

Факультет прикладної математики та інформатики

(повне найменування назва факультету)

кібербезпеки

(повна назва кафедри)

## Дипломна робота

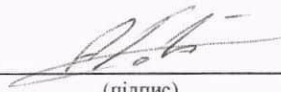
### Виявлення аномалій для систем мережевого вторгнення

Виконав: студент групи ПМК-42с

спеціальності

125 «Кібербезпеки»

(шифр і назва спеціальності)



Повторацький А.О.

(підпис)

(прізвище та ініціали)

Керівник

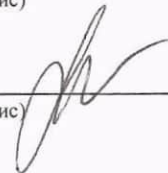


Карпюк Р.В

(підпис)

(прізвище та ініціали)

Рецензент



Клакович Л.М.

(підпис)


(прізвище та ініціали)



ЛЬВІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ  
ІМЕНІ ІВАНА ФРАНКА

Факультет Прикладної математики та інформатики  
Кафедра Кібербезпеки  
Спеціальність: 125 «Кібербезпека»  
«шифр і назва»

«ЗАТВЕРДЖУЮ»

Завідувач кафедри 

"31 "серпня 2022 року

**ЗАВДАННЯ**

на кваліфікаційну бакалаврську роботу студента

**Повторацького Андрія**

(прізвище, ім'я, по батькові)

1. **Тема роботи:** Виявлення аномалій для систем мережевого вторгнення.  
Керівник роботи Карпюк Роман \*\*\*\*\*,  
затверджені наказом університету від «13» вересня 2021 року № 15
2. **Строк подання студентом роботи** «13» червня 2023 року
3. **Вихідні дані до роботи:** \_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_
4. **Зміст пояснювальної записки (перелік питань, які потрібно розробити)**
  1. Базові питання щодо виявлення аномалій та розвитку систем
  2. Методи виявлення аномалій
  3. Дослідження системи виявлення вторгнень
  4. Розробка моделі виявлення аномалій системи мережевих вторгнень
5. **Перелік графічного матеріалу:**  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

## 6. Консультанти розділів роботи

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

7. Дата видачі завдання 31 серпня 2022 р.

## КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної роботи	Строк виконання етапів роботи	Примітка
1	Класифікація ресурсів	20.09.22	✓
2	Що таке аналітика	01.10.22	✓
3	Системи вивчення аномалій	10.10.22	✓
4	Системи IDS	01.11.22	✓
5	Машинне навчання	01.01.23	✓
6	Графи алгоритмів	01.03.23	✓
7	Функція потерь із ML	01.05.23	✓
8	Метрики якості і функціонування	01.06.23	✓
9	Оптимізація	11.06.23	✓
10			
11			
12			

Студент

(підпис)

Повторацький А.

(ініціали, прізвище)

Керівник роботи

(підпис)

Карпюк Р.\*

(ініціали, прізвище)

## ЗМІСТ

Перелік умовних скорочень.....	8
Вступ.....	9
Розділ 1. Теоретичні основи безпеки мережі.....	12
1.1. Огляд питання мережевої безпеки.....	12
1.2. Класифікація комп'ютерних атак .....	14
1.3. Системи виявлення атак та аномальної мережевої активності .....	19
Висновки до розділу 1 .....	23
Розділ 2. Мережеві аномалії та методи їх виявлення .....	25
2.1. Аномалії та їх види.....	25
2.2. Класифікація методів виявлення аномалій .....	29
2.3. Огляд інструментів виявлення аномалій.....	34
Висновки до розділу 2.....	42
Розділ 3. Дослідження алгоритмів та системи розпізнавання аномалій трафіку .....	44
3.1. Структура та архітектура системи виявлення вторгнень .....	44
3.2. Огляд підходів розпізнавання аномалій трафіку .....	50
3.3. Використання методів машинного навчання для IDS.....	56
3.4. Вибір основного алгоритму для системи розпізнавання аномалій трафіку .....	59
Висновки до розділу 3.....	64
Розділ 4. Розробка та тестування системи виявлення аномалій системи мережевих вторгнень .....	65
4.1. Середовище та інструменти розробки.....	65
4.2. Архітектура та зовнішній вигляд розробленої системи виявлення аномалій .....	69

	7
4.3. Проведення тестування алгоритму, взятого за основу розробленої системи виявлення аномалій .....	72
4.3. Проведення тестування та аналіз розробленої системи виявлення аномалій .....	75
Висновки до розділу 4.....	79
Висновки.....	81
Список використаних джерел.....	83
Додатки .....	88

**ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ**

IPS	–	система запобігання вторгненням (Intrusion Prevention System)
IDS	–	система виявлення вторгнень (Intrusion Detection System)
ІІІ (AI)	–	штучний інтелект (Artificial Intelligence)
Big Data	–	великі дані
Cloud	–	хмарні обчислення
ML	–	машинне навчання (Machine Learning)
	–	
	–	

## ВСТУП

### **Актуальність.**

На сьогодні безпека є однією з найбільших проблем майже для всіх мереж у будь-якій галузі. Існує незліченна кількість шкідливих спроб долучитися до приватної інформації, мереж та веб-служб компаній, тому різні компанії використовували різні методи для забезпечення інструментів, апаратних та програмних компонентів, таких як брандмауери, механізми шифрування та віртуальні приватні мережі для захисту особистої інформації.

Масове поширення та фактичне створення багатоелементних і багаторівневих корпоративних мереж призводить до необхідності забезпечувати безпеку й у середині мережі.

Розвиток сучасної сфери інформаційних технологій надає безліч можливостей, але водночас створює нові загрози та уразливості. Зловмисники, використовуючи комп'ютерні атаки, можуть завдати серйозного шкоди індивідам, організаціям і навіть державам.

Тема "Виявлення аномалій для систем мережевого вторгнення" є надзвичайно актуальною і має велике значення в сучасному цифровому світі. Основні обґрунтування і актуальність даної теми наступні:

Захист мереж та інформаційних систем від кіберзагроз є нагальним завданням. Існує безліч шкідливих програм, хакерських атак, фішингових схем та інших загроз, які намагаються зламати системи, викрасти конфіденційну інформацію або завдати шкоди. Виявлення аномалій для систем мережевого вторгнення дозволяє ранньо виявити незвичайні, підозрілі або небезпечні дії в мережі, забезпечуючи реагування та захист в реальному часі.

Кіберзлочинці постійно вдосконалюють свої методи атак і злому систем. Вони шукають нові шляхи та розробляють складні атаки, щоб обійти захисні механізми. Традиційні методи виявлення вторгнень, такі як сигнатурний аналіз, стають менш ефективними. Виявлення аномалій дозволяє виявити нові, раніше невідомі загрози та атаки, незалежно від їх конкретних характеристик або сигнатур.

Сучасні мережеві середовища стають все більш складними і розподіленими. Вони включають фізичні, віртуальні та хмарні ресурси, різноманітні платформи та протоколи. Це створює багато потенційних точок вразливості і збільшує ймовірність вторгнень. Виявлення аномалій дозволяє контролювати ці складні середовища, виявляти незвичайні або нормальним чином неприпустимі дії та запобігати можливим загрозам.

Багато країн встановлюють строгі законодавчі вимоги щодо захисту інформації та персональних даних. Організації повинні дотримуватися цих вимог, щоб уникнути штрафів і репутаційної шкоди. Виявлення аномалій для систем мережевого вторгнення є важливим елементом забезпечення відповідності законодавству та регуляторним вимогам щодо кібербезпеки.

Загалом, дана тема є вкрай актуальною і важливою в сучасному світі, де кіберзагрози стають все складнішими і поширенішими. Розробка та впровадження ефективних методів виявлення аномалій допоможе забезпечити безпеку мереж та захист від потенційних кібератак, зменшити ризики витоку конфіденційної інформації та підвищити рівень кібербезпеки в цілому.

**Метою** роботи є розробка і реалізація ефективного алгоритму чи системи, яка здатна виявляти незвичайні, підозрілі або шкідливі дії в мережі з метою попередження або реагування на потенційні вторгнення.

Робота спрямована на дослідження методів аналізу мережевого трафіку, розробку алгоритмів аналізу поведінки, класифікації аномалій та визначення причин їх виникнення.

Для досягнення цієї мети були сформовані наступні завдання::

1. провести огляд і аналіз існуючих технологій та методів виявлення аномалій систем мережевого вторгнення;
2. проаналізувати архітектури систем виявлення вторгнень;
3. розробити алгоритм виявлення аномалій систем мережевого вторгнення з використанням методів машинного навчання;
4. розробити та протестувати програмний модуль виявлення аномалій системи мережевих вторгнень.

Отже, метою роботи є створення ефективного і надійного засобу виявлення аномалій для систем мережевого вторгнення, що допоможе покращити безпеку мереж та запобігти потенційним кібератакам.

**Об'єктом** дослідження є аномалії для систем мережевого вторгнення.

# РОЗДІЛ 1.

## ТЕОРЕТИЧНІ ОСНОВИ БЕЗПЕКИ МЕРЕЖІ

### 1.1. Огляд питання мережевої безпеки

Безпека мережі - це широкий термін, який охоплює безліч технологій, пристроїв і процесів. Простіше кажучи, це набір правил і конфігурацій, призначених для захисту цілісності, конфіденційності та доступності комп'ютерних мереж і даних за допомогою програмних і апаратних технологій.

Сучасна мережева архітектура є складною і стикається з середовищем загроз, яке постійно змінюється, і зловмисниками, які завжди намагаються знайти та використати вразливі місця. Ці вразливості можуть існувати в багатьох областях, включаючи пристрої, дані, програми, користувачів і місця розташування. З цієї причини сьогодні використовується багато інструментів і програм керування безпекою мережі, які вирішують окремі загрози та експлойти, а також невідповідність нормативним вимогам [1].

Розглянемо як працює безпека мережі. Вирішуючи питання безпеки мережі в організації, слід враховувати багато рівнів. Атаки можуть відбуватися на будь-якому рівні в моделі рівнів безпеки мережі, тому обладнання, програмне забезпечення та політика безпеки мережі мають бути розроблені для кожної області.

Безпека мережі зазвичай складається з трьох різних елементів контролю: фізичного, технічного та адміністративного, що показано на рис. 1.1.

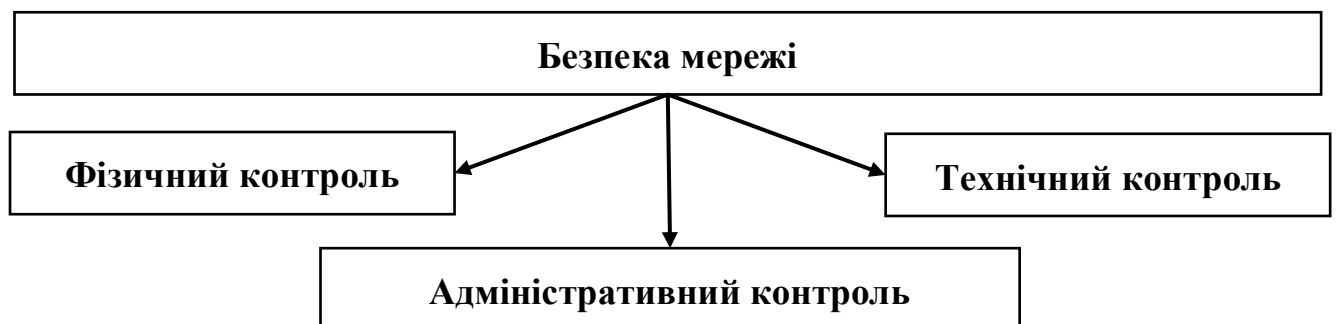


Рис. 1.1. Елементи контролю безпеки мережі

Нижче наведено короткий опис різних типів мережевої безпеки та принципів роботи кожного елемента керування.

Фізична безпека мережі. Засоби контролю фізичної безпеки призначені для запобігання неавторизованому персоналу отримати фізичний доступ до компонентів мережі, таких як маршрутизатори, кабельні шафи тощо. Контрольований доступ, такий як замки, біометрична аутентифікація та інші пристрої, є важливим у будь-якій організації [2].

Технічна мережева безпека. Засоби технічного контролю безпеки захищають дані, які зберігаються в мережі або передаються через мережу, в мережу або з неї. Захист подвійний; йому потрібно захищати дані та системи від неавторизованого персоналу, а також захищати від зловмисних дій співробітників.

Адміністративна мережева безпека. Адміністративні засоби контролю безпеки складаються з політик безпеки та процесів, які контролюють поведінку користувачів, зокрема спосіб автентифікації користувачів, їхній рівень доступу, а також те, як співробітники ІТ вносять зміни в інфраструктуру.

Безпека є однією з найбільших проблем у сучасних мережах, і це стосується майже всіх галузей діяльності. Зростання кількості підступних атак та шкідливих програм вимагає від підприємств приділяти особливу увагу забезпеченню безпеки своїх мереж і захисту конфіденційної інформації.

Ці заходи разом з іншими методами та підходами до забезпечення безпеки мережі допомагають компаніям захищати свою інформацію та забезпечувати безпеку у мережевому середовищі. Проте, зрозуміло, що безпека є постійним викликом, і вона повинна оновлюватися та пристосовуватися до швидко змінюючого ландшафту загроз і технологій.

Локальні обчислювальні мережі сучасних підприємств наповнені різними додатками та вирішують безліч завдань. Поведінка мережі формується користувачами, сервісами програмно-апаратних засобів, іншими мережевими пристроями. Під нормальним функціонуванням мережі ми маємо на увазі наступне: гарантоване надання сервісів та стійкість до різних стресових впливів,

які у загальному розгляді можна розділити на мимовільні (відмова обладнання, помилки у програмах) та довільні (цілеспрямовані атаки).

## **1.2. Класифікація комп'ютерних атак**

Багато людей покладаються на Інтернет для багатьох своїх професійних, соціальних та особистих видів діяльності. Але є також люди, які намагаються пошкодити наші комп'ютери, підключені до Інтернету, порушують нашу конфіденційність і виводять з ладу Інтернет-послуги.

Враховуючи частоту та різноманітність існуючих атак, а також загрозу нових і більш руйнівних атак у майбутньому, безпека мережі стала центральною темою в області комп'ютерних мереж [3].

Наскільки вразливі комп'ютерні мережі? Які типи атак є найбільш поширеними сьогодні?

Шкідливе програмне забезпечення - скорочення від шкідливого програмного забезпечення, спеціально розробленого для порушення, пошкодження або отримання авторизованого доступу до комп'ютерної системи. Значна частина зловмисного програмного забезпечення, яке існує сьогодні, саморозмножується: як тільки воно заражає один хост, з цього хоста воно шукає доступу до інших хостів через Інтернет, а від нещодавно інфікованих хостів воно шукає доступу до ще інших хостів. Таким чином, зловмисне програмне забезпечення, що самовідтворюється, може поширюватися експоненційно швидко.

Ефективний захист від потенційних мережових атак неможливий без їх детальної класифікації, що полегшує їх виявлення та завдання протидії їм. В даний час відома велика кількість різних типів класифікаційних ознак. В якості таких ознак може бути обрано, наприклад, поділ на пасивні та активні, зовнішні та внутрішні атаки, навмисні та ненавмисні тощо. На жаль, незважаючи на те, що деякі з існуючих класифікацій мало застосовні на практиці, їх активно використовують при виборі СВА та їх експлуатації.

Одну з існуючих класифікацій було запропоновано Пітером Меллом [4], у якій він всі можливі мережові атаки на типи, які представлено на рис. 1.2.

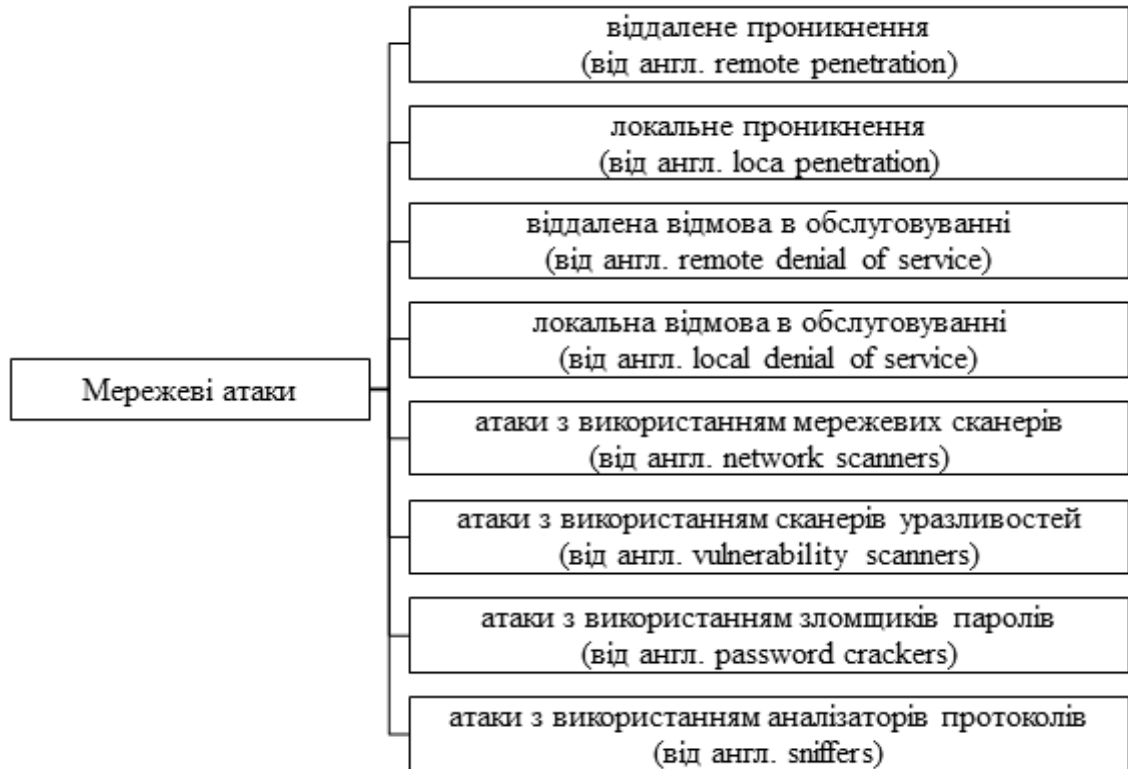


Рис. 1.2. Класифікація мережевих атак за П.Меллом

Наведена класифікація є досить повною з практичної точки зору, оскільки вона охоплює майже всі можливі дії зловмисника. Однак для протидії мережним атакам цього недостатньо, так як її використання в даному вигляді не дозволяє визначати елементи мережі, схильні до впливу тієї чи іншої атаки, а також наслідки, до яких може призвести успішна реалізація атак. У такому разі ми не включаємо в аналіз найважливіший компонент, а саме — модель загроз для безпеки, з побудови якої повинні розпочинатися всі заходи щодо забезпечення захисту інформації.

Аналогічним недоліком страждає і компактніша класифікація, що запропонована компанією Internet Security Systems, Inc., у якій міститься лише п'ять типів атак (рис. 1.3).

У своїх продуктах, призначених для захисту мереж, серверів і робочих станцій (наприклад, Real Secure, System scanner та ін.) компанія Internet Security Systems використовує кілька інших класифікаційних ознак можливих мережевих атак, вони більш ефективні з точки зору захисту від вторгнень (рис. 1.4).

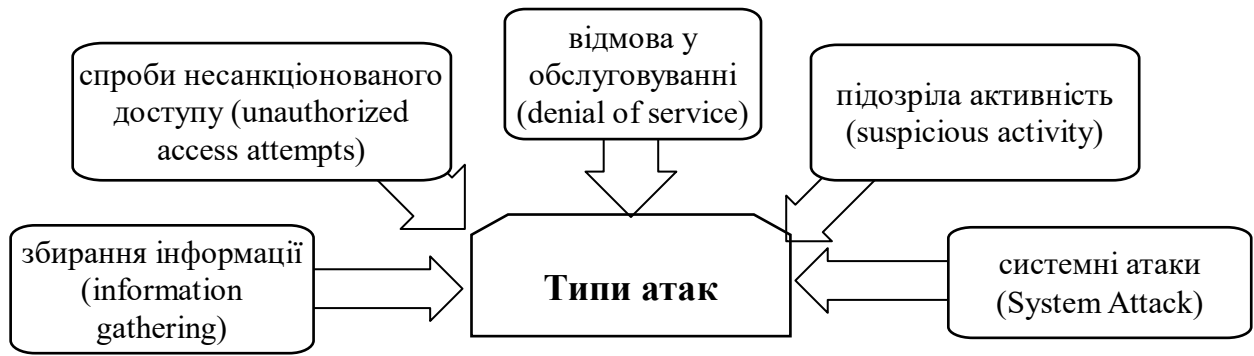


Рис. 1.3. Класифікація мережевих атак, яка запропонована компанією Internet Security Systems, Inc.

