzing
{raw_file}')
                print(f'Analyzed {len(df)} network flows')
                print(f'Found {len(possible_attacks)}
possible attacks')
                for index, row in
possible_attacks.iterrows():
                    print(f'Possible attack from
{row["saddr"]}:{row["sport"]} to
{row["daddr"]}:{row["dport"]}')
                print()

        time.sleep(10)

if __name__ == '__main__':

    with ProcessPoolExecutor(max_workers=2) as executor:
        recording_future =
executor.submit(record_traffic_realtime)
        analyzing_future =
executor.submit(analyze_traffic_realtime)

        analyzing_result = analyzing_future.result()
        if analyzing_result:
            print(analyzing_result)
```