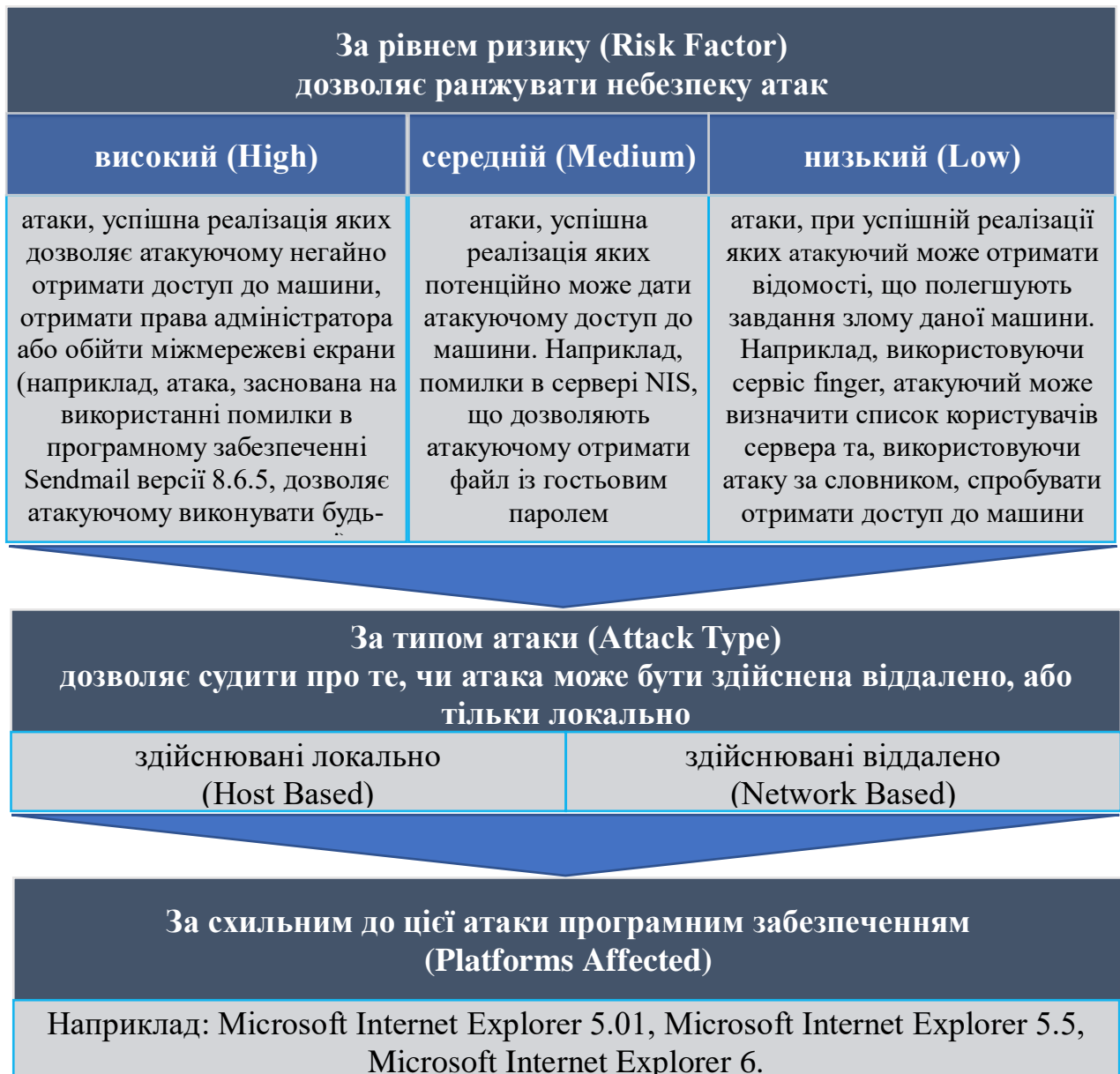


Рис. 1.4. Класифікаційні ознаки можливих мережевих атак (компанія Internet Security Systems)

Ще одним варіантом класифікації мережесих атак є поділ їх за характером дій, що використовуються в атаці. Така класифікація наведена на рис. 1.5.

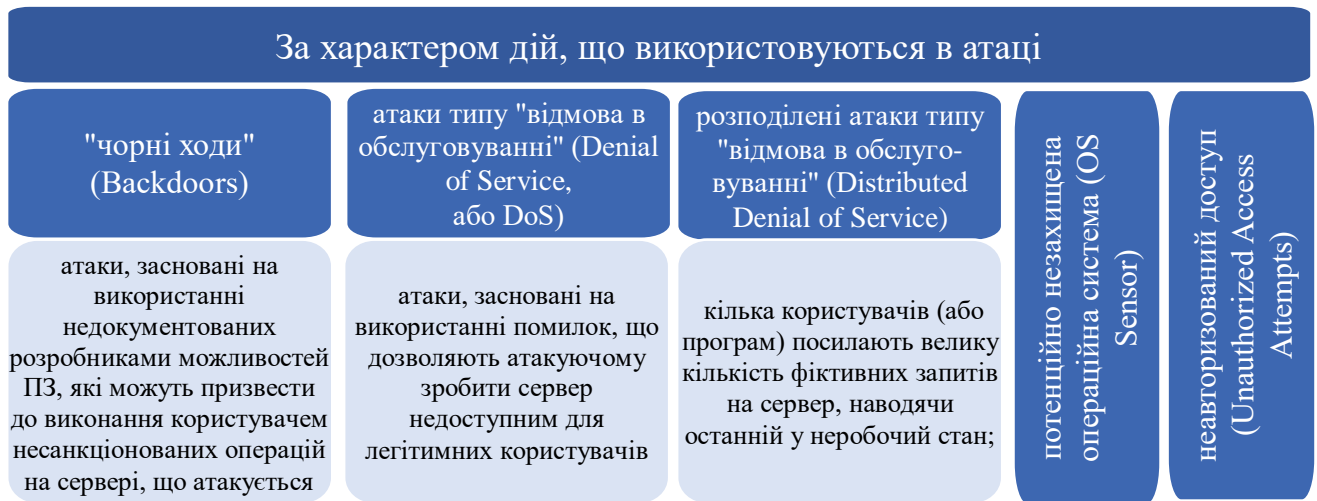


Рис. 1.5. Класифікаційні за характером дій

До недоліків наведених класифікаційних ознак можна віднести те, що вони не дозволяють описати мету атаки, а також її наслідки. Наприклад, класифікаційна ознака "за характером дій" містить два класи атак типу "відмова в обслуговуванні", але в той же час не містить класів, що описують атаки, спрямовані на перехоплення трафіку.

Інший підхід був застосований у класифікації, використаній у досить відомому програмному продукті Nessus, призначеному для аналізу безпеки серверів. Тут використовується класифікація "за характером уразливості", що використовується для реалізації різних атак (рис. 1.6).

Крім того, за типом програмного середовища вони поділяються на уразливості в операційній системі, уразливості в певному сервісі та вразливості в певному програмному забезпеченні. Для визначення вразливості в операційній системі використовується параметр Host/OS, вразливості в конкретних сервісах і в певному програмному забезпеченні класифікуються за групами.

У цій класифікації більш детально, порівняно з попередніми, опрацьовано атаки, що використовують уразливості в системному, прикладному та мережевому програмному забезпеченні.



Рис. 1.6. Класифікація уразливостей, що використовується у програмному продукті Nessus.

Однак ця класифікація не охоплює всіх наявних мережєвих атак, за межами розгляду залишаються такі небезпечні атаки, як атаки типу "відмова в обслуговуванні", перехоплення даних і атаки, спрямовані на мережєве обладнання.

Позитивною рисою цієї класифікації є наявність класу "інші помилки, що не увійшли до інших категорій", оскільки формально до будь-якої атаки, зокрема нової, завдяки цьому класу буде застосовна ця класифікація. Однак, з іншого боку, цей клас даремний, оскільки не містить жодної додаткової інформації.

Як видно з описаних класифікацій, далеко не всі вони є повними. У деяких випадках під виглядом єдиної класифікації робиться спроба поєднати кілька класифікацій, проведених за різними характеристиками.

Поява нових атак призводить до зниження ефективності застосування існуючих класифікацій, тому їх використання без внесення змін неможливо. Ця ситуація пояснюється величезною кількістю різних мережесих атак і постійною появою нових атак, деякі з яких не підкоряються критеріям існуючих класифікацій.

Отже, застосування існуючих класифікацій не можна назвати раціональним. Існує об'єктивна необхідність у створенні нової гнучкої класифікаційної схеми можливих мережесих атак, побудованої з урахуванням зазначених вище недоліків.

Аналіз інформаційних джерел дозволив зазначити лише кілька прикладів загроз та атак, з якими може зіткнутися мережа організації. Ці атаки можуть мати різні цілі, включаючи отримання конфіденційної інформації, відключення мережі або пошкодження репутації організації. Підтримка безпеки мережі вимагає впровадження відповідних заходів і захисту, щоб запобігти цим загрозам і реагувати на них, якщо вони все ж стануться.

### **1.3. Системи виявлення атак та аномальної мережесих активності**

Більшість сучасних підходів до процесу виявлення атак використовують деяку форму аналізу на основі правил [5]. Цей аналіз спирається на наборі заздалегідь визначених правил, які надаються адміністратором, автоматично створюються системою або використовуються обидва варіанти. Експертні системи представляють найпоширенішу форму підходів до виявлення атак на основі правил. Початкові зусилля з дослідження виявлення атак показали неефективність будь-якого підходу, який вимагає ручного перегляду журналу реєстрації подій. Інформація, необхідна для ідентифікації атак, представлена у великій кількості даних аудиту, і ефективний огляд цих даних вимагає використання автоматичної системи для їх аналізу.

Експертна система складається з набору правил, які охоплюють знання людини-експерта. Ці правила використовуються системою для отримання інформації про стан захисту з даних, отриманих від системи виявлення атак. Експертні системи дозволяють агрегувати величезний досвід людини в

комп'ютерному додатку, який потім використовує ці знання для виявлення дій, які відповідають певним характеристикам зловживання або нападу.

При цьому експертні системи вимагають постійного оновлення, щоб залишатися актуальними. У той час як експертні системи пропонують хорошу можливість перегляду даних в журналах, необхідні оновлення можуть бути або проігноровані, або виконані вручну (адміністратором). Як мінімум, це призведе до появи експертної системи з недостатніми (ослабленими) можливостями. У гіршому випадку відсутність обслуговування знизить ступінь захищеності всієї мережі, вводячи користувачів в оману, що мережа безпечна.

Системи, засновані на правилах, страждають від нездатності виявити сценарії атаки, які можуть відбуватися протягом тривалого періоду часу. Хоча система може виявляти окремі приклади підозрілої активності, про них можна не повідомляти, якщо вони відбуваються ізольовано один від одного. Сценарії атаки, в яких кілька хакерів працюють узгоджено, також важко виявити за допомогою цих методів, оскільки вони не фокусуються на стані змін (переходів) в атаці, а замість цього звертають увагу на наявність (присутність) окремих елементів. Будь-який поділ атаки, або з часом, або між декількома, очевидно, не пов'язаними хакерами, важко виявити за допомогою цих методів.

За останні кілька років було розроблено велику кількість підходів до виявлення атак на базі неекспертних систем. Незважаючи на те, що багато хто з них виглядають досить перспективно, експертні системи залишаються найпоширенішим підходом до виявлення атак.

Системи виявлення атак, як і більшість сучасних програмних продуктів, повинні відповідати ряду вимог. Це і сучасні технології розробки, і орієнтація на особливості сучасних інформаційних мереж, і сумісність з іншими програмами. Щоб зрозуміти, як правильно використовувати системи виявлення атак (ОА), ви повинні чітко уявляти, як вони працюють і які їх вразливості. Структура системи виявлення атак наведена на рис. 1.7.

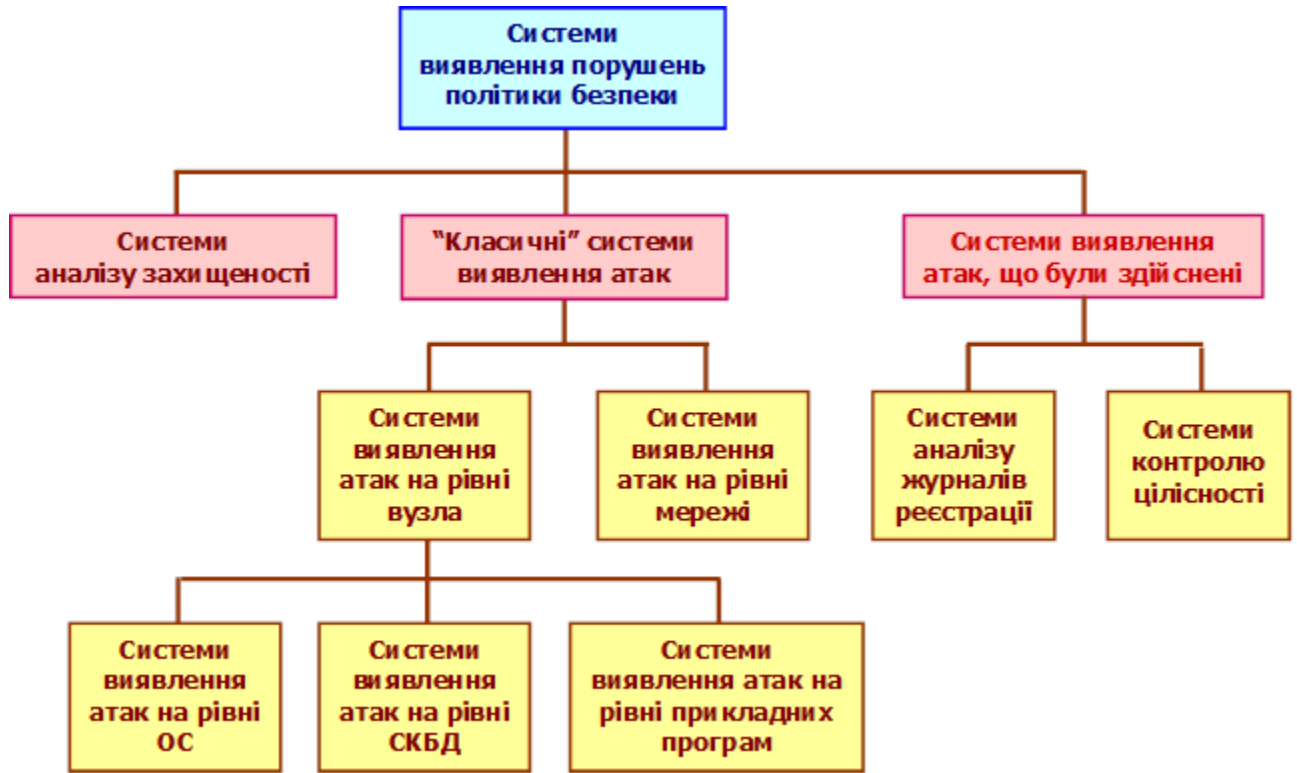


Рис. 1.7. Узагальнена класифікація СВБ

Розглянемо принципи, на яких базується ідея виявлення комп'ютерних атак [6]. Існує дві основні технології побудови СОА (рис. 1.8). Їх суть полягає в тому, що СОА володіють певним набором знань або про методи вторгнення, або про нормальну поведінку спостережуваного об'єкта.



Рис. 1.8. Методи аналізу подій, на яких базується технології побудови СОА

Технологія виявлення атак шляхом ідентифікації аномальної поведінки ґрунтується на наступній гіпотезі. Аномальна поведінка користувача (тобто атака або яесь вороже дію) часто проявляється як відхилення від нормальної поведінки.

Події при спробі вторгнення відрізняються від подій нормальної діяльності користувачів або взаємодії вузлів мережі і, відповідно, можуть бути визначені. Прикладом аномальної поведінки можуть бути велика кількість з'єднань за короткий проміжок часу, високе завантаження центрального процесора тощо.

Сенсори збирають дані про події, створюють шаблони нормальної діяльності та використовують різні метрики для визначення відхилення від нормального стану. Якщо можна було б однозначно описати профіль нормальної поведінки користувача, будь-яке відхилення від нього можна ідентифікувати як аномальну поведінку. Однак аномальна поведінка не завжди є атакою. Наприклад, одночасне надсилання великої кількості запитів від адміністратора мережі підсистема виявлення атак може ідентифікувати як атаку типу «відмова в обслуговуванні».

Технологія виявлення аномалій орієнтована виявлення нових типів атак. Проте її недолік – необхідність постійного навчання. Поки що технологія виявлення аномалій не набула широкого поширення. Пов'язано це з тим, що ця технологія важко реалізується на практиці. Проте зараз намітився певний інтерес до неї.

Системи виявлення аномальної поведінки засновані на тому, що SOA знає деякі ознаки, які характеризують правильну або прийнятну поведінку об'єкта спостереження. Під нормальною або правильною поведінкою маються на увазі дії, що виконуються об'єктом, які не суперечать політиці безпеки.

Системи виявлення зловживань засновані на тому, що SOA знає деякі ознаки, які характеризують поведінку зловмисника. Найбільш поширеною реалізацією технології виявлення шкідливої поведінки є експертні системи (наприклад, Snort, RealSecure IDS, Enterasys Advanced Dragon IDS).

На рисунку 1.9 представлені технології систем виявлення атак.

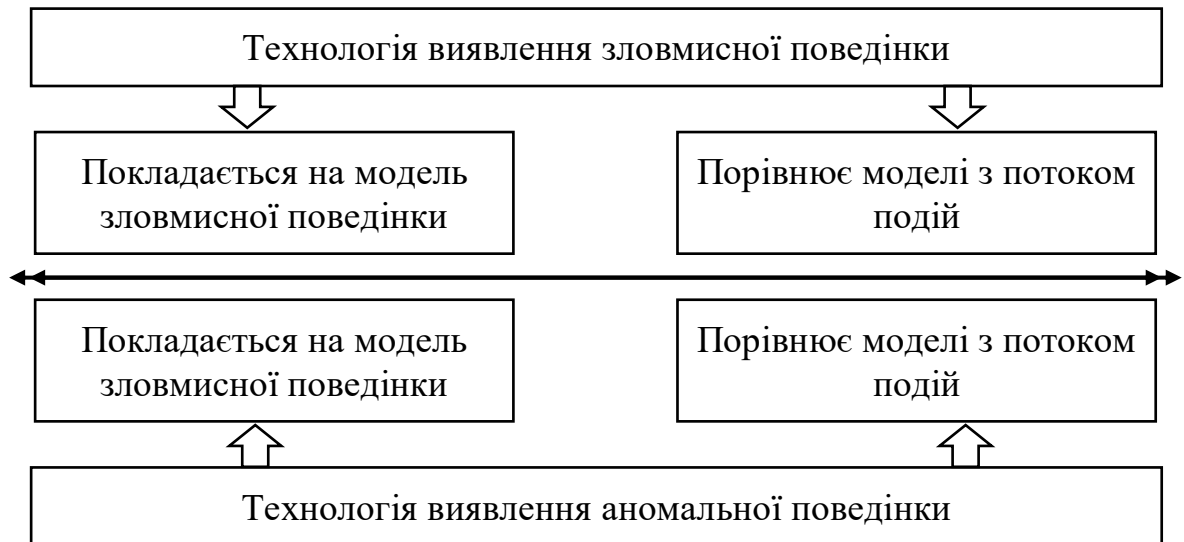


Рис. 1.9. Існуючі технології систем виявлення атак

Датчики аномалій виявляють незвичайну поведінку, аномалії функціонування окремого об'єкта - труднощі їх застосування на практиці пов'язані з нестійкістю самих об'єктів, що охороняються, і зовнішніх об'єктів, що взаємодіють з ними. Об'єктом спостереження може бути мережа в цілому, окремий комп'ютер, мережева служба (наприклад, FTP-сервер), користувач, тощо.

## Висновки до розділу 1

Розвиток сучасної сфери інформаційних технологій призводить до зростання загроз і уразливостей, які створюють можливості для здійснення комп'ютерних атак з боку зловмисників.

Мережева безпека є надзвичайно важливою складовою для забезпечення безпеки і захищеності інформаційних систем та мереж. Системи виявлення атак (Intrusion Detection Systems - IDS) є ключовим елементом в цьому процесі, оскільки вони дозволяють виявляти та реагувати на підозрілу та шкідливу активність в мережах.

Виявлення атак передбачає аналіз мережевого трафіку та інших відомостей для виявлення знаків аномальної або зловмисної діяльності. Це може включати

аналіз пакетів даних, системних журналів, а також використання сигнатурних баз даних та алгоритмів машинного навчання для виявлення відомих та нових типів атак.

Важливо мати систему виявлення атак, яка може ідентифікувати ці типи атак та вживати відповідних заходів для захисту мережі. Усунення вразливостей, моніторинг мережевого трафіку та виявлення підозрілої активності є критичними етапами у захисті мереж та інформаційних систем від атак і порушень безпеки.

## РОЗДІЛ 2.

### МЕРЕЖЕВІ АНОМАЛІЇ ТА МЕТОДИ ЇХ ВИЯВЛЕННЯ

#### 2.1. Аномалії та їх види

Виявлення і ідентифікація мережевих аномалій є важливим етапом у забезпеченні безпеки мережі організації. Цей процес дозволяє виявити незвичайну або підозрілу активність, яка може бути результатом атаки злоумисника, помилки в роботі мережі або інших аномалій.

У процесі інформатизації суспільства мережеві послуги стають все більш поширеними і використовуються у різних сферах життя. Забезпечення керованості інформаційно-обчислювальних систем, а також збереження цілісності, доступності та конфіденційності даних є важливими завданнями для адміністраторів цих систем.

У процесі інформування суспільства починають стрімко розвиватися мережеві служби і відбувається процес впровадження їх практично в усі верстви суспільства. Перед адміністраторами інформаційно-обчислювальних систем стоїть завдання забезпечення керованості цих систем, а також цілісності, доступності та конфіденційності даних. В кінцевому підсумку забезпечити нормальне функціонування системи з максимальним виключенням нестандартної поведінки системи (мережевих аномалій).

Аномалія - це відхилення від норми, від загальної закономірності, яке характеризує некоректність поведінки. Стандартним типом аномалії трафіку є вихід параметра інформативного сигналу за межі його діапазону допустимих значень як за величиною, так і за швидкістю зміни в часі [7].

«Аномалія» або «викид» відноситься до будь-якого ідентифікованого значення, яке суттєво відхиляється від типових моделей поведінки. Це може бути подія, предмет або спостереження, які не відповідають тому, що вважається нормою. На рис. 2.1 наведено діаграму мережевого трафіку, де аномалії позначені червоними точками.

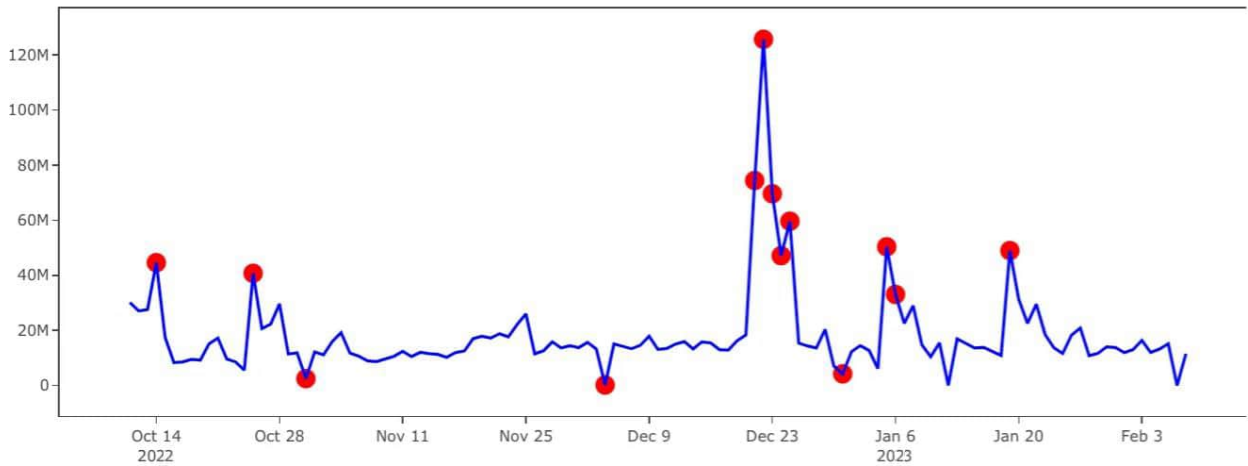


Рис. 2.1. Приклад виявлення мережових аномалій у трафіку

Завдяки збору даних, аналітиці даних і програмному забезпеченню віртуалізації даних ми покращили контроль і видимість усіх наших внутрішніх бізнес-операцій. Маючи можливість накопичувати мільйони показників із найприхованіших куточків нашого бізнесу, ми можемо створювати значні набори даних для успішного керування ефективністю нашої бізнес-діяльності у великих масштабах.

У цих наборах даних містяться шаблони даних і стандарти, які висвітлюють типові базові лінії поведінки. Ці шаблони даних переконливо вказують на те, що «бізнес як завжди». Будь-яке значення в наборі даних, яке суттєво відхиляється від цього шаблону даних або стандарту, вважатиметься аномалією, яка часто вказує на критичну або неочікувану проблему.

Наприклад, під час виявлення шахрайства з кредитними картками аномалією може бути транзакція, яка не відповідає попередній купівельній поведінці користувача. У безпеці мережі аномалія трафіку може вказувати на спробу вторгнення.

У контексті мережевої безпеки аномалії можна розділити на точкові, контекстні та колективні аномалії [8].

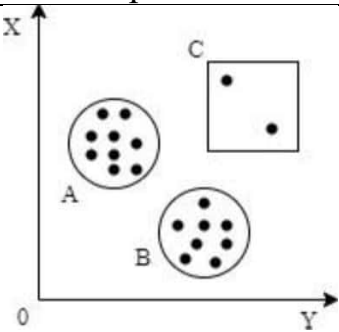
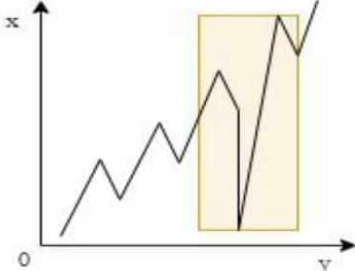
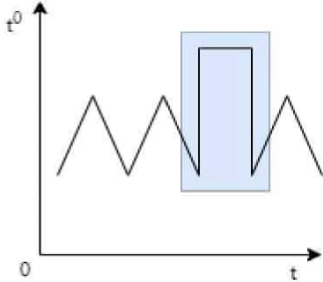
Точкові аномалії характеризуються появою одного об'єкта, який не узгоджується з іншим набором даних. Прикладом може служити ізольований екземпляр мережевого трафіку, який відрізняється від звичайних екземплярів, за певний період часу.

Контекстуальні аномалії характеризуються появою екземпляра даних, який є аномалією в цьому контексті. Контекст формується на основі структур у наборах даних. Для його написання використовуються два основних набори змінних: контекстуальний і поведінковий. Перші використовуються для визначення середовища для кожного екземпляра. Другий визначає всі інші характеристики, пов'язані з конкретним екземпляром даних.

Згідно з Chandola et al. (2009), аномалії можна класифікувати на три типи, опис яких наведено в табл. 2.1.

Таблиця 2.1

Таблиця класифікації аномалій

№	Назва	Приклад	Опис
1	Точкові аномалії		Якщо окремих екземпляр даних можна вважати аномальним по відношенню до решти даних, то такий екземпляр називається точковою аномалією. Це найпростіший тип аномалії, на якому зосереджено більшість досліджень з виявлення аномалій.
2	Контекстні аномалії		Якщо екземпляр даних є аномальним у певному контексті, але не в іншому, то його називають контекстною аномалією (в літературі також зустрічається термін "умовна аномалія").
3	Колективні аномалії	 <p>ділянка синього кольору є колективною аномалією</p>	Якщо сукупність пов'язаних екземплярів даних є аномальною по відношенню до всього набору даних, це називається колективною аномалією. Окремі екземпляри даних у колективній аномалії можуть не бути аномаліями самі по собі, але їхня поява разом як сукупності є аномальною.

Заходи та методи, які зазвичай використовуються при виявленні аномалій, включають наступне.

1. Порогові значення: спостереження за об'єктом виражаються у вигляді числових інтервалів. Вихід за межі цих інтервалів вважається аномальною поведінкою. Спостережуваними параметрами можуть бути, наприклад, наступні: кількість файлів, до яких користувач звертався в даний період часу, кількість невдалих спроб входу, завантаження процесора, тощо. Порогові значення можуть бути статичними і динамічними (тобто змінювати, адаптація до певної системи).

2. Статистичні заходи: рішення про наявність атаки приймається на підставі великого обсягу зібраних даних шляхом статистичної попередньої обробки; - параметричний: для виявлення атак будується спеціальний «профіль нормальної системи» на основі шаблонів (тобто деякої політики, якої зазвичай повинен дотримуватися даний об'єкт).

3. Непараметричний: тут профіль будується вже на основі спостереження за об'єктом в період навчання.

4. Міри (підписи) на основі правил: вони дуже схожі на непараметричні статистичні показники. У період навчання складається уявлення про нормальну поведінку об'єкта, яке фіксується у вигляді спеціальних «правил». Отримані підписи «хорошого» поведінки об'єкта.

5. Інші заходи: нейронні мережі, генетичні алгоритми, які дозволяють класифікувати певний набір ознак, видимих сенсору-сенсору. У сучасних SOA в основному використовуються перші два методи. Слід зазначити, що при використанні даної технології є дві крайності: - виявлення аномальної поведінки, яка не є атакою, і класифікація її як атаки (помилка другої ролі).

6. Пропуск атаки, яка не відповідає визначенню аномальної поведінки (помилка першого роду). Цей випадок набагато небезпечніше, ніж помилкова кваліфікація аномальної поведінки як нападу.

Тому при роботі з операційними системами цієї категорії перед рядовими користувачами і фахівцями стоять дві досить нетривіальні завдання:

- 1) побудова профілю об'єкта - складне і трудомістке завдання, що вимагає від фахівця з безпеки великої попередньої роботи, високої кваліфікації і досвіду;
- 2) визначення граничних значень характеристик поведінки суб'єкта для зниження ймовірності настання одного з двох крайніх випадків.

## **2.2. Класифікація методів виявлення аномалій**

Один із аспектів, на який звертають увагу адміністратори, є виявлення і управління мережевими аномаліями. Мережеві аномалії можуть бути ознакою нестандартного або підозрілого поведінки системи, що може бути результатом вторгнень або інших загроз для безпеки мережі.

Для забезпечення штатного функціонування системи та виключення нестандартної поведінки, адміністратори використовують системи виявлення вторгнень (IDS) та інші інструменти виявлення аномалій. Ці системи спостерігають за мережевим трафіком та аналізують його, шукаючи відхилення від очікуваної норми.

Адміністратори також можуть використовувати різні методи аналізу поведінки системи, включаючи статистичні підходи, машинне навчання та глибоке навчання, для виявлення аномалій. Ці методи дозволяють виявити незвичайні зміни в мережевому трафіку, використанні ресурсів або поведінці користувачів.

Метою цих заходів є не лише виявлення мережових аномалій, але й запобігання можливим атакам, збереження безпеки мережі та забезпечення штатного функціонування системи. Адміністратори також можуть використовувати різні стратегії реагування на аномалії, включаючи блокування вторгнів або відновлення системи до стану до аномалії.

Загалом, виявлення і управління мережевими аномаліями є важливою складовою безпеки мережевої інфраструктури, що дозволяє адміністраторам вчасно реагувати на потенційні загрози і забезпечити безпеку та надійність системи.

Метод виявлення аномалій мережевого трафіку на початковому етапі передбачає перехоплення вхідного та вихідного трафіку. При цьому

перехоплений трафік шукає заголовки IP-пакетів [9], з яких витягуються всі необхідні атрибути: обсяг IP-пакета, IP-адреси джерела, IP-адреса призначення, дата отримання IP-пакета, година отримання IP-пакета. Отримана інформація зберігається у базі даних мережевої статистики.

Наукові дослідження щодо аномалій систем мережевого вторгнення широко проводяться в галузі кібербезпеки та мережевих технологій.

Питанням застосування методів машинного навчання для виявлення аномалій в мережевому трафіку на основі аналізу поведінки мережі присвячено ряд наукових робіт [10,11].

В літературних джерелах можна знайти огляд різних методів та алгоритмів виявлення аномалій в мережевих системах, включаючи статистичні, що базуються на агрегації та машинному навчанні [12].

Отже, багато дослідницьких зусиль було присвячено розробці високоефективних методів виявлення аномалій, які були застосовані до різноманітних реальних сценаріїв, включаючи IDS [13-16], що є темою цієї роботи; виявлення шахрайства [17-19]; діагностика медичних аномалій [20-22]; виявлення несправностей у промисловому середовищі [23-26]; виявлення аномалій у міських транспортних потоках [27].

Існує багато різних досліджень на цю тему, які надають повний огляд предмету [28-30], і зокрема в області IDS [31-33]. Крім того, в останні роки зростає кількість робіт, що використовують методи DL для виявлення аномалій [34-35], а також робіт, присвячених виявленню аномалій у потоках даних [36] або розподіленим підходам в умовах виклику великих даних для боротьби з прокляттям розмірності [37].

У сфері виявлення аномалій в системах мережевого вторгнення проводиться велика кількість досліджень. Розвиток нових методів, алгоритмів та підходів продовжується з метою вдосконалення ефективності та точності виявлення аномалій та захисту мережевої інфраструктури.

Методи виявлення аномалій можна розділити на 4 великі групи: поведінкові методи, методи машинного навчання, методи обчислювального інтелекту та методи, засновані на знаннях, як показано на малюнку 2.2.

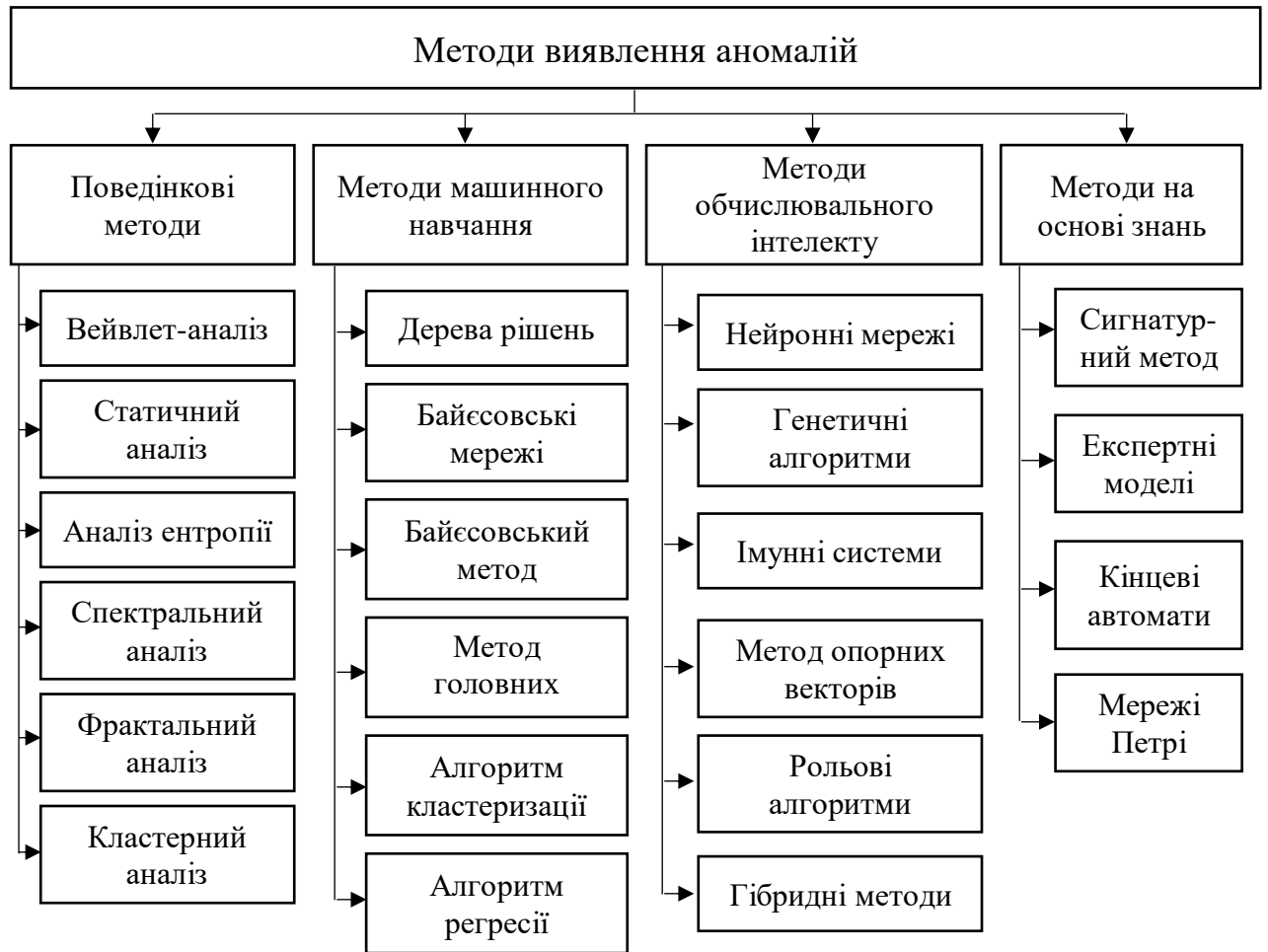


Рис. 2.2. Загальні методи виявлення аномалій

Сучасні тенденції у галузі виявлення аномалій у мережевому трафіку орієнтовані на використання розширених методів аналізу даних та машинного навчання для виявлення аномальних активностей і потенційних загроз у мережах. Деякі з найважливіших тенденцій в цій галузі включають:

1. Використання алгоритмів машинного навчання: Застосування методів машинного навчання, таких як нейронні мережі, методи кластеризації та випадкові ліси, дозволяє автоматично виявляти аномалії на основі аналізу великого обсягу даних про мережевий трафік.
2. Використання поведінкових аналітичних моделей: Виявлення аномалій може базуватися на аналізі нормального поведінки мережі та виявленні відхилень від цього шаблону. Це дозволяє виявляти нові типи загроз, які

можуть бути невідомими для традиційних сигнатурних методів виявлення загроз.

3. Аналіз потоків даних: Замість аналізування окремих пакетів даних, з'являється тенденція до аналізу потоків даних, що дозволяє збирати та аналізувати дані на рівні вищого рівня абстракції. Це дозволяє виявляти складніші аномалії та загрози.
4. Використання аналізу змісту пакетів: Для виявлення аномалій може бути використана аналітика змісту пакетів, така як аналіз заголовків пакетів, вмісту протоколів та використання специфічних сигнатур атак.
5. Використання розподілених систем виявлення аномалій: Великі компанії та організації можуть використовувати розподілені системи виявлення аномалій для збору та аналізу даних з різних джерел. Це дозволяє отримувати більш повну картину стану мережі та виявляти загрози, які можуть бути помічені тільки при аналізі даних з різних джерел.

Важливо відзначити, що сучасні аналітичні методи виявлення аномалій постійно розвиваються, і дослідження в галузі безпеки мереж продовжуються для виявлення нових типів загроз і покращення ефективності систем виявлення аномалій.

Дослідження існуючих систем мережного вторгнення є важливим кроком для забезпечення безпеки мереж та виявлення потенційних загроз і атак.

Загалом, системи мережного вторгнення є важливим інструментом для забезпечення безпеки мереж і виявлення потенційних загроз. Проте, їх ефективність залежить від налаштування, інтеграції з іншими системами та постійного оновлення, щоб виявляти нові типи атак.

Відповідно до [38], [39] всі методи виявлення аномалій складаються з наступних основних модулів чи етапів (рис. 2.3).

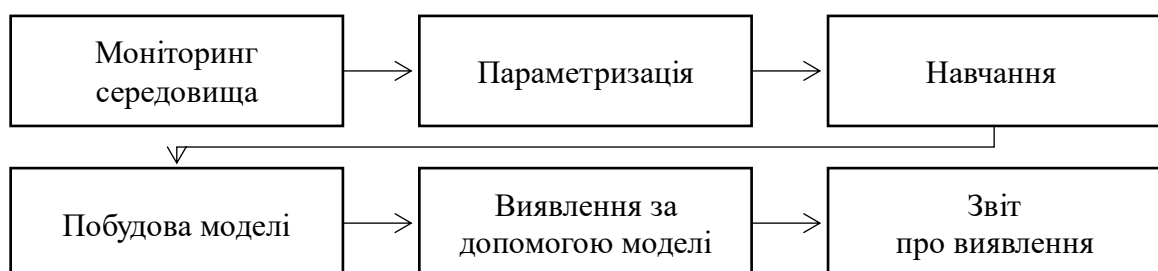


Рис. 2.3. Загальна схема виявлення аномалій

Ці етапи параметризація, навчання та виявлення. Параметризація включає збір вихідних даних з контрольованого середовища. Вихідні дані повинні бути типовими для системи, яка має бути змодельована. Етап навчання моделює систему за допомогою ручних чи автоматичних методів.

Загалом виявлення аномалій можна поділити на кілька основних компонентів (рис. 2.4):

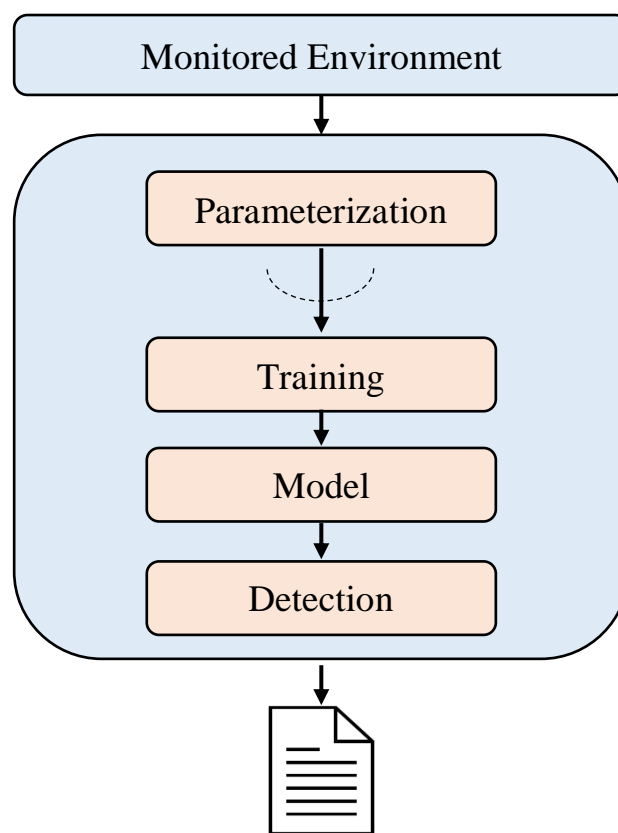


Рис. 2.4. Основні компоненти процесу виявлення аномалій

1. параметризація – дані, що відстежуються, відокремлюються від вхідних даних у формі, придатній для подальшої обробки;
2. навчання – при виборі цього режиму модель мережі (стан навчання) оновлюється. Це оновлення може виконуватись як автоматично, так і вручну;

3. виявлення - створена (навчена) модель використовується для порівняння даних із контрольованої мережі. Якщо вони відповідають певним критеріям, створюється звіт про виявлення аномалій.

Аномальний стан в мережевому трафіку (аномалії) - стан, при якому прогнозований результат в будь-який момент часу відрізняється від дійсного (істинного). Тоді аномаліями трафіку будемо називати будь-які відхилення показників мережі від заздалегідь зафіксованих.

Виявлення аномалій в системі можна описати на ступиним загальним алгоритмом, який наведено на рис. 2.5.

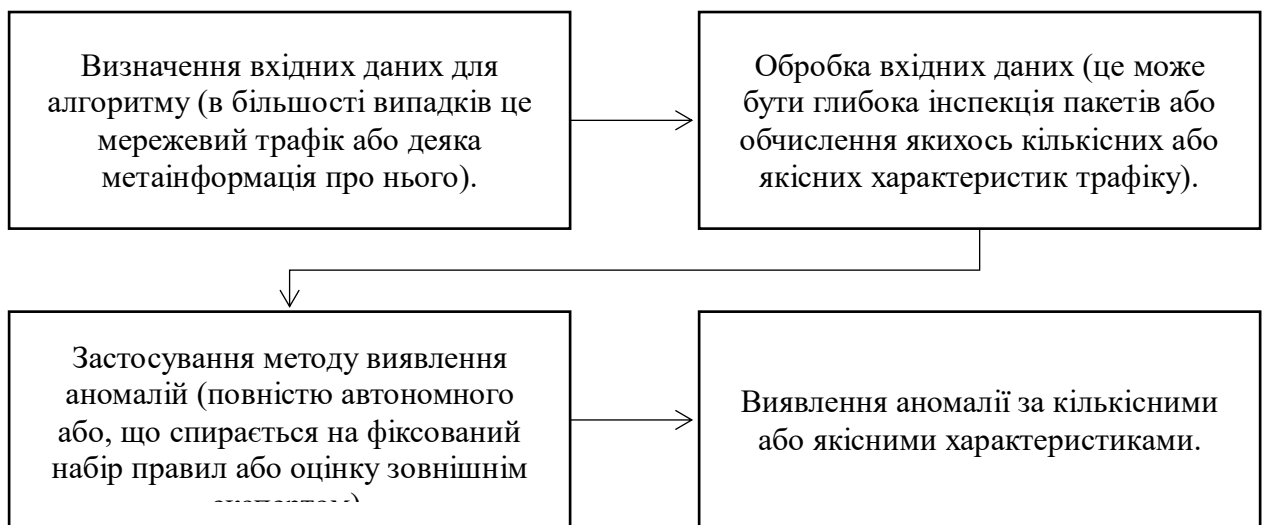


Рис. 2.5. Загальний алгоритм виявлення аномалій в системі

Головні відмінності в рішеннях, які виявляють аномалії у інформаційних системах, виражаються шляхом реалізації кроків 2 - «Обробка вхідних даних» та 3 - «Застосування методу виявлення аномалій».

### 2.3. Огляд інструментів виявлення аномалій

Аналіз інструментів виявлення аномалій у поведінці мережі є важливим компонентом систем безпеки мережі. Ці інструменти використовуються для

моніторингу активності мережі, виявлення незвичайних або підозрілих зразків поведінки і сповіщення про можливі загрози або вторгнення.

Ось деякі інструменти, які часто використовуються для виявлення аномалій у поведінці мережі (рис. 2.6).

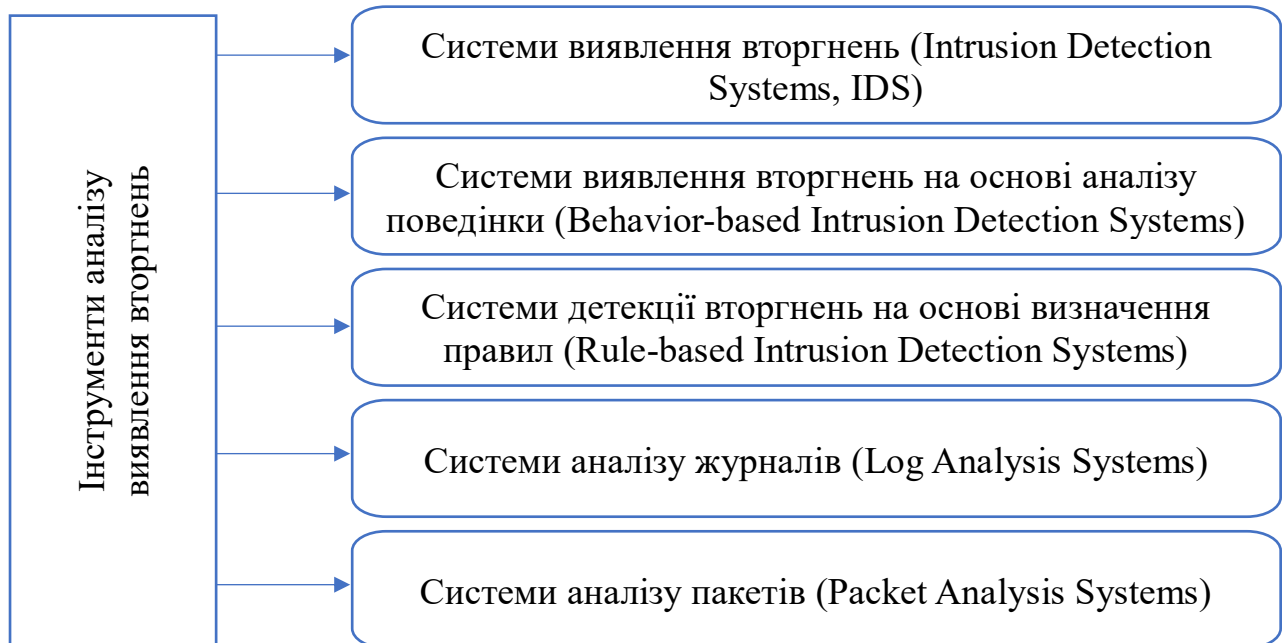


Рис. 2.6. Інструменти виявлення мережених вторгнень

Системи виявлення вторгнень (Intrusion Detection Systems, IDS) аналізують мережевий трафік і шукають незвичайні або підозрілі зразки активності. Вони можуть використовувати правила, сигнатури або аналітичні алгоритми для виявлення вторгнень або аномалій. IDS можуть бути розгорнуті як відокремлені фізичні пристрої або програмне забезпечення на сервері.

Системи виявлення вторгнень на основі аналізу поведінки (Behavior-based Intrusion Detection Systems) аналізують нормальний шаблон поведінки мережі і спостерігають за відхиленнями від цього шаблону. Вони використовують статистичні методи, машинне навчання або інші алгоритми для виявлення аномалій у поведінці мережі.

Системи детекції вторгнень на основі визначення правил (Rule-based Intrusion Detection Systems) використовують набір правил, що визначають

підозрілі або заборонені дії в мережі. Якщо активність мережі відповідає цим правилам, система сповіщає про потенційну загрозу.

Системи аналізу журналів (Log Analysis Systems) аналізують журнали подій і логи активності мережі для виявлення підозрілих або аномальних подій. Вони можуть використовувати алгоритми машинного навчання для виявлення незвичайних патернів або надзвичайних дій.

Системи аналізу пакетів (Packet Analysis Systems) аналізують вміст мережевих пакетів для виявлення підозрілих або заборонених дій. Вони можуть аналізувати заголовки пакетів, дані, шифровані комунікації та іншу інформацію для виявлення аномалій.

Ці інструменти можуть використовуватися окремо або в поєднанні для створення комплексної системи виявлення аномалій у поведінці мережі. Вони допомагають виявляти підозрілі дії, вторгнення або аномальну активність, що дозволяє прийняти відповідні заходи для забезпечення безпеки мережі.

Відповідно до опитування Neustar International Security Council, чотири з п'яти підприємств збільшать свої бюджети на кібербезпеку в 2022 році. Безпека мережі за допомогою методів виявлення аномалій поведінки мережі є життєво важливою для досягнення цілісного захисту.

Розглянемо кілька інструментів мережевої безпеки.

Awake Security — це інструмент мережевої безпеки на основі штучного інтелекту від публічної американської компанії Arista Networks. Його функціональність змагальної моделі спеціально створена для виявлення аномалій поведінки мережі.

Основні функції Awake Security включають: безперервний моніторинг мережі, аналіз зашифрованого трафіку, детальна обізнаність про поведінку мережі, сповіщення в реальному часі, вбудовані або інтегровані системи реагування, які пропонує відкритий інтерфейс прикладного програмування (API) для інтеграції із зовнішніми системами реагування та інші можливості.

Також Awake можна розгортати модульним способом, де датчики розгортаються в розподілених мережевих розташуваннях, а центр із підтримкою

машинного навчання знаходиться в центрі. Це забезпечує більшу масштабованість.

Даний програмний продукт може бути не ідеальним для компаній, яким потрібна інтегрована система виявлення аномалій і реагування на них. Крім того, ви повинні мати встановлений слід IoT, щоб максимізувати його потенціал.

Також на ринку даних програмних інструментів розглядається Flowmon NBAD. Flowmon є постачальником інструментів для моніторингу продуктивності та безпеки мережі, що базується в Чехії. Він пропонує інструменти встановлення аномалій мережі та аналізу поведінки для захисту від невідомих зловмисних програм.

Flowmon Anomaly Detection System (ADS) — це потужний інструмент, якому довіряють CISO та інженери безпеки в усьому світі. Він використовує складні алгоритми та машинне навчання для автоматичного визначення мережових аномалій і ризиків, які обходять традиційні рішення, такі як брандмауер, IDS/IPS або антивірус.

Загальний вигляд робочого середовища наведено на рис. 2.7.

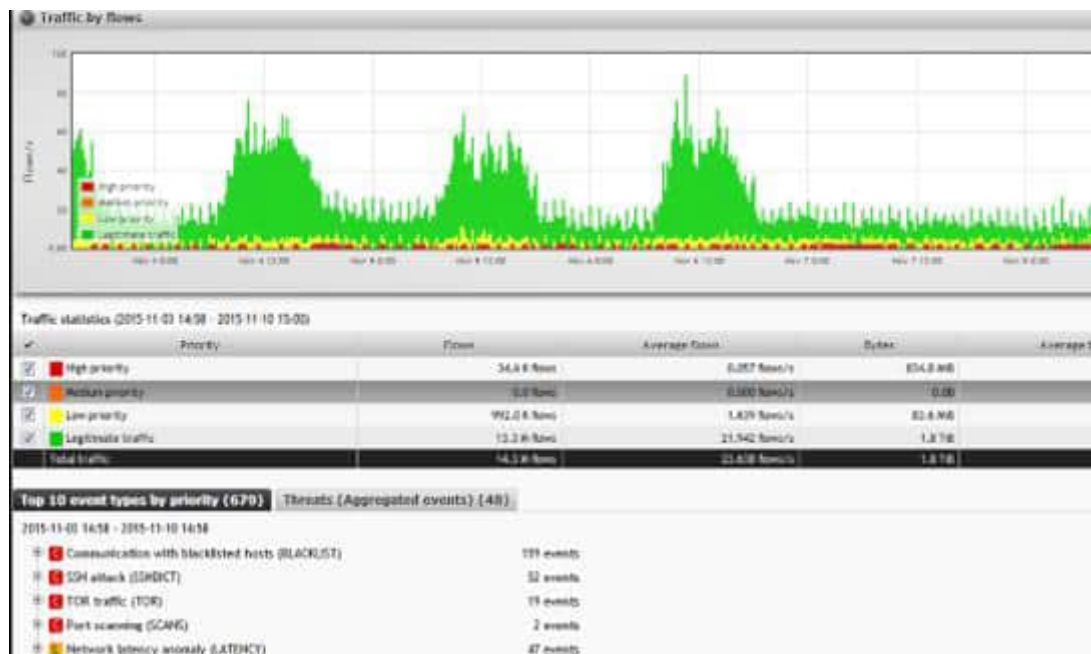


Рис. 2.7. Робоче вікно Flowmon NBAD

За допомогою даного інструменту здійснюється безперервний моніторинг мережі: постійно відстежує поведінку мережі та події, такі як додавання нових пристроїв, обсяги запитів, завантаження даних тощо.

До його функціонала також відноситься: аналіз зашифрованого трафіку, забезпечення детальною обізнаністю про ботнети, потенційне шкідливе програмне забезпечення, внутрішні загрози, витік даних тощо на основі мережевої активності.

Flowmon NBAD автоматично виявляє аномалії та надсилає сповіщення через інтегровану SIEM та використовує триадний підхід центру безпеки (SOC) для підключення SIEM, NDR і систем захисту кінцевих точок.

Також програмний продукт має функції, розроблені для виявлення внутрішніх загроз за допомогою аналізу поведінки мережі, наприклад машинне навчання (ML), евристики та бази даних репутації, на додаток до виявлення на основі сигнатур.

На рівні джерела є зонди та датчики. Flowmon Probe є найпотужнішим експортером даних про потоки на ринку, який генерує дані на рівні додатків і вимірює продуктивність.

Проблеми Flowmon можуть працювати на апаратному, віртуальному або хмарному рівні, пропускна здатність мережі від 10 Мбіт/с до 100 Гбіт/с, і вони забезпечують повну видимість L2, L3/4, L7.

Зонди безпосередньо збирають інформацію L2–L4 про IP-адреси, протоколи, час відповіді сервера, час проходження туди й назад, тремтіння тощо, одночасно використовуючи декапсуляцію трафіку для моніторингу реальної розмови між користувачем і програмою замість самого тунелю. Крім того, розширення IPFIX Flowmon надає додаткові дані L7, такі як імена хостів, URL-адреси, інформацію про браузер для протоколів HTTP/S та інші поля для протоколів, таких як DNS, DHCP, SQL, SMTP або Samba/CIFS тощо.

NetFlow Collector — це окремий пристрій для збору, тривалого зберігання та аналізу даних потоку з пристроїв із підтримкою потоку (балансувальників навантаження, комутаторів і маршрутизаторів), спеціальних зондів та інших джерел потоку. Він оснащений розширеною звітністю та візуалізацією даних.

Він має дуже потужний інтерфейс користувача, заснований на кількох інформаційних панелях:

Однією з цікавих функцій Flowmon є виявлення аномалій на основі поведінки (NBAD), яке постійно спостерігає за мережевим трафіком, аналізуючи зв'язок, щоб знайти аномалії та виявити підозрілу поведінку (рис. 2.8).

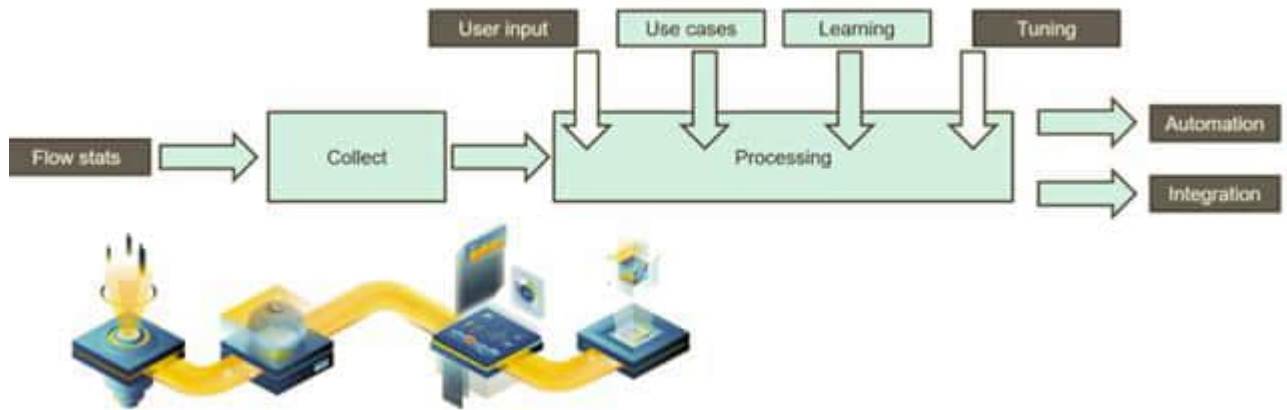


Рис. 2.8. Схема виявлення аномалій на основі поведінки (NBAD)

Flowmon надзвичайно багатий на функції та покладається на найсучасніший штучний інтелект (ШІ) та ML. Однак користувачі помітили проблеми з моделлю ліцензування, такі як політика кількох ліцензій і різні ціни в різних регіонах.

IBM QRadar Security Information and Event Management (SIEM) — це набір аналітичних рішень безпеки компанії. Завдяки IBM QRadar Network Insights підприємства можуть знаходити важливу інформацію про поведінку мережі та знати, які аномалії потребують негайного втручання. IBM QRadar Security Information and Event Management (SIEM) — це розширений інструмент, який допомагає виявляти та визначати пріоритетність загроз системам безпеки в реальному часі. Він може стати рушійною силою Центру безпеки вашої компанії (SOC), забезпечуючи безпеку вашої IT-інфраструктури. Це ідеальне рішення, якщо ваш бізнес хоче захистити дані клієнтів разом із власною інтелектуальною власністю.

Перетворіть свій SOC на інформаційний центр безпеки QRadar, щоб забезпечити свій захист провідною глобальною платформою безпеки (рис. 2.9).

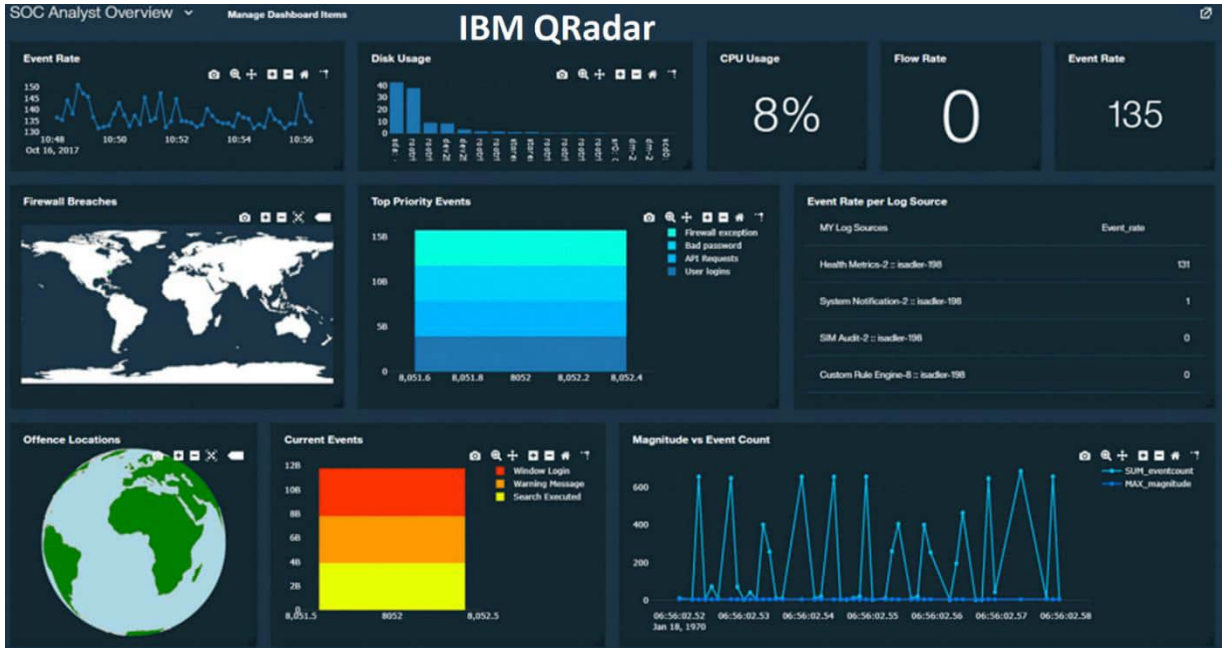


Рис. 2.9. Загальний вигляд інструменту IBM QRadar Network Insights

Основні функції цього інструменту включають:

1. безперервний моніторинг мережі: IBM може допомогти налаштувати інструмент для охоплення корпоративної мережі будь-якого типу та масштабу для постійного моніторингу без сліпих зон.
2. аналіз зашифрованого трафіку: його функція Inspector витягує методи шифрування та дані рівня з мережевого трафіку для аналізу поведінки без дешифрування.
3. детальна обізнаність про поведінку мережі: генерує статистичні дані та звіти про боковий рух трафіку між пристроями, викрадання даних і мережеві загрози.
4. сповіщення в режимі реального часу: користувачі можуть налаштувати сповіщення та робочі процеси за допомогою готових до використання посібників IBM.
5. Вбудовані або інтегровані системи реагування: інструмент не залежить від постачальника та може інтегруватися з будь-якою SIEM. Користувачі також можуть вибрати для реагування власний Security QRadar SIEM від IBM.

IBM доповнює виявлення аномалій поведінки мережі своїми провідними в галузі керованими послугами. Це зменшує ваші внутрішні зусилля та допомагає зберегти наявні інвестиції.

IBM QRadar Network Insights — комплексне рішення, яке може ефективно масштабуватися. Однак майте на увазі, що користувачі повідомляють про складний процес налаштування, і він може не підходити для малого бізнесу.

Для виявлення аномалій мережі також використовується інструмент NetFlow, який дозволяє вести спостереження за мережею від ManageEngine, відділу управління ІТ корпорації Zoho.

Він містить програмне забезпечення для виявлення аномалій мережі, яке вивчає поведінку мережі, щоб виявити підозрілу активність (рис.2.10).

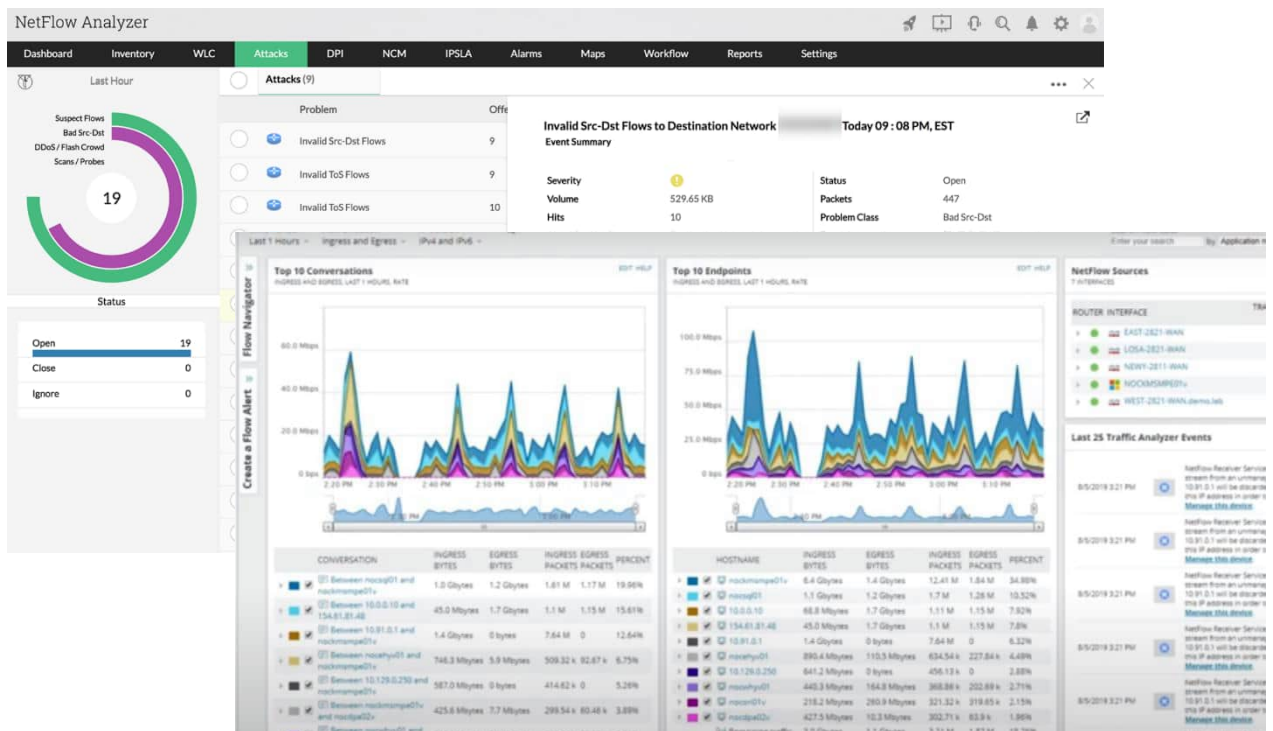


Рис. 2.10. Інструмент системи виявлення вторгнень NetFlow від корпорації

Zoho

Має широкий спектр інструментів:

- постійно аналізує поведінку мережі, щоб встановити стандартизовану базу продуктивності;
- використовує функцію Continuous Stream Mining Engine для дослідження моделей поведінки зашифрованого та незашифрованого трафіку;
- створює звіти Forensics, щоб забезпечити контекстну обізнаність та історичні тенденції;

- запускає автоматичні сповіщення в режимі реального часу, коли поведінка мережі перевищує базовий поріг, і прискорює аналіз першопричини;
- клієнти можуть вибрати інструменти реагування ManageEngine та SIEM або інтегруватися зі сторонніми системами через API.

Також NetFlow має кілька вбудованих алгоритмів, які дозволяють йому швидко зрозуміти природу прийнятної поведінки мережі в організації. Це допомагає зменшити кількість помилкових спрацьовувань і персоналізувати статистичні дані.

NetFlow – це детектор аномалій поведінки мережі відповідає потребам малих і середніх організацій. Однак йому може бути не вистачає складних можливостей AI/ML інструментів корпоративного рівня.

Інвестиції в безпеку мережі зростають, і інструменти виявлення аномалій поведінки мережі є важливими для зміцнення периметру вашого підприємства. Особливо в епоху віддаленої та гібридної роботи ці інструменти актуальні як ніколи.

У своєму опитуванні «Стан безпеки мережі 2022» Barracuda виявила, що 94% співробітників використовують корпоративні пристрої у своєму домашньому Інтернеті, ризикуючи мережевими атаками . 64% також спрямовують корпоративний трафік через публічних хмарних провайдерів. Десять інструментів, які ми обговорювали, сканують ці та інші поведінки мережі на наявність аномалій. Вони в режимі реального часу сповіщають про дійсні аномалії, щоб IT-команди могли їх усунути та постійно зосереджуватися на безпеці підприємства.

## **Висновки до розділу 2**

Виявлення аномалій у мережі є важливою складовою безпеки мережових інфраструктур. Це дозволяє вчасно виявляти та реагувати на незвичайну активність, яка може свідчити про потенційну загрозу або вторгнення.

Існують різні методи та технології виявлення аномалій, такі як сигнатурний аналіз, аналіз вимог, машинне навчання та системи виявлення та

запобігання вторгнень. Кожен з цих методів має свої переваги та обмеження і може бути використаний для розробки ефективних систем виявлення аномалій.

## РОЗДІЛ 3.

### ДОСЛІДЖЕННЯ АЛГОРИТМІВ ТА СИСТЕМИ РОЗПІЗНАВАННЯ АНОМАЛІЙ ТРАФІКУ

#### 3.1. Структура та архітектура системи виявлення вторгнень

Виявлення мережових вторгнень на основі аномалій відіграє життєво важливу роль у захисті мережових робіт від шкідливих дій. В останні роки методи інтелектуального аналізу даних набули важливого значення для вирішення проблем безпеки в мережі.

Система виявлення вторгнень – це, як правило, програмне забезпечення або апаратний пристрій, який відстежує вхідний і вихідний мережовий трафік на наявність ознак зловмисної діяльності або порушень політики безпеки. Системи виявлення вторгнень і продукти IDS часто порівнюють із сигналізацією зловмисника, яка сповіщає вас про будь-які дії, які можуть поставити під загрозу ваші дані або мережу.

IDS можна класифікувати за місцем виявлення: мережа або хост, що визначає тип даних, або за методом виявлення: сигнатура або аномалії.

З точки зору місця, де аналізується активність, існує два основних типи:

- система виявлення мережових вторгнень (NIDS). Вони встановлюються в запланованій точці мережі для перевірки трафіку з усіх пристроїв в мережі.
- системи виявлення вторгнень на основі хостів (HIDS). Вони працюють на незалежних хостах або пристроях мережі. HIDS відстежує вхідні та вихідні пакети з пристрою.

Система виявлення вторгнень (IDS) є важливим механізмом аналізу поведінки комп'ютерної мережі і виступає як значний елемент інфраструктури мережової безпеки.

Системи виявлення вторгнень (IDS) спрямовані на виявлення вторгнень з низьким рівнем помилкових тривог і високим рівнем виявлення. Хоча методи інтелектуального аналізу даних на основі classification популярні, вони не ефективні для виявлення невідомих атак. Методам навчання без нагляду було

надано більш пильний погляд на мережеві IDS, які незначні для виявлення динамічної діяльності вторгнення.

Для проведення класифікації IDS необхідно врахувати кілька факторів (рис. 3.1) [40].

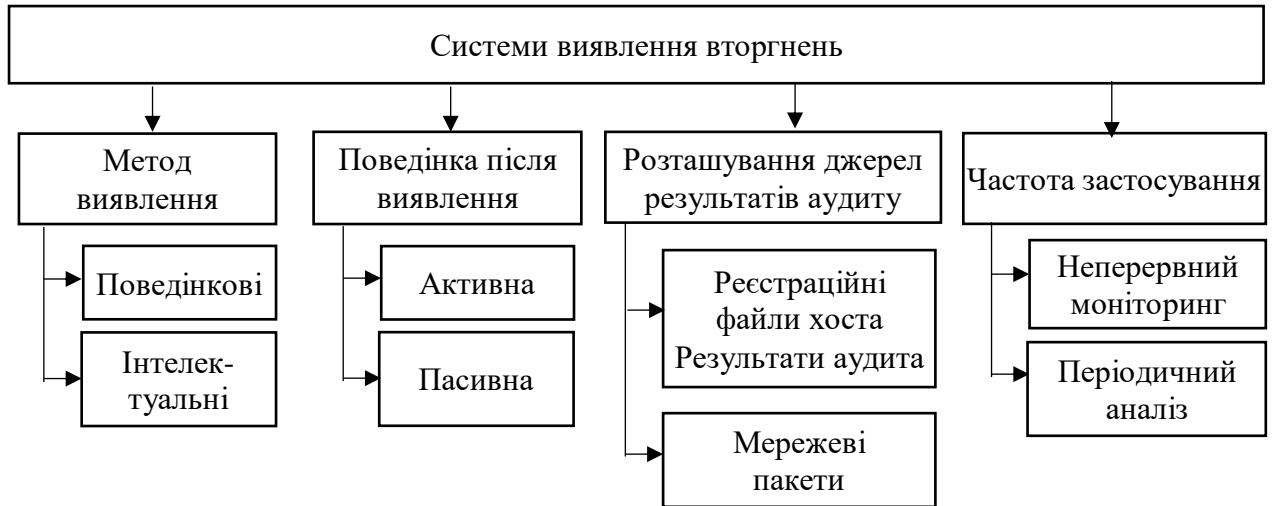


Рис. 3.1. Характеристики систем виявлення вторгнень [41]

Метод виявлення визначає характеристики аналізатора. Коли IDS використовує інформацію про нормальній поведінці контрольованої системи, вона називається поведінковою. Коли IDS працює з інформацією про атаки, вона називається інтелектуальною. Поведінка після виявлення вказує на реакцію IDS на атаки. Реакція може бути активною – IDS робить коригувальні (усуває лазівки) або дійсно активні (закриває доступ для можливих порушників, роблячи недоступними послуги) дії. Якщо IDS лише видає попередження, її називають пасивною.

Розташування джерел результату аудиту поділяє IDS залежно від виду вихідної інформації, яку вони аналізують. Вхідними даними можуть бути результати аудиту, системні реєстраційні файли чи мережні пакети (рис. 3.2).

Частота використання відображає або безперервний моніторинг контрольованої системи з боку IDS або відповідні періодичним запускам IDS для проведення аналізу.

Класифікувати IDS можна також за кількома параметрами [41]. За способами реагування розрізняють статичні та динамічні IDS. Статичні засоби

роблять «знімки» (snapshot) середовища та здійснюють їх аналіз, розшукуючи вразливе ПЗ, помилки в конфігураціях і т.д.



Рис.3.2. Класифікація систем виявлення вторгнень [41]

Статичні IDS перевіряють версії працюючих у системі додатків на наявність відомих уразливостей та слабких паролів, перевіряють вміст спеціальних файлів у директоріях користувачів чи перевіряють конфігурацію відкритих мережевих сервісів. Статичні IDS виявляють сліди вторгнення. Динамічні IDS здійснюють моніторинг у реальному часі всіх дій, що відбуваються в системі, переглядаючи файли аудиту або мережеві пакети, які передаються за певний проміжок часу. Динамічні IDS реалізують аналіз у реальному часі та дозволяють постійно стежити за безпекою системи.

За способом збору інформації розрізняють мережні та системні IDS. Мережеві (NIDS) контролюють пакети в мережевому оточенні і виявляють спроби зловмисника проникнути всередину системи, що захищається, або реалізувати атаку «відмова в обслуговуванні». Ці IDS працюють із мережними потоками даних.

У разі аналізу протоколів теж є свої переваги та недоліки. Через передпроцеси, що вимагають ретельної експертизи протоколів, аналіз протоколу може бути досить повільним. Крім того, правила перевірки системи протоколу важко написати і зрозуміти. Можна навіть сказати, що в цьому випадку

доводиться сподіватися на сумлінність виробника програми, оскільки правила відносно складні та важкі для самостійного налаштування.

На перший погляд, IDS на основі аналізу протоколу працюють повільніше, ніж системи на основі сигнатури, вони більш «грунтовні» в сенсі масштабності та результатів. Крім того, ці системи шукають «генетичні порушення» і часто можуть відловлювати найсвіжіші “експлоїти нульового дня”.

Системи виявлення вторгнень є важливим компонентом захисту мережі, доповнюючи інші механізми безпеки та допомагаючи реагувати на потенційні загрози та атаки.

Системами виявлення вторгнень (IDS) називають безліч різних програмних та апаратних засобів, що об'єднуються однією спільною властивістю - вони займаються аналізом використання довірених їм ресурсів і, у разі виявлення будь-яких підозрілих або просто нетипових подій, здатні робити деякі самостійні дії щодо виявлення, ідентифікації та усунення причин [42].

Але системи виявлення вторгнень лише один із інструментів захисного арсеналу і він не повинен розглядатися як заміна будь-якого з інших захисних механізмів. Захист інформації найбільш ефективний, коли в інтрамережі підтримується багаторівневий захист. Вона складається з наступних компонентів [42].

Інтенсивне зростання кількості інтернет-атак дійсно підвищує важливість безпеки мережевої діяльності в комп'ютерних мережах. У зв'язку з цим системи безпеки, включаючи системи виявлення вторгнень (IDS), стають необхідними для боротьби з атаками і захисту мережі від потенційних загроз.

Основна мета IDS полягає у виявленні підозрілої або незвичайної активності в мережі, що може вказувати на потенційну атаку або порушення безпеки.

IDS контролює мережевий трафік, аналізує його і шукає ознаки, які вказують на вторгнення або аномальну активність. Ці ознаки можуть включати підозрілі шаблони поведінки, незвичайні або несанкціоновані з'єднання, спроби зламу паролів, недостовірні чи підроблені пакети даних тощо. IDS використовує правила, сигнатури або аналітичні алгоритми для виявлення цих ознак.

Коли IDS виявляє підозрілу активність, він сповіщає системного адміністратора або саму систему про потенційну атаку або загрозу. Це може бути зроблено шляхом надсилання сповіщення адміністратору, активування автоматичних заходів захисту, які забороняють підозрілі дії або інших методів, що залежать від конфігурації IDS.

Важливо зауважити, що IDS не завжди може запобігти атакам безпосередньо, він слугує переважно для виявлення аномалій і сповіщення про них. Застосування заходів захисту та реагування на виявлені загрози залежить від системного адміністратора або відповідальної особи з безпеки мережі.

У систем виявлення вторгнень доцільно розрізняти локальну та глобальну архітектуру. У межах локальної архітектури реалізуються елементарні складові, які можуть бути об'єднані обслуговування корпоративних систем [42].

Основні елементи локальної архітектури та зв'язку з-поміж них показані рис. 3.3.

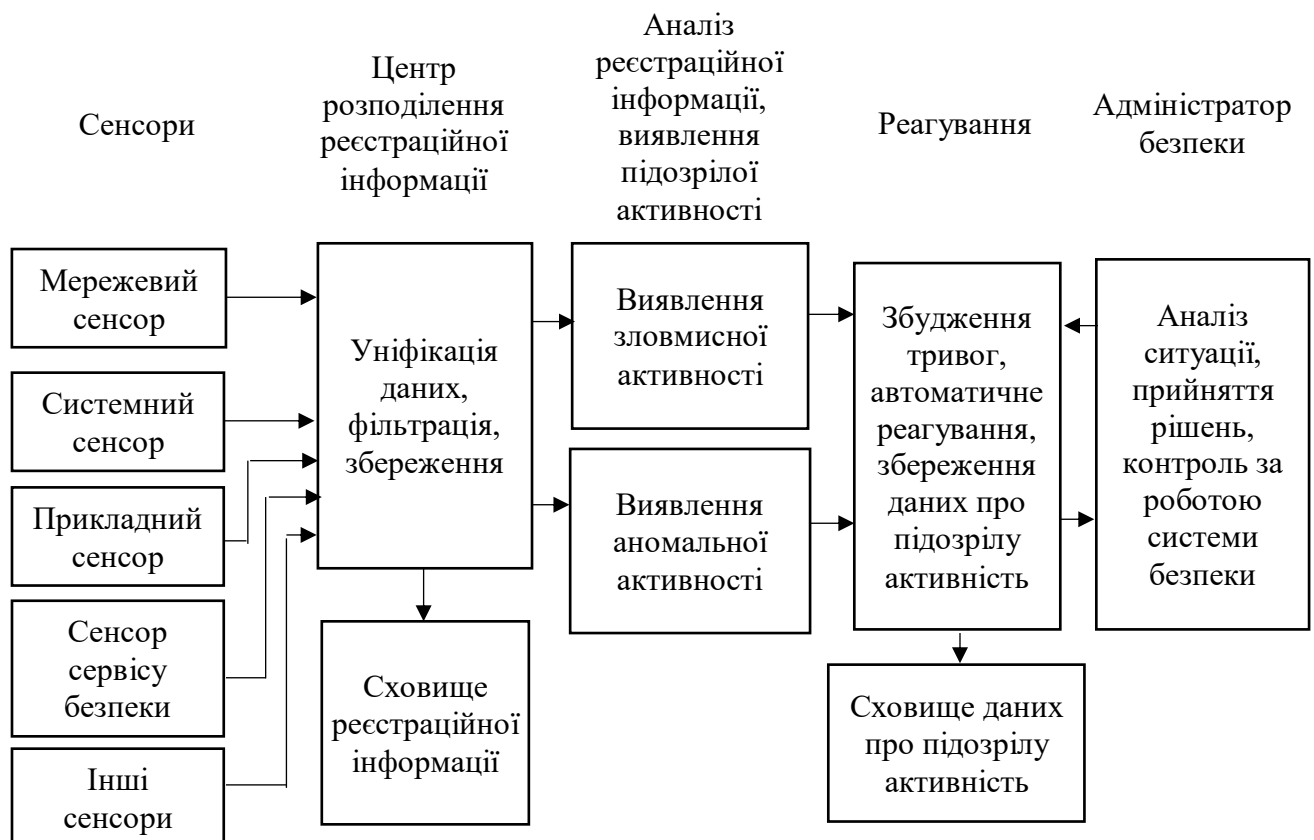


Рис. 3.3. Основні елементи локальної архітектури систем виявлення вторгнень

Первинний збір даних здійснюють агенти, звані також сенсорами. Реєстраційна інформація може вилучатись із системних або прикладних журналів (технічно нескладно отримувати її і безпосередньо від ядра ОС), або видобуватись з мережі за допомогою відповідних механізмів активного мережного обладнання або шляхом перехоплення пакетів за допомогою встановленої в режим моніторингу мережної картки.

На рівні агентів (сенсорів) може виконуватись фільтрація даних з метою зменшення їхнього обсягу. Це вимагає від агентів деякого інтелекту, зате розвантажує інші компоненти системи.

Агенти передають інформацію до центру розподілу, який приводить її до єдиного формату, можливо, здійснює подальшу фільтрацію, зберігає у базі даних та спрямовує для аналізу статистичного та експертного компонентів. Один центр розподілу може обслуговувати кілька детекторів.

Змістовий активний аудит починається зі статистичного та експертного компонентів. Якщо в процесі статистичного або експертного аналізу виявляється підозріла активність, відповідне повідомлення надсилається вирішувачу, який визначає, чи є тривога виправданою, і вибирає спосіб реагування.

Ефективна система виявлення вторгнень повинна вміти чітко пояснити, чому вона підняла тривогу, наскільки серйозна ситуація та які рекомендовані способи дії. Якщо вибір повинен залишатися за людиною, то нехай він зводиться до кількох елементів меню, а не вирішення концептуальних проблем.

Глобальна архітектура має на увазі організацію однорангових та різнорангових зв'язків між локальними системами виявлення вторгнень (рис. 3.4).

На одному рівні ієрархії розташовуються компоненти, що аналізують підозрілу активність із різних точок зору. Наприклад, на хості можуть розташовуватись підсистеми аналізу поведінки користувачів та додатків. Їх може доповнювати підсистема аналізу активності мережі. Коли один компонент виявляє щось підозріле, то у багатьох випадках доцільно повідомити про це сусідам або для вжиття заходів, або для посилення уваги до певних аспектів поведінки системи.

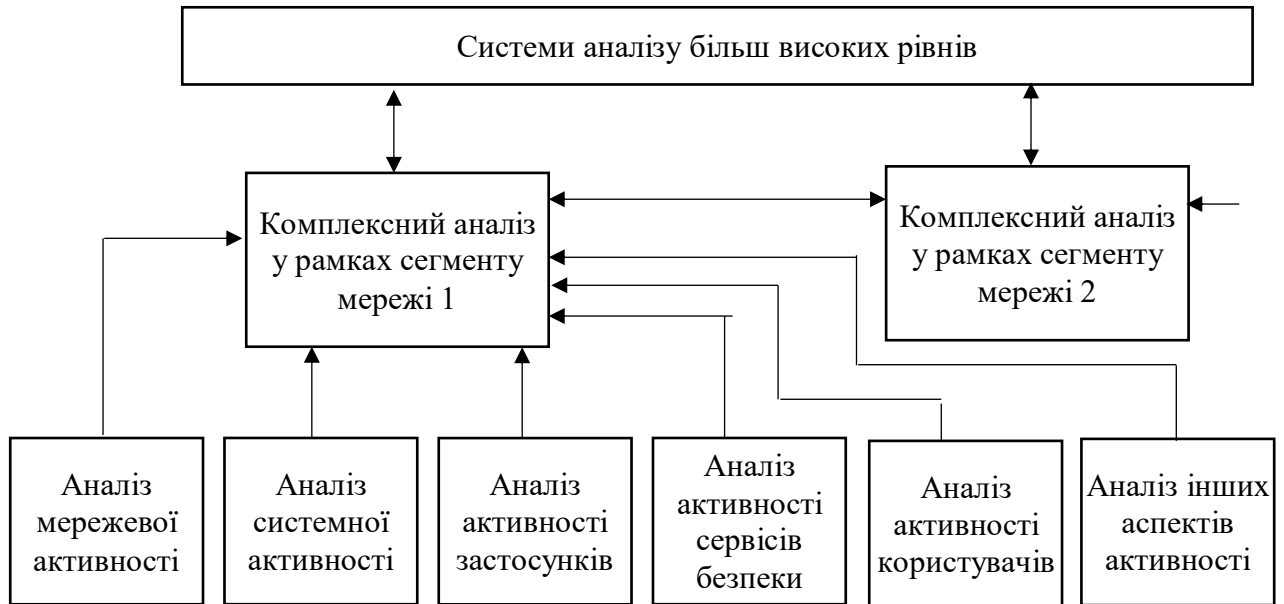


Рис. 3.4. Глобальна архітектура систем виявлення вторгнень

Різноманітні зв'язки використовуються для узагальнення результатів аналізу та отримання цілісної картини того, що відбувається. Іноді локальний компонент недостатньо підстав для порушення тривоги, але "за сукупністю" підозрілі ситуації можуть бути об'єднані і спільно проаналізовані, після чого поріг підозрілості виявиться перевищеним. Цілісна картина, можливо, дозволить виявити скоординовані атаки на різні ділянки інформаційної системи та оцінити збитки в масштабі організації.

### 3.2. Огляд підходів розпізнавання аномалій трафіку

В науковій літературі зустрічаються різні підходи до розуміння типологізації систем виявлення аномалій. Здебільшого, якщо розглядати більш широко, то виявлення аномалій це один з процесів який складає усю сукупність з виявлення вторгнень (IDS). [43]

Інша складова процесу виявлення вторгнень заснована на певних підписах (SIDS).

В свою чергу, підходів до розуміння видів саме систем виявлення аномалій - багата кількість.

Один з варіантів класифікації методів виявлення аномалій, представлено на рис. 3.5.):



Рис. 3.5. Види методів виявлення аномалій.

Широке поширення має підхід, заснований на статистиці, який передбачає збір та вивчення трафіку через набір елементів та з використанням статистичної моделі нормальної його поведінки.

Статистичні методи використовуються у вигляді певних тестів. Більш доречно застосовувати до точкових аномалій.

Для окремих аномалій знаходяться екстремальні значення за допомогою стандартизованої оцінки ( $Z$ -оцінка,  $Z$ -value, standart score) або коефіцієнта ексцесу (Kurtosis measure).

Проаналізуємо використання  $Z$ -оцінки, яка дозволяє визначити кількість стандартних відхилень від середнього значення для даного числа вибірки [44-46]. Прикладом може бути виявлення апаратної аномалії, що унеможливорює стандартну роботу системи передачі (отримання) трафіку - виявлення затримок (лагів). Пакети даних пересилаються від моменту запиту користувача до його фактичного отримання за певний проміжок часу.

Представимо, що усього користувач повинен отримати 5 пакетів даних (вибірка). Кожний пакет надходить до користувача за таких часових рамок: Перший пакет - за 7.5 мкс, другий - за 6.5 мкс., третій - за 7 мкс., четвертий за 7 мкс., п'ятий - за 8 мкс.

На першому кроці визначаємо середнє значення обраної вибірки:

$$(7,5+6,5+7+7+8)/5=7,2$$

На наступному кроці визначаємо дисперсію, тобто значення, яке характеризує міру зміни чисел відносно середнього значення. Для цього слід відійняти з кожного числа вибірки отримане вище середнє значення. В подальшому, отримані результати слід звести у квадрат. Після цього, слід скласти всі отриманні значення:

$7,5 - 7,2 =$	$0,3$	$\times$	$0,3$	$=$	$0,09$
$6,5 - 7,2 =$	$(- 0,7)$	$\times$	$(- 0,7)$	$=$	$0,49$
$7 - 7,2 =$	$(- 0,2)$	$\times$	$(- 0,2)$	$=$	$0,04$
$7 - 7,2 =$	$(- 0,2)$	$\times$	$(- 0,2)$	$=$	$0,04$
$8 - 7,2 =$	$0,8$	$\times$	$0,8$	$=$	$0,64$
					$\Sigma$ 1,3

Отже, загальної дисперсія знаходиться за формлою;

$$\frac{1,3}{5(\text{кількість значень у вибірці}) - 1} = 0,325$$

Таким чином дисперсія отриманих користувачем пакетів трафіку дорівнює 0,325. Знаходимо стандартне відхилення вибірки слід виийняти корінь з отриманого результату дисперсії:

$$\sqrt{0,325} = 0,57$$

Тепер визначаємо Z-оцінку:

$$Z = \frac{X_i - \bar{X}}{O},$$

де  $X_i$  - будь-яке число з обраної вибірки,  $\bar{X}$  – середнє значення,  $O$  - стандартне відхилення.

Отже, обчислюємо значення Z-оцінки для четвертого пакету отриманої інформації:

$$Z = \frac{7 - 7,2}{0,57} = -0,351$$

Таким чином, четвертий пакет даних віддалений на (- 0,351) від середнього значення у виборці. Негативний показник вказує на те, що опрацьоване число нижче середнього значення (з знаком «-»). У випадку позитивного числа - навпаки, більше ніж середнє число.

Методи згладжування рядів: змінна середня, експоненціальне згладжування і регресія. Найпростіший спосіб згладити тимчасовий ряд - використовувати замість вихідних значень половину суми сусідніх.

Якщо використовувати не одне, а кілька що передує значень, тобто середнє арифметичне сусідніх значень, то таке згладжування називається простим ковзним середнім з шириною вікна.

Як за приклад приведеному нижче, за основу графіку узяті певний показник цін продукту на заданому проміжку часу (рис. 3.6.):



Рис. 3.6. Показник ковзної середньої

Як бачимо, тренд з самого початку йшов до низу, надалі, починаючи з сьомого числа тренд пішов догори. Ковзна середня так само пішла до гори. Залежність показника від самого тренду наступна: у разі зростаючого тренду - він перетинає показник знизу до гори, у разі низпадаючого тренду навпаки - згори до низу, як почало простежуватись від третього та одинадцятого чисел.

Наступний метод – метод автокореляції. Вихідний ряд / функція представляється у вигляді нескінченної суми елементів і береться кілька перших значущих коефіцієнтів.

Для пошуку автокореляції просто зрушуємо функцію в бік і шукаємо такий стан, щоб відстань / площа між вихідної і зрушеною функцією (виділено

червоним) було мінімально. Очевидно для алгоритму слід задати крок зсуву і максимальна межа, при досягненні якого вважаємо, що пошук періоду не вдался.

Ще одна група методів, заснована на знаннях і система намагається ідентифікувати запитувані дії на основі існуючих системних даних, таких як специфікації протоколів та екземпляр мережевий трафіку. Для використання методів, побудованих на основі знань, необхідно застосування певних модельних тестів. Вони описують дані точки, що відхилилися від певної, наперед встановленої, моделі та, як наслідок, є аномальними. Розрахунок відбувається за допомогою неповного сингулярного розкладання. В результаті система шукає максимально схожа матриця.

Широкий спектр методів відноситься до категорії машинного навчання до яких відносяться такі як метод опорних векторів для одного класу (OneClassSVM), ізолюючий ліс (IsolationForest), еліпсоїдальної апроксимації даних (EllipticEnvelope), тощо

OneClassSVM - це одна з форм класичного алгоритму, однак, як впливає з назви, для його навчання нам досить мати всього один клас.

Якщо ми маємо справу з завданням, де для тренування нам доступні тільки «хороші» (істинні) спостереження без аномалій, то ми можемо скористатися цією моделлю і навчитися для кожного нового спостереження говорити, чи є воно аномальним чи ні.

Загальна ідея: перетворити простору ознак і провести розділяє гіперплощину так, щоб спостереження знаходились якнайдалі від початку координат.

Позитивне: Модель здатна проводити нелінійні розділяють кордони. Зручно використовувати, коли в даних недостатньо «поганих» спостережень, щоб використовувати стандартний підхід навчання з учителем - бінарну класифікацію.

Негативне: Модель може дуже сильно «перенавчитися» і видавати велику кількість хибно негативних результатів, якщо розділяє зазор між отриманими показниками без похибки, тобто будь-який зазор - аномалія. До того ж, потрібно бути абсолютно впевненим, що тренувальні дані не містять аномалій, інакше

алгоритм буде вважати їх нормальними спостереженнями.

Isolation Forest. Ідея даного алгоритму заснована на такому принципі: проводиться випадкове розбиття простору ознак, таке, що ізольовані точки відсікаються від нормальних кластеризованих даних. Остаточний результат усереднюється по декільком запусках стохастичного алгоритму.

Алгоритм розпізнає аномалії різних видів: як ізольовані точки з низькою локальною щільністю, так і кластери аномалій малих розмірів. Іншими словами, поділ даних здійснюється з вихідної точки, яка буде ділитись декілька разів.

Наступний рівень буде також ділитись певну кількість разів.

Кожний розподіл передбачає собою позитивні та негативні розподілення. Позитивні розподілення в подальшому буде ділитись. Негативні розподілення не підлягають діленню.

Графік буде виглядати у вигляді певних дерев, з кінцевими точками - екстремумами (рис. 3.7.):

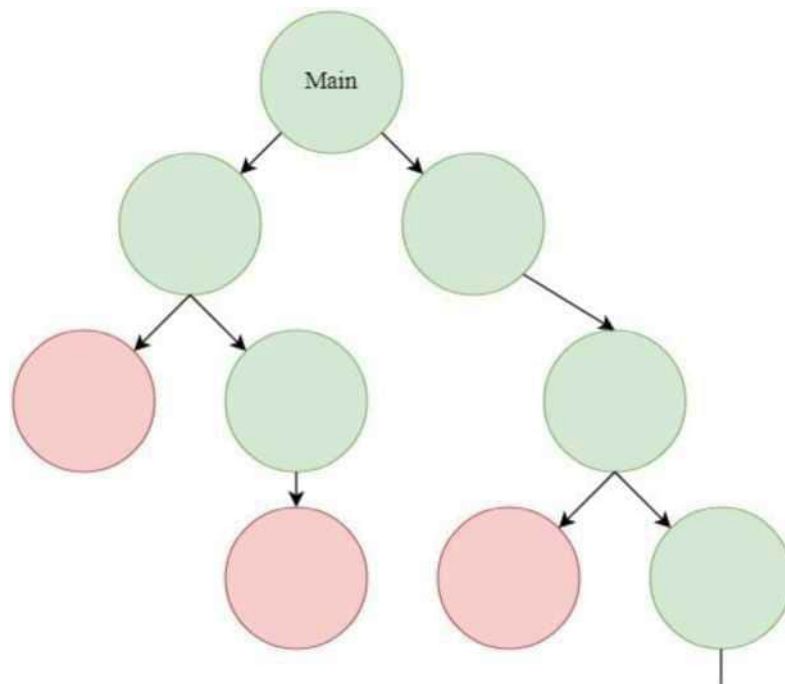


Рис. 3.7. Модель Isolation Forest

Як бачимо, ті позначки, які не пройшли контроль (червоні) - з них не виходить розподіл, зелені у свою чергу - розподіляються.

EllipticEnvelope. Припустимо, що наші дані нормально розподілені. Це

припущення, яке не може бути вірним для всіх наборів даних, але коли це так, це доводить ефективний метод для визначення аномалій.

Scikit-Learn-хcovariance.ElasticEnvelope то функція, яка намагається з'ясувати ключові параметри загального розподілу обраних даних, припускаючи, що всі наші дані є виразом базового багатовимірного розподілу Гаусса.

Отже, існує широка гама різних алгоритмів, методів та підходів, які базуються на використанні нейронних мереж та штучного інтелекту. Проведемо порівняльний аналіз цих алгоритмів (табл. 3.1).

Таблиця 3.1

Порівняльна таблиця алгоритмів

Тип	Швидкість оброблення даних та отримання результатів	Навчання	Об'єм аналізу даних	Актуальність
На основі статистики	Швидка (потрібні високі програмні можливості)	Самостійно навчається виходячи з наявних даних	Великий (залежить від програмного навантаження)	Середня
На основі знань	Середні (залежать від наявності істинної інформації)	Потрібно постійно додавати нові дані	Великий (залежить від програмного навантаження)	Середня
На основі машинного навчання	Швидка (залежить від можливості вірного з'ясування даних)	Самостійно навчається виходячи з наявних даних	Середній (залежить від використаних методів аналізу)	Велика
Інші методи	Середні (залежить від використаних методів)	Комбіновані методи	Комбіновані методи	Середня

Отже, аналіз порівняння вищезгаданих методів та алгоритмів здійснювався за такими характеристиками, як швидкість, навчання, об'єм, актуальність. Це можна використовувати як певні вихідні дані за яким здійснюється покращення системи аналізу аномалії у мережі.

### 3.3. Використання методів машинного навчання для IDS

Машинне навчання може побудувати необхідну модель автоматично на основі деяких навчальних даних. Застосування такого підходу потребує наявності необхідної підготовки даних, але це завдання є менш складним порівняно з обчисленням аномальної моделі [47]. Зі збільшенням складності та

кількості різних атак, методи машинного навчання, які дозволяють створювати та підтримувати системи виявлення аномалій (ADS) з меншим втручанням людини, є єдиним практичним підходом для створення наступного покоління систем виявлення вторгнень.

Застосування методів машинного навчання для виявлення вторгнень дозволить автоматично побудувати модель, засновану на наборі навчальних даних, що містить екземпляри даних, описаних за допомогою набору атрибутів (ознак). Атрибути можуть бути різних типів, наприклад, якісними або кількісними. Були розглянуті різні алгоритми виявлення аномалій, у таблиці 3.2 представлені переваги та недоліки кожного з них [48, 49, 50].

Таблиця 3.2

### Переваги та недоліки алгоритмів виявлення аномалій

Методи	Переваги	Недоліки
K-найближчих сусідів	Легко реалізуємо, коли є кілька предикторів. Застосовується для побудови моделей, що обробляють нестандартні типи даних, такі як текст.	Великі вимоги щодо обсягу пам'яті. Залежить від вибору функції подібності, яка використовується для порівняння екземплярів. Відсутність принципового способу вибору, крім через перехресну перевірку або аналогічний спосіб. Дорога обчислювальна техніка.
Нейронна мережа	Нейронна мережа може виконувати завдання, які не виконає лінійна програма. Коли один елемент не справляється із завданням, метод може продовжити роботу завдяки паралельній обробці даних. Нейронну мережу не потрібно перепрограмувати. Може бути реалізована у будь-якому додатку.	Нейронна мережа потребує навчання. Високий час обробки великих нейронних мереж.
Машина опорних векторів	Знаходження оптимального поділу гіперплощини. Обробляє більшу розмірність даних. Зазвичай працює дуже добре.	Потребує як позитивних, так і негативних прикладах. Потрібно вибрати хорошу функцію ядра. Вимагає багато пам'яті та процесорного часу. Є деякі чисельні проблеми стійкості під час вирішення обмеження QP
Дерево рішень	Простий у реалізації. Потребує невеликої підготовки даних. Можливість обробляти як числові та інші типи даних. Використовує модель білої скриньки. Можливість перевірки моделі за	Проблема навчання оптимального дерева рішень, як відомо, є NP-повним за декількома аспектами оптимальності і навіть для простих завдань. При створенні дерева рішень можуть вийти неоптимальні та дуже складні дерева, які погано обробляють дані.

Методи	Переваги	Недоліки
	допомогою статистичних тестів. Працює з великими даними за короткий проміжок часу.	Існують завдання, які неможливо відобразити деревом рішень, тому що воно не описує її повністю.
Самоорганізовані карти	Простий у реалізації. Працює з нелінійним набором даних. Візуалізація багатовимірних даних на 1 або 2-мірному просторі робить його унікальним, особливо зменшення розмірності.	Потрібно багато часу для обчислень.
К-середніх	Низька складність	Необхідність вказівки К. Чутливі до перешкод та сторонніх точок даних. Кластери чутливі до первісного значення.
Алгоритм нечіткої кластеризації Fuzzy C-means	Дозволяє точці даних бути у кількох кластерах.	Необхідно визначити кількість кластерів С. Необхідно визначити граничне значення учасників. Кластери чутливі до початкового завдання центроїдів.
Апроксимація	Можна легко змінити модель, щоб адаптувати до різних розподілів наборів даних. Кількість параметрів не збільшується зі збільшенням навчальних даних.	У деяких випадках спостерігається повільна збіжність.

Виявлення аномалій включає контрольовані і неконтрольовані методи. Порівняльний аналіз показав, що контрольовані методи навчання значно перевершують неконтрольовані, якщо тестові дані не містять невідомих атак. Серед контрольованих методів найкраща продуктивність досягається за рахунок нелінійних методів, таких як SVM, багат шаровий перцептрон та методів, що базуються на правилах. Неконтрольовані методи, такі як К- середніх, SOM, і один клас SVM показують більш високу продуктивність порівняно з іншими методами, хоча вони відрізняються ефективності виявлення всіх класів атак [51], [52].

Аналіз показав, що контрольовані методи навчання значно перевершують неконтрольовані, якщо дані не містить невідомих атак. Серед контрольованих методів, найкраща продуктивність досягається за рахунок нелінійних методів, таких як SVM, багат шаровий перцептрон та методи, що базуються на правилах. Неконтрольовані методи, такі як К-Середня, SOM, і один клас SVM показують

більш високу продуктивність в порівнянні з іншими методами, хоча вони показують різну ефективність виявлення всіх класів атак.

### **3.4. Вибір основного алгоритму для системи розпізнавання аномалій трафіку**

Виходячи з проведених вище дослідженнях щодо підходів до виявлення аномалій у мережі, було за основу взято підходи машинного навчання і розглядався процес покращення моделі.

На основі характеристик, які притаманні системам розпізнавання аномалій заснованими на машинному навчанні - обрано модель, що базується на аналізі основних компонентів (PCA).

PCA (Principal Component Analysis) є визнаним методом машинного навчання, який широко використовується в дослідницькому аналізі даних. Він дозволяє розкрити внутрішню структуру даних та пояснити відмінності між ними.

Головна мета PCA - зменшити розмірність даних, зберігаючи при цьому якомога більше варіації у них. Він перетворює початкові змінні (або ознаки) у новий набір незалежних змінних, які називаються головними компонентами. Головні компоненти відсортовані за спаданням варіації, що дозволяє виокремити головні характеристики даних.

Одна з основних переваг PCA полягає в тому, що він може використовуватись для зменшення розмірності даних без втрати істотної інформації. Це дозволяє спростити аналіз даних та покращити ефективність алгоритмів машинного навчання, особливо при роботі з великими наборами даних.

Крім зменшення розмірності, PCA також може використовуватись для візуалізації даних, виявлення аномалій, фільтрації шуму та вирішення проблеми лінійної залежності між ознаками.

Загалом, PCA є потужним інструментом для дослідницького аналізу даних, оскільки він допомагає виявити внутрішні закономірності та структуру даних, що дозволяє зробити більш обґрунтовані висновки та прийняти рішення.

РСА допомагає створити модель у сценаріях, де можна легко отримати навчальні дані з одного класу, наприклад, допустимі транзакції, але важко отримати достатню кількість вибірок цільових аномалій.

РСА виконується шляхом аналізу даних з кількома змінними. Він виконує пошук кореляції між змінними та визначає поєднання значень, які найкраще фіксують відмінності результатів. Ці комбіновані значення функцій використовуються створення більш компактного простору функцій, званого головними компонентами.

Для виявлення аномалії кожен новий вхід аналізується. Алгоритм виявлення аномалії обчислює її проекцію на власні вектори разом із нормалізованою помилкою відновлення. Нормалізована помилка використовується як оцінка аномалії. Чим більше значення помилки, тим більше аномалій в екземплярі.

Наприклад, щоб виявити шахрайські транзакції, часто не вистачає прикладів шахрайства для навчання. Але у вас може бути багато прикладів хороших угод. Компонент виявлення аномалій RSA вирішує проблему, аналізуючи доступні функції, щоб визначити, що становить «нормальний» клас.

Потім компонент застосовує метрики відстані визначення варіантів, що представляють аномалії. Цей підхід дозволяє навчати модель, використовуючи існуючі незбалансовані дані.

Таким чином, RSA служить «забором», що виокремлює певні характеристики серед масиву інформації, що позитивно впливає на виявлення аномалій, шляхом звуження області пошуку останніх [53].

Використання систем Generative Adversarial Networks (GAN) для виявлення аномалій є одним зі способів застосування цієї технології. GAN є типом глибоких нейронних мереж, які складаються з двох головних компонентів: генератора і дискримінатора.

Генератор має за мету створювати нові зразки, що схожі на дані тренувального набору, тоді як дискримінатор намагається відрізнити справжні зразки від зразків, згенерованих генератором. Цей процес триває досягнення

рівноваги між генератором і дискримінатором, коли генеровані зразки стають важко відрізнити від справжніх зразків.

Застосування GAN для виявлення аномалій відбувається на основі спостереження, що аномалії мають відмінні властивості в порівнянні з нормальними даними. В процесі навчання GAN модель навчається розпізнавати нормальні зразки, тому відхилення від цих норм може бути ознакою аномалій.

Один з підходів до використання GAN для виявлення аномалій полягає в тому, щоб навчити генератор створювати нові зразки, які максимально наближені до нормальних даних. Потім дискримінатор використовується для класифікації зразків як "нормальні" або "аномальні". Якщо дискримінатор не може відрізнити зразки, згенеровані генератором, від нормальних зразків, це може вказувати на наявність аномалій.

Однак варто враховувати, що використання GAN для виявлення аномалій може бути складним завданням і вимагати достатньої кількості тренувальних даних, а також налагодження гіперпараметрів для досягнення оптимальних результатів. Крім того, важливо мати на увазі, що GAN може бути схильним до генерування фальшивих позитивів або пропускати деякі аномалії, тому доцільно розглядати цей підхід як одну з компонент системи виявлення аномалій, а не єдиний метод.

Звичайно, система, яка покликана виявляти аномалії мережі не може бути заснована виключно на статистично-економічному показнику (підході).

Слід також враховувати той факт, що систему слід навчати виокремлювати.

Тобто, якщо PCA акумулює інформацію, та приводить її до певного виду, то фактичне віднаходження аномалій слід покласти на іншу модель структури.

В запропонованій моделі розглядається підхід заснований на використанні системи GAN [54].

Генеративно-змагальна нейромережа (Generative adversarial network, GAN) - архітектура, що складається з генератора (G) та дискримінатора (D), налаштованих на роботу один проти одного.

Потенціал GAN величезний, оскільки він імітує будь-який розподіл даних.

GAN навчають створювати структури.

Коротко розглянемо роботу алгоритму:

Дискримінатор (D). Дискримінаційні алгоритми намагаються класифікувати вхідні дані. Враховуючи особливості отриманих даних, вони намагаються визначити категорію, до якої ці дані належать.

Наприклад, аналізуючи всі слова у листі - дискримінаційний алгоритм може передбачити, чи є повідомлення спамом або ні.

Спам - це категорія. Пакет слів, зібраний з електронної пошти - образи, які становлять вхідні дані.

Математично категорії позначають  $y$ , а образи позначають  $x$ . Запис  $p(y|x)$  використовується для позначення «ймовірності  $y$  при заданому  $x$ », яка позначає «ймовірність того, що електронний лист є спамом при наявному наборі слів» [55].

Отже, дискримінаційні функції зіставляють образи із категорією. Вони зайняті лише цією кореляцією.

Генератор (G). Генеративні алгоритми зайняті зворотним. Замість передбачати категорію за наявними образами, вони намагаються підібрати образи до цієї категорії.

У той час як дискримінаційні алгоритми хвилює взаємозв'язок між  $y$  та  $x$ , генеративні алгоритми хвилює «звідки беруться  $x$ ».

Вони дозволяють знаходити  $p(x|y)$ , ймовірність  $x$  при даному  $y$  або ймовірність образів при даному класі (генеративні алгоритми також можуть використовуватися як класифікатори. Вони можуть робити більше, ніж класифікувати вхідні дані.)

Ще одне уявлення про роботу генеративних алгоритмів можна отримати, розділяючи дискримінаційні моделі від генеративних таким чином:

1. Дискримінаційні моделі вивчають кордон між класами;
2. Генеративні моделі моделюють розподіл окремих класів [55].

Принцип роботи GAN:

1. Генератор створює нові екземпляри даних, а дискримінатор оцінює їх на справжність; тобто. дискримінатор вирішує, чи кожен екземпляр даних, який

він розглядає, відноситься до набору тренувальних даних чи ні.

2. Паралельно з цим, генератор створює нові дані, які він передає дискримінатору. Він робить це, сподіваючись, що вони будуть прийняті справжніми, хоча є підробленими.
3. Ціль генератора полягає в тому, щоб генерувати дані, які будуть пропущені дискримінатором. Мета дискримінатора - визначити, чи є ці дані справжнім.

Виконання дій в GAN:

1. Генератор отримує рандомну інформацію.
2. Ця генерована інформація подається до дискримінатора поряд з потоком іншої інформації, - взятої з фактичного набору даних.
3. Дискримінатор приймає як реальну, так і підроблену інформацію і повертає ймовірності числа від 0 до 1, причому 1 являє собою справжню (істину) інформацію, а 0 - представляє фальшиву.

GAN - є звичайною базою за своєю суттю. З розвитком технологій, модернізувався підхід до його використання.

Іншими різновидами GAN є: PGGAN, TWINGAN, BIGAN, AnoGAN, та ін.

Таблиця 3.3.

Порівняльна таблиця алгоритмів GAN

	Рівні покращення	Направленість	Особливість
GAN	Основний рівень (без покращень)	Аналіз даних	Звичайна взаємодія структурних елементів
PGGAN	Другий рівень покращення	Аналіз фотографій через вимір даних	Для початку роботи повинні бути фото високої якості
TWINGAN	Третій рівень покращення	Зміна (обробка) фотографій	Концепт заснований на аналізі будь-яких фотографій
BIGAN	Комбінований рівень	Аналіз даних на предмет неточностей, розбіжностей, аномалій	До структурних елементів доданий кодер «E»

У табл. 3.3 проведено сисле порівняння вищезгаданих алгоритмів GAN за такими показниками (характеристиками): рівні покращення, направленість, особливість - що у свою чергу дає певні вихідні дані яким повинна буду покращена система аналізу аномалії у мережі.

### **Висновки до розділу 3**

Системи мережевого вторгнення є потужним зняряддям для атак на мережеві інфраструктури та інформаційні системи. Вони можуть спричинити серйозну шкоду, включаючи крадіжку даних, переривання роботи мережі або навіть фінансові втрати.

Дослідження та розробки в галузі виявлення аномалій та систем мережевого вторгнення є актуальними і мають великий потенціал для підвищення рівня безпеки мереж та інформаційних систем. Подальший розвиток цих технологій вимагає постійного вдосконалення методів аналізу, розуміння нових видів загроз та використання новітніх алгоритмів та технік для ефективного виявлення аномалій та реагування на них.

## РОЗДІЛ 4.

# РОЗРОБКА ТА ТЕСТУВАННЯ СИСТЕМИ ВИЯВЛЕННЯ АНОМАЛІЙ СИСТЕМИ МЕРЕЖЕВИХ ВТОРГНЕНЬ

### 4.1. Середовище та інструменти розробки

Для розробки запропонованого алгоритму використовувалась високорівнева мова програмування загального призначення Python, яка належить до інтерпретованих мов [56].

Тобто, код, написаний на мові Python, інтерпретується в момент виконання програмою-інтерпретатором без попередньої компіляції. Це означає, що ви можете написати код на Python, зберегти його у текстовому файлі з розширенням .py і виконати його, передаючи його програмі-інтерпретатору Python.

У процесі виконання програми, програма-інтерпретатор читає рядки коду по одному, інтерпретує їх та виконує відповідні дії. Цей процес відбувається в режимі "зчитування-виконання", де кожен рядок коду інтерпретується безпосередньо перед його виконанням.

Перевагою інтерпретованої мови, такої як Python, є можливість швидкого змінення та виконання коду без необхідності компіляції. Це спрощує процес розробки та дозволяє легко експериментувати з кодом, вносити зміни та відлагоджувати програму без зайвих кроків компіляції.

Однак, важливо відзначити, що інтерпретовані мови, зазвичай, мають меншу швидкодію порівняно з компільованими мовами. Це пов'язано з тим, що інтерпретатор має витрати на кожен інтерпретацію рядка коду, тоді як компіляція зазвичай перетворює весь код у машинний код заздалегідь, що дозволяє йому виконуватися швидше.

У Python існує можливість компіляції коду в байт-код з використанням модуля `py_compile`, але це необов'язкова операція, і більшість користувачів Python використовують інтерпретацію без компіляції.

Python підтримує об'єктно-орієнтований, структурний, функціональний та аспектно-орієнтований підходи до програмування. Це робить його мовою з

великою гнучкістю і дає розробникам можливість використовувати різні парадигми залежно від потреб конкретного проекту.

Об'єктно-орієнтований підхід в Python дозволяє створювати класи, об'єкти, методи та інші об'єкти, які взаємодіють один з одним. Класи описують структуру та поведінку об'єктів, а об'єкти є екземплярами класів. Це дозволяє розробникам організувати код в окремі, повторно використовувані компоненти та реалізовувати концепції спадкування, поліморфізму та інкапсуляції.

Структурний підхід в Python дозволяє організувати код за допомогою функцій, процедур та модулів. Цей підхід ставить акцент на декомпозицію програми на окремі частини, які можуть виконувати певні завдання.

Функціональне програмування в Python базується на ідеї функцій як об'єктів першого класу. Ви можете використовувати функції як аргументи інших функцій, повертати їх як результати, а також використовувати функціональні конструкції, такі як `map`, `filter`, `reduce`, `lambda`-функції та інші, для зручного та елегантного опрацювання даних.

Аспектно-орієнтоване програмування (АОП) в Python реалізується за допомогою сторонніх бібліотек, таких як `AspectLib` або `PyAOP`. АОП дозволяє відокремлювати характеристики, які перетинаються декількома частинами програми, в окремі аспекти. Це дозволяє зменшити повторення коду та полегшити внесення змін до програми.

Python надає багато можливостей для використання різних підходів до програмування і працює як мультипарадигмова мова, що дозволяє розробникам вибирати найбільш підходящі інструменти та стилі програмування для своїх проектів.

У Python використовується динамічна типізація. Тобто тип змінної визначається момент присвоювання значення. При зміні значення може змінюватися тип даних.

Python підтримує функціональне програмування і має деякі особливості, які спільні з мовами сімейства Lisp:

- функції вищих систем `filter`, `map` і `reduce`;
- генератори списків (`list comprehensions`);

- генераторні вирази;
- множинність.

Python є однією з найпопулярніших мов програмування для розробки систем машинного навчання та штучного інтелекту. Велика кількість бібліотек та інструментів, доступних для Python, робить його привабливим вибором для реалізації проектів у цих галузях.

Два з найвідоміших і поширених інструменти для машинного навчання та штучного інтелекту у Python - це TensorFlow і scikit-learn.

TensorFlow є відкритою бібліотекою машинного навчання, розробленою компанією Google. Вона надає потужні інструменти для побудови та навчання нейронних мереж, включаючи глибоке навчання. TensorFlow забезпечує широкий спектр функціональності для обробки даних, візуалізації, побудови моделей та їх оптимізації.

Scikit-learn є іншою популярною бібліотекою машинного навчання для Python. Вона надає інструменти для класифікації, регресії, кластеризації, вимірювання якості моделей, підбору гіперпараметрів та багато іншого. Scikit-learn є потужним інструментом для широкого спектра задач машинного навчання і має простий та зрозумілий інтерфейс.

Ці бібліотеки, разом з іншими розширеннями та інструментами Python, роблять його основним вибором для розробки і реалізації алгоритмів машинного навчання та штучного інтелекту.

Наприклад, бібліотека з відкритим вихідним кодом TensorFlow, створена дослідницькою командою Google Brain, написана за допомогою Python. Google використовує цю бібліотеку для програмування та навчання нейронних мереж, які використовуються для вивчення штучного інтелекту.

Ще одна відома бібліотека - Scikit-learn. Бібліотека scikit-learn написана на мові Python з включеннями Cython.

Cython - це розширення мови Python, яке дозволяє писати код з міксом Python та C. Воно надає можливість використовувати статичний типізований код, який може бути скомпільований в C-код для поліпшення продуктивності.

Cython дозволяє розширити можливості Python, додавши швидкісність та компіляцію певних частин коду.

Scikit-learn використовує Cython для написання деяких критичних частин своєї бібліотеки. Це дозволяє отримати покращену продуктивність, особливо для обчислювально важливих операцій, таких як обробка даних, розрахунків метрик, підбір параметрів моделі та інші.

Використання Cython у scikit-learn допомагає забезпечити більшу швидкість в порівнянні з виключно інтерпретованою Python, що робить бібліотеку ефективнішою для обробки великих обсягів даних та складних моделей машинного навчання.

TensorFlow - це бібліотека, розроблена Google Brain Team для прискорення машинного навчання, у тому числі, нейронних мереж. Чудово працює на обраному Python [56].

Архітектура Tensorflow умовно ділиться на такі етапи:

1. Етап попередньої обробки даних;
2. Етап побудови моделі;
3. Етап навчання та/або оцінки моделі.

Іншими словами - ми повинні скласти граф операцій, потім передати до цього графу дані і дати команду провести обчислення.

Tensorflow приймає вхідні дані у вигляді багатовимірного масиву.

Цей масив називається також відомого як тензор.

Тензор - це вектор або матриця n-вимірювань, що являє собою всі види даних. Всі значення в тензорі містять ідентичний тип даних із відомою (або частково відомою) формою. Форма даних - це розмірність матриці чи масиву.

Дії які слід виконати з цим тензором слід викласти у певній послідовності - блок-схемі, яка називається графом (графіком).

Тензор може бути отриманий із вхідних даних або результатів обчислень. У TensorFlow всі операції проводяться всередині графіка.

Графік - набір обчислень, які відбуваються послідовно. Кожна операція називається операційним вузлом та пов'язана один з одним.

Тензор проходить увесь шлях графіку, за наслідком чого, ми можемо отримати результат.

На графіку показані операції та зв'язки між вузлами. Проте він не відображає значення. Край вузлів - це тензор, тобто. спосіб заповнення операції даними.

Бібліотека Scikit-Learn дозволяє здійснювати попередню обробку даних, зменшувати розмірності, обирати моделі, проводити регресійний та кластерний аналізи, тощо.

Попередня обробка даних потрібна задля того щоб задати певні значення усієї сукупності даних. Чи, наприклад, для того щоб опрацювати дані за певним критерієм або значенням.

Зменшення розмірності включає вибір або вилучення найбільш важливих компонентів (ознак) багатовимірного набору даних. Scikit-learn пропонує кілька підходів до зменшення розмірності. Одним із них є аналіз основних компонентів (РСА), який ми також використовуємо як основу нашої покращеної системи.

Моделі потрібні для безпосереднього машинного навчання. Відтак, усю сукупність розподіляєм за певними якостями, які поетапно впроваджуємо до системи, система звикає та подальшому вже ніби сама сприймає показники як потрібно користувачу.

## 4.2. Архітектура та зовнішній вигляд розробленої системи виявлення аномалій

Для розробленої системи виявлення аномалій продемонструємо запуск та зовнішній вигляд покращеної системи (рис. 4.1).

```
python3 main.py bigan kdd run --nb_epochs=10 --w=0.1 --m=cross-e --d=1 --nc=28
```

```
## ## ##### ## ####
### ## ## #### ## ##
##### ## ## ## ##
##### ## ##### ## ##
## ## ## ## ## ## ##
## ## ## ## ## ## ##
## ## ## ## ## ##
```

Рис. 4.1. Виконання команди запуску програми

При реалізації покращеної системи виявлення аномалій визначмо початкові параметри запуску, які представлено на рис. 4.2.

```
Batch size: 50
Starting learning rate: 1e-05
EMA Decay: 0.9999
Weight: 0.1
Method for discriminator: cross-e
Degree for L norms: 1
```

Рис. 4.2. Параметри запуску програми

Вище вже частково було наведена робота окремих структурних елементів покращеної системи виявлення аномалій.

Нижче (рис 4.3) буде продемонстрована робота кожного окремого елемента, та робота єдиною системою.

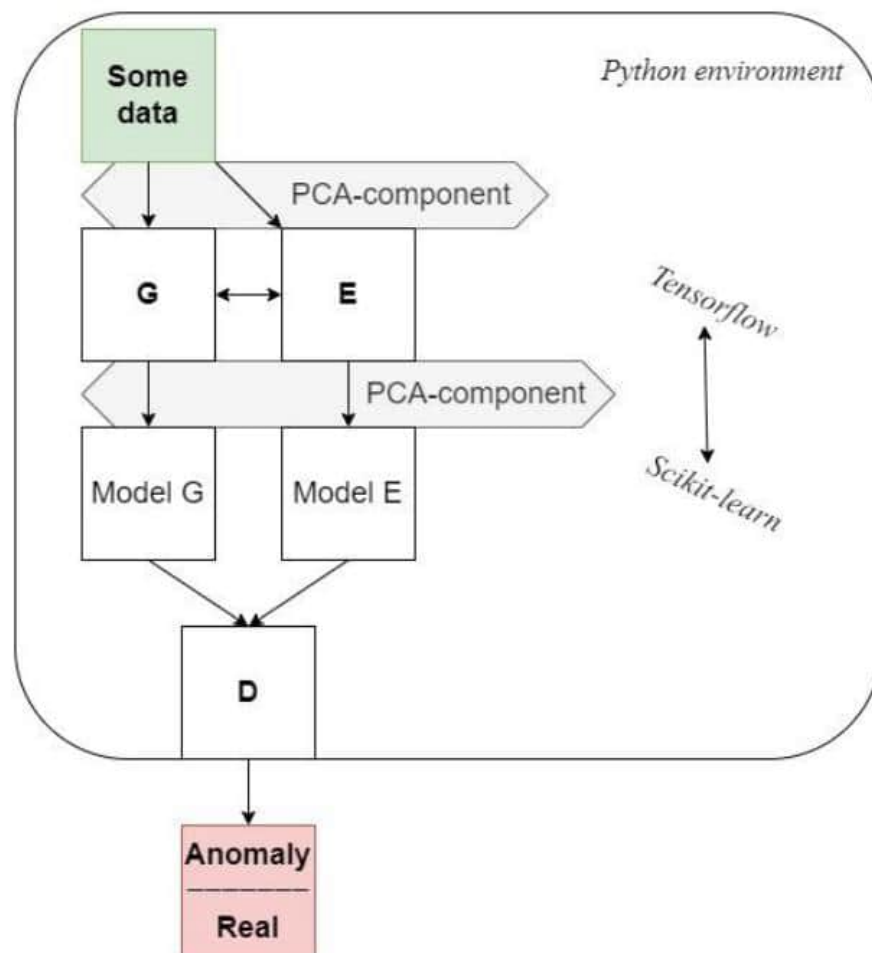


Рис. 4.3. Принцип роботи покращеної системи пошуку аномалій в мережі

Перейдемо до більш детального розгляду конкретного етапу:

1. Все починається з входу будь-якої інформації (даних, тощо) - значення Some Data.
2. Перед тим як акумулювати, чи впроваджувати дії з цією інформацією її слід опрацювати. Алгоритм PCA вправно справиться з розшаруванням інформації.
3. Після первинної обробки PCA, інформація розподіляється для Генератора (G) та кодеру (E)
4. Робота G - згенерувати, E навчити та опрацювати інформацію.
5. Після G та E маємо два різних масиви опрацьованих даних, які слід знов розподілити за критеріями.
6. Вторинна обробка PCA.
7. Після того як з інформацією виконувались дії покликані на її опрацювання вона формується за певною моделлю, яку буде співставляти D.
8. Робота D на кінцевому етапі виміряти чи є інформація яка до нього надійшла аномалією чи ні.

Окремо слід вказати, що зазвичай до D надходить ще і реальна (актуальна, істина) інформація, яка не є аномалією, однак, тут резюмується, що іде етап реал-тайм ініціювання.

А загалом окрім Моделей G та E до D потрапляє і активна інформація для виміру та співставлення.

Беручи до уваги що вся діяльність здійснюється у мережах коду Python, то відповідні структури (Tensorflow та Sciklit-learn) вмикаються за потреби та супроводжують процес від початку і до самого кінця.

Вони працюють майже за тих же обставин, що і PCA, однак остання виконує конкретні дії по обробці інформації.

Для зберігання датасету та результату роботи системи використовується база даних MySQL.

### 4.3. Проведення тестування алгоритму, взятого за основу розробленої системи виявлення аномалій

Для тестування алгоритму, взятого за основу покращеної системи пошуку аномалій на основі штучного інтелекту, було обрано набір MNIST.

**Mnist** – перероблена підбірка оригінального датасету Mnist за класифікацією цифр. Всі зображення були центровані, після чого випадкові 100 пікселів було оголошено ознаками. 6903 цифри нуль вважаються нормальними даними; 700 випадкових зображень цифри шість – аномальними. Перед поділом на навчальну (5000 об'єктів) і тестову вибірку датасет був перемішаний.

Набір даних MNIST є популярним набором для тестування та розробки алгоритмів машинного навчання в області комп'ютерного зору. Він складається з рукописних цифр від 0 до 9, що представлені у вигляді чорно-білих зображень розміром 28x28 пікселів.

Цей набір даних зазвичай використовується для завдань класифікації, де основною метою є навчання моделі розпізнавати рукописні цифри. Однак, ви можете використовувати набір даних MNIST для тестування вашого алгоритму пошуку аномалій.

У контексті пошуку аномалій, ви можете використати набір даних MNIST як "нормальні" зразки, а потім намагатися виявити аномалії або відхилення від цих нормальних зразків. Наприклад, ви можете навчити модель на нормальних зразках MNIST і потім використовувати її для виявлення неправильно розпізнаних або важких для класифікації зображень.

Важливо пам'ятати, що хоча набір даних MNIST є популярним для класифікації, використання його для пошуку аномалій може вимагати додаткових зусиль. Вам можуть знадобитися методи аномалійного виявлення, такі як автоенкодері або однокласові методи, які визначають нормальний простір і виявляють відхилення від нього.

Загалом, використання набору даних MNIST для тестування алгоритму пошуку аномалій може бути цікавим варіантом, але вам необхідно зрозуміти, як

саме ви будете визначати аномалії та як вони відрізняються від нормальних зразків.

Набір MNIST - це велика колекція рукописних цифр.

Виходячи з того, що першочергово GAN покликаний аналізувати зображення, MNIST, якраз і містить у собі дані у вигляді рукописних зображень (рис 4.4.).



Рис 4.4. Набір рукописних символів наведених у MNIST

MNIST часто використовують для тестування алгоритмів машинного навчання.

MNIST - це скорочення від модифікованої бази даних Національного інституту стандартів та технологій.

За різними оцінками MNIST містить колекцію від 50 000 - до 70 000 зображень 28 x 28 рукописних цифр від 0 до 9.

Крім цього, дані вже поділено на набори для навчання та тестування.

Створимо 10 різних наборів даних від MNIST, послідовно створюючи кожен клас цифр аномалією та розглядаючи решту 9 цифр як звичайні приклади.

Тож, тренувальний набір складається з 80% нормальних даних, а тестовий набір складається з решти 20% нормальних даних і всіх аномальних даних.

Набір даних незбалансований, тому слід порівняти моделі, використовуючи площу під кривою точністю відкликання (AUPRC).

Тестування проводимо серед згаданих вище за текстом AnoGAN та варіаційного автоенкодера (VAE).

На малюнки нижче (рис. 4.5.) можливо спостерігати, що наша модель значно перевершує базовий рівень автоенкодера VAE.

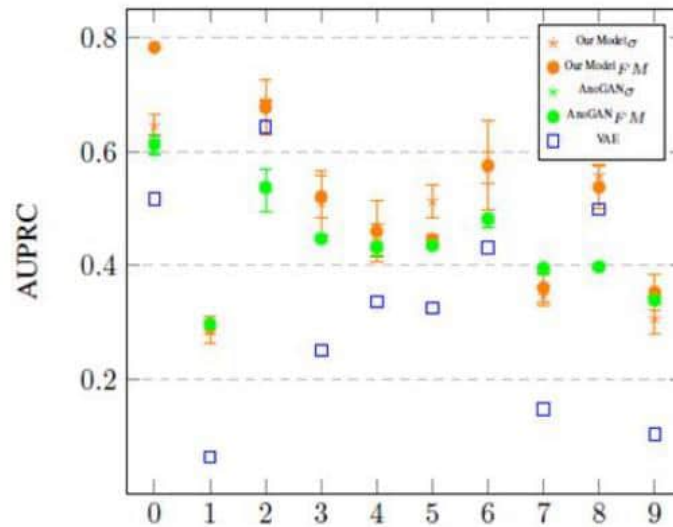


Рис. 4.5. Результати тестів

Аналогічно, покращена модель перевершує AnoGAN, але приблизно в 800 разів швидший час отримання висновку (рис. 4.7. нижче).

Operation	Kernel	Strides	Features Maps/Units	BN?	Non Linearity
<i>E(x)</i>					
Convolution	3 × 3	1 × 1	32	×	Linear
Convolution	3 × 3	2 × 2	64	✓	Leaky ReLU
Convolution	3 × 3	2 × 2	128	✓	Leaky ReLU
Dense			200	×	Linear
<i>G(z)</i>					
Dense			1024	✓	ReLU
Dense			7*7*128	✓	ReLU
Transposed Convolution	4 × 4	2 × 2	64	✓	ReLU
Transposed Convolution	4 × 4	2 × 2	1	×	Tanh
<i>D(x)</i>					
Convolution	4 × 4	2 × 2	64	×	Leaky ReLU
Convolution	4 × 4	2 × 2	64	✓	Leaky ReLU
<i>D(z)</i>					
Dense			512	×	Leaky ReLU
<i>Concatenate D(x) and D(z)</i>					
<i>D(x,z)</i>					
Dense*			1024	×	Leaky ReLU
Dense			1	×	Sigmoid
Optimizer	Adam( $\alpha = 10^{-5}$ , $\beta_1 = 0.5$ )				
Batch size	100				
Latent dimension	200				
Epochs	100				
Leaky ReLU slope	0.1				
Weight, bias initialization	Isotropic gaussian ( $\mu = 0$ , $\sigma = 0.02$ ), Constant(0)				

Рис. 4.6. Архітектура та параметри BIGAN у MNIST

Також, можливо прийти до висновку, що при використанні варіант LD (з відповідністю ознак в оцінці аномалії) - працює краще, ніж перехресно-ентропійний варіант, про який вже згадувалось.

Наведений вище рисунок репрезентує використану архітектуру та параметри BIGAN під час проведення тестів за допомогою MNIST.

Для зрівняння слід навести архітектура та параметри GAN у MNIST:

Operation	Kernel	Strides	Features Maps/Units	BN?	Non Linearity
<i>G(z)</i>					
Dense			1024	✓	ReLU
Dense			7*7*128	✓	ReLU
Transposed Convolution	4 × 4	2 × 2	64	✓	ReLU
Transposed Convolution	4 × 4	2 × 2	1	×	Tanh
<i>D(x)</i>					
Convolution	4 × 4	2 × 2	64	×	Leaky ReLU
Convolution	4 × 4	2 × 2	64	✓	Leaky ReLU
Dense*			1024	✓	Leaky ReLU
Dense			1	×	Sigmoid
Optimizer	Adam( $\alpha = 10^{-3}$ , $\beta_1 = 0.5$ )				
Batch size	100				
Latent dimension	200				
Epochs	100				
Leaky ReLU slope	0.1				
Weight, bias initialization	Isotropic gaussian ( $\mu = 0$ , $\sigma = 0.02$ ), Constant(0)				

Рис. 4.7. Архітектура та параметри GAN у MNIST

### 4.3. Проведення тестування та аналіз розробленої системи виявлення аномалій

Набір KDD99 було обрано для проведення апробації та отримання результатів тестів покращеної системи пошуку аномалій на основі штучного інтелекту.

KDD99 є набором даних, що використовується для вивчення проблем безпеки мережі та виявлення вторгнень (Intrusion Detection System, IDS).

KDD99 був створений в рамках проекту Knowledge Discovery and Data Mining (KDD) для оцінки та порівняння алгоритмів виявлення вторгнень. Він містить реальні дані з вторгнень в мережу, зібрані з декількох джерел, включаючи лабораторні середовища та симуляції.

Цей набір даних містить різні типи мережевих з'єднань, такі як нормальні, вторгнені та деякі типи вторгнень, такі як злам користувача, сканування портів, DoS атаки тощо. Він містить загалом 41 атрибут, включаючи IP-адресу, порт, протокол, прапорці, тривалість з'єднання та інші параметри.



час підключення тощо. Всі ці атрибути надають інформацію про характеристики та зміни, що відбуваються під час з'єднання, і використовуються для аналізу та виявлення вторгнень.

Тепер деякі з цих сполук є шкідливими. У шкідливих зразків свій унікальний відбиток пальця (унікальна комбінація різних значень функцій), яка відокремлює їх від хороших.

KDD99 також підтримує кластеризацію за k-відмінних.

В наборі даних KDD99 можна використовувати метод головних компонентів (Principal Component Analysis, PCA) для перевірки кореляції між атрибутами або для зменшення розмірності даних.

PCA є статистичним методом, який перетворює вихідний набір даних в новий набір компонентів (головних компонентів) шляхом лінійної комбінації початкових атрибутів. Головні компоненти відображають напрямок максимальної дисперсії даних і впорядковані за спаданням дисперсії.

За допомогою PCA можна виявити кореляцію між атрибутами, оцінити внесок кожного атрибута в змінність даних та зменшити розмірність набору даних, залишаючи при цьому більш значущі атрибути.

Таким чином можливо зменшити деякі розміри даних.

Після PCA ваші дані матимуть простішу уяву (з меншою кількістю вимірювань) і можуть дати кращу продуктивність та результати.

Треба зазначити, що для експерименту використовувався лише 10-відсотковий набір даних KDD99. Якщо використовувати експериментальну установку на 10-відсотковому наборі даних KDD99, це означає, що відібралося лише 10% випадково обраних зразків (з'єднань) з повного набору даних KDD99 для проведення тестів і експериментів.

Завдяки пропорції викидів у наборі даних «звичайні» дані розглядаються як аномалії за цих обставин.

Ще 20% зразків з найвищими оцінками аномалії  $A(x)$  класифікуються як аномалії (позитивний клас).

Для навчання було обрано випадковим чином відібрали 50% всього набір даних, а решта 50% набору даних було використано для тестування.

А отже, лише зразки даних із нормальним (хорошим, істинним) класом використовувалися для навчання системи, тому всі аномальні вибірки були вилучені з роздільного тренувального набору.

Цей покращений метод загалом дуже конкурентоспроможний з іншими найсучаснішими методами і досягає більш високого запам'ятовування (рис. 4.9).

Operation	Units	Non Linearity	Dropout
<i>E(x)</i>			
Dense	64	Leaky ReLU	0.0
Dense	32	Linear	0.0
<i>G(z)</i>			
Dense	64	ReLU	0.0
Dense	128	ReLU	0.0
Dense	121	Linear	0.0
<i>D(x)</i>			
Dense	128	Leaky ReLU	0.2
<i>D(z)</i>			
Dense	128	Leaky ReLU	0.2
<i>Concatenate D(x) and D(z)</i>			
<i>D(x,z)</i>			
Dense*	128	Leaky ReLU	0.2
Dense	1	Linear	0.0
Optimizer	Adam( $\alpha = 10^{-5}$ , $\beta_1 = 0.5$ )		
Batch size	50		
Latent dimension	32		
Epochs	50		
Leaky ReLU slope	0.1		
Weight, bias initialization	Xavier Initializer, Constant(0)		

Рис. 4.9. Архітектура та параметри BIGAN у KDD99

На рис. 4.10 наводиться архітектура та параметри GAN у KDD99 для порівняння результатів.

Operation	Units	Non Linearity	Dropout
<i>G(z)</i>			
Dense	64	ReLU	0.0
Dense	128	ReLU	0.0
Dense	121	Linear	0.0
<i>D(x)</i>			
Dense	256	Leaky ReLU	0.2
Dense	128	Leaky ReLU	0.2
Dense*	128	Leaky ReLU	0.2
Dense	1	Sigmoid	0.0
Optimizer	Adam( $\alpha = 10^{-5}$ , $\beta_1 = 0.5$ )		
Batch size	50		
Latent dimension	32		
Epochs	50		
Leaky ReLU slope	0.1		
Weight, bias initialization	Xavier Initializer, Constant(0)		

Рис. 4.10. Архітектура та параметри GAN у KDD99

Наведений вище рисунок репрезентує використану архітектуру та параметри BIGAN під час проведення тестів за допомогою KDD99.

Знову ж таки, наша модель перевершує AnoGAN, а також має від 700х до 900х швидший час висновку (рис. 4.11.).

Dataset	$L_D$	AnoGAN	Our Model	Speed Up
MNIST	$\sigma$	6657	8	~830
	FM	7260	9.4	~770
KDD	$\sigma$	2578	2.7	~950
	FM	3527	5,3	~660

Рис. 4.11. Результати отриманих показників тестів даної системи MNIST та KDD99

Підсумовуючи проведені тести, звірки архітектурі та використані алгоритми вбачається, що розроблений модифікований алгоритм BiGAN (має кращі показники між інші алгоритми - AnoGAN та GAN.

#### Висновки до розділу 4

У даному розділі було розглянуто інтерфейс розробленої системи виявлення аномалій та наведено алгоритм і його принцип роботи.

Вказано, що останні моделі GAN можна використовувати для досягнення найсучаснішої продуктивності для виявлення аномалій складних наборах даних мереж.

Аналіз програми на перших етапах підвернений великій похибці, однак з часом - та безпосереднім аналізом все більшої кількості даних (елементів, інформації) - здатний на великі, точні та швидкі результати.

А конкретно, під час тестування; використання GAN, який одночасно вивчає кодер (BIGAN), усуває необхідність у дороговартісній процедурі щоб відновити латентне представлення для заданого входу.

Крім цього, була виявлена тенденція, що покращена система відповідає вимогам продуктивності та швидкоплинності - одним з найважливішим критеріям систем виявлення аномалій.

Отримані показники можливо використати в подальшому для ще більше детальної оцінки покращеного методу, оцінки інших стратегій навчання, а також дослідження точності кодера на продуктивність виявлення аномалій.

## ВИСНОВКИ

Розвиток сучасної сфери інформаційних технологій призводить до зростання загроз і уразливостей, які створюють можливості для здійснення комп'ютерних атак з боку зловмисників.

Мережева безпека є надзвичайно важливою складовою для забезпечення безпеки і захищеності інформаційних систем та мереж. Системи виявлення атак (Intrusion Detection Systems - IDS) є ключовим елементом в цьому процесі, оскільки вони дозволяють виявляти та реагувати на підозрілу та шкідливу активність в мережах.

Виявлення атак передбачає аналіз мережевого трафіку та інших відомостей для виявлення знаків аномальної або зловмисної діяльності. Це може включати аналіз пакетів даних, системних журналів, а також використання сигнатурних баз даних та алгоритмів машинного навчання для виявлення відомих та нових типів атак.

Важливо мати систему виявлення атак, яка може ідентифікувати ці типи атак та вживати відповідних заходів для захисту мережі. Усунення вразливостей, моніторинг мережевого трафіку та виявлення підозрілої активності є критичними етапами у захисті мереж та інформаційних систем від атак і порушень безпеки.

Виявлення аномалій у мережі є важливою складовою безпеки мережевих інфраструктур. Це дозволяє вчасно виявляти та реагувати на незвичайну активність, яка може свідчити про потенційну загрозу або вторгнення.

Існують різні методи та технології виявлення аномалій, такі як сигнатурний аналіз, аналіз вимог, машинне навчання та системи виявлення та запобігання вторгнень. Кожен з цих методів має свої переваги та обмеження і може бути використаний для розробки ефективних систем виявлення аномалій.

Системи мережевого вторгнення є потужним знаряддям для атак на мережеві інфраструктури та інформаційні системи. Вони можуть спричинити серйозну шкоду, включаючи крадіжку даних, переривання роботи мережі або навіть фінансові втрати.

Дослідження та розробки в галузі виявлення аномалій та систем мережевого вторгнення є актуальними і мають великий потенціал для підвищення рівня безпеки мереж та інформаційних систем. Подальший розвиток цих технологій вимагає постійного вдосконалення методів аналізу, розуміння нових видів загроз та використання новітніх алгоритмів та технік для ефективного виявлення аномалій та реагування на них.

У роботі запропоновано покращений метод виявлення аномалій в мережі, який за допомогою врахування особливостей використаних структурних елементів скорочує час, необхідний для віднаходження аномалій та загалом спрощує процес аналізу даних.

За рахунок функціоналу системи – участь користувача у її роботі мінімізована, а неточність результатів завдяки алгоритмам – зменшена. Запропонована покращена система надає можливість забезпечити актуальне та вчасне виявлення неточностей та аномалій у потоці даних мережі. Подальше вдосконалення системи виявлення аномалій пов'язано із застосуванням методів, підходів, та теорії природи аномалій. Просування саме в такому напрямку використання отриманих знань зменшить частоту аномалій ще на шляху до їх активізації, що є безумовною позитивною якістю, яка зменшить наступні витрати на знешкодження наслідків аномалій.

**СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ**

1. Методи аналізу та моделювання безпеки розподілених інформаційних систем. Навчальний посібник / В.В. Литвинов, В.В. Казимир, І.В. Стеценко [та ін.]. – Чернігів: Чернігівський національний технологічний університет, 2016. – 254 с.
2. Denning D. An Intrusion Detection Model / D. Denning // IEEE Transactions on Software Engineering. – 1987. – Vol. SE-13, N 1. – P. 222 – 232.
3. М. В. Грайворонський, О. М. Новіков. Безпека інформаційно-комунікаційних систем — 2009. — 608 с.
4. Intrusion Detection and Prevention Systems, Karen Scarfone, Peter Mell, Handbook of Information and Communication Security pp 177-192.
5. Michael Sikorski, Andrew Hoing. Practical Malware Analysis. - No Strach Press, 2018. - 802 с.
6. Stefan Axelsson. Intrusion Detection Systems: A Survey and Taxonomy. - Department of Computer Engineering Chalmers University of Technology G 'oteborg, Sweden, 2015. - 27 с.
7. Next Generation Intrusion Detection Expert System (NIDES) / D. Anderson [et al] // Software Design, Product Specification and Version Description Document, Project 3131. – 1994. – July 11. – P. 1 – 94.
8. Киричек, Г. Г.; Гаркуша, В. Ю. Процес виявлення зловживань і аномалій в мережі. Наукові праці Донецького національного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка, 2019, 1-2: 42-46.
9. Толкаченко, Є. А., Данилюк, В. С., Семенюк, М. О., Фльора, А. С. Огляд сучасних методів в системах виявлення вторгнень для потреб інформаційно-телекомунікаційних систем спеціального призначення. Водний транспорт, 2021, 57-61.
10. Kwon, YooJin, et al. Behavior analysis and anomaly detection for a digital substation on cyber-physical system. Electronics, 2019, 8.3: 326.
11. Wang, Ruoying, et al. Deep learning for anomaly detection. In: Proceedings of the 13th international conference on web search and data mining. 2020. p. 894-896/

12. Ahmed, Mohiuddin; Mahmood, Abdun Naser; HU, Jiankun. A survey of network anomaly detection techniques. *Journal of Network and Computer Applications*, 2016, 60: 19-31.
13. Garcia-Teodoro, Pedro, et al. Anomaly-based network intrusion detection: Techniques, systems and challenges. *computers & security*, 2009, 28.1-2: 18-28.
14. Завада, А.; Самчишин, О.; Охрімчук, В. Аналіз сучасних систем виявлення атак і запобігання вторгненням. *Інформаційні системи*, 2012, 97-106.
15. Мохнін, М.І.; Святошенко, В. О. Розробка системи виявлення вторгнення в реальному часі для забезпечення безпеки комп'ютерних мереж. 2017.
16. Yeung, Dit-Yan; Chow, Calvin. Parzen-window network intrusion detectors. In: *2002 International Conference on Pattern Recognition. IEEE*, 2002. p. 385-388.
17. Василюшин, В. В.. Ідентифікація аномалій мережевого трафіку з використанням нейронних мереж. 2022. Bachelor's Thesis. ТНТУ.
18. Berkovsky, V. V.; Bessonov, A. S. Аналіз та класифікація методів виявлення вторгнень в інформаційну систему. *Системи управління, навігації та зв'язку. Збірник наукових праць*, 2017, 3.43: 57-62.
19. Bhuyan, Monowar H.; Bhattacharyya, Dhruva Kumar; Kalita, Jugal K. Network anomaly detection: methods, systems and tools. *Ieee communications surveys & tutorials*, 2013, 16.1: 303-336.
20. Fotiadou, K.; Velivassaki, T.-H.; Voulkidis, A.; Skias, D.; Tsekeridou, S.; Zahariadis, T. Network Traffic Anomaly Detection via Deep Learning. *Information* 2021, 12, 215. <https://doi.org/10.3390/info12050215>
21. Sedik, Ahmed, et al. Efficient anomaly detection from medical signals and images. *International Journal of Speech Technology*, 2019, 22: 739-767.
22. Moustafa, Nour; HU, Jiankun; Slay, Jill. A holistic review of network anomaly detection systems: A comprehensive survey. *Journal of Network and Computer Applications*, 2019, 128: 33-55.
23. Hooshmand, Mohammad Kazim; Hosahalli, Doreswamy. Network anomaly detection using deep learning techniques. *CAAI Transactions on Intelligence Technology*, 2022, 7.2: 228-243.

24. Mishra, P.; Varadharajan, V.; Tupakula, U.; Pilli, E.S. A detailed investigation and analysis of using machine learning techniques for intrusion detection. *IEEE Commun. Surv. Tutor.* 2018, 21, 686–728.
25. OGBECHIE, Alberto, et al. Dynamic Bayesian network-based anomaly detection for in-process visual inspection of laser surface heat treatment. In: *Machine Learning for Cyber Physical Systems: Selected papers from the International Conference ML4CPS 2016*. Springer Berlin Heidelberg, 2017. p. 17-24.
26. Garg, Sahil, et al. A hybrid deep learning-based model for anomaly detection in cloud datacenter networks. *IEEE Transactions on Network and Service Management*, 2019, 16.3: 924-935.
27. Fernandes, Gilberto, et al. A comprehensive survey on network anomaly detection. *Telecommunication Systems*, 2019, 70: 447-489.
28. Evangelou, M., & Adams, N. M. (2020). An anomaly detection framework for cybersecurity data. *Computers & Security*, 97, 101941. <https://doi.org/10.1016/j.cose.2020.101941>.
29. Chandola, Varun; Banerjee, Arindam; Kumar, Vipin. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 2009, 41.3: 1-58.
30. Smiti, A. (2020). A critical overview of outlier detection methods. *Computer Science Review*, 38, 100306. <https://doi.org/10.1016/j.cosrev.2020.100306>.
31. Liu, Hongyu; Lang, Bo. Machine learning and deep learning methods for intrusion detection systems: A survey. *applied sciences*, 2019, 9.20: 4396.
32. K. Park, Y. Song, and Y. G. Cheong, “Classification of attack types for intrusion detection systems using a machine learning algorithm,” in *Proceedings of the 2018 IEEE Fourth International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 282–286, Bamberg, Germany, March 2018.
33. Xiang, Ling, et al. Condition monitoring and anomaly detection of wind turbine based on cascaded and bidirectional deep learning networks. *Applied Energy*, 2022, 305: 117925.

34. Aldweesh, Arwa; Derhab, Abdelouahid; Emam, Ahmed Z. Deep learning approaches for anomaly-based intrusion detection systems: A survey, taxonomy, and open issues. *Knowledge-Based Systems*, 2020, 189: 105124.
35. Pang, Guansong, et al. Deep learning for anomaly detection: A review. *ACM computing surveys (CSUR)*, 2021, 54.2: 1-38.
36. Boukerche, A., Zheng, L., & Alfandi, O. (2020). Outlier detection: Methods, models, and classification. *ACM Computing Surveys (CSUR)*, 53(3), 1-37.
37. Jain, Meenal; Kaur, Gagandeep. Distributed anomaly detection using concept drift detection based hybrid ensemble techniques in streamed network data. *Cluster Computing*, 2021, 24: 2099-2114.
38. Kwon, D.; Kim, H.; Kim, J.; Suh, S.C.; Kim, I.; Kim, K.J. A survey of deep learning-based network anomaly detection. *Clust. Comput.* 2019, 22, 949–961.
39. Gupta, R.; Tanwar, S.; Tyagi, S.; Kumar, N. Machine learning models for secure data analytics: A taxonomy and threat model. *Comput. Commun.* 2020, 153, 406–440.
40. Gupta, R.; Tanwar, S.; Tyagi, S.; Kumar, N. Machine learning models for secure data analytics: A taxonomy and threat model. *Comput. Commun.* 2020, 153, 406–440.
41. D. C. Le and N. Zincir-Heywood, “Anomaly detection for insider threats using unsupervised ensembles,” *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, pp. 1152–1164, 2021.
42. Z-Score. Короткий огляд. Режим доступу: <http://surl.li/aswqs>
43. Scikit Learn - популярній бібліотека Python [Введення в Scikit-learn](#). Режим доступу: <http://surl.li/aswrk>
44. Як працює метод основних компонентів (PCA). Режим доступу: <http://surl.li/aswrv>
45. R. Patil, R. Biradar, V. Ravi, P. Biradar, and U Ghosh, “Network traffic anomaly detection using PCA and BiGAN,” *Internet Technology Letters*, vol. 5, no. 1, p. e235, 2022.
46. Генеративно-змагальна нейромережа (GAN) Режим доступу: <http://surl.li/aswsc>

47. Houssam Zenati, Chuan-Sheng F, Bruno Lecouat, GauravManek, Vijay Ramaseshan Chandrasekhar / «Efficient gan-based anomaly detection» // Workshop track - ICLR 2018.

48. PGGAN - прогресивна генеративна неймережа від Nvidia. Режим доступу: <http://surl.li/aswsw>

49. Комплексна платформа машинного навчання з відкритим кодом. TensorFlow. Режим доступу: <https://www.tensorflow.org/>

50. Характеристика TensorFlow. Режим доступу: <http://surl.li/aswtj>

51. Python. Режим доступу: <https://www.python.org/>

52. Набір даних MNIST в Python - базовий імпорт та побудова / [Режим доступу - електронне джерело]: <http://surl.li/aswul>

53. База даних MNIST із рукописними цифрами. Режим доступу: <http://surl.li/aswum>

54. KDD Cup 1999 Data. Режим доступу: <http://surl.li/aswus>

55. The Significant Features of the UNSW-NB15 and the KDD99 Data Sets for Network Intrusion Detection Systems. Режим доступу: <http://surl.li/aswuw>

56. Analysis and preprocessing of the kdd cup 99 dataset using python and scikit-learn. Режим доступу: <https://github.com/timeamagyar/kdd-cup-99-python>

## ДОДАТКИ

## Додаток А

## Лістинг програми

```

import argparse as ap
import importlib as il
import logging
import os
import shutil
import urllib3
import zipfile

import data

console = logging.StreamHandler()
console.setLevel(logging.INFO)
console.setFormatter(logging.Formatter('%(asctime)s %(levelname)-3s
@%(name)s] %(message)s', datefmt='%H:%M:%S'))
logging.basicConfig(level=logging.DEBUG, handlers=[console])
logging.getLogger("tensorflow").setLevel(logging.WARNING)
logger = logging.getLogger("AnomalyDetection")

def run(args):
    print("""
    ## ## ##### ## ###
    ### ## ## #### ## ##
    ##### ## ## ## ## ##
    ##### ## ##### ## ##
    ## ## ## ## ## ## ##
    ## ## ## ## ## ## ##
    ## ## ## ## ## #####
    """)

    has_effect = False

    if args.example and args.dataset and args.split:
        try:
            mod_name = "{}-{}_{}".format(args.example, args.split, args.dataset)
            logger.info("Running script at {}".format(mod_name))

            mod = il.import_module(mod_name)
            mod.run(args.nb_epochs, args.w, args.m, args.d, args.label, args.nc, args
                .rd)

        except Exception as e:
            logger.exception(e)
            logger.error("Error has occured while running the script.")
    else:
        if not has_effect:
            logger.error("Script halted without any effect.")

```

```

def path(d):
    try:
        assert os.path.isdir(d)
        return d
    except Exception as e:
        raise ap.ArgumentTypeError("Example {} cannot be located.".format(d))

if __name__ == "__main__":
    parser = ap.ArgumentParser(description='Run examples from the DL 2.0 Anomaly Detector.')
    parser.add_argument('example', nargs="?", type=path, help='the folder name of the example you want to run e.g bigan')
    parser.add_argument('dataset', nargs="?", choices=['kdd'], help='the name of the dataset you want to run the experiments on')
    parser.add_argument('split', nargs="?", choices=['run'], help='train the example or evaluate it')
    parser.add_argument('--nb_epochs', nargs="?", type=int, help='number of epochs you want to train the dataset on')
    parser.add_argument('--label', nargs="?", type=int, help='anomalous label for the experiment')
    parser.add_argument('--w', nargs="?", default=0.1, type=float, help='weight for the sum of the mapping loss function')
    parser.add_argument('--m', nargs="?", default='fm', choices=['cross-e', 'fm'], help='mode/method for discriminator loss')
    parser.add_argument('--d', nargs="?", default=1, type=int, help='degree for the L norm')
    parser.add_argument('--rd', nargs="?", default=42, type=int, help='random_seed')
    parser.add_argument('--nc', nargs="?", default=0, type=int, help='Number of columns')

    run(parser.parse_args())
import tensorflow as tf
import data.kdd as data

""" KDD BiGAN architecture. """

learning_rate = 0.00001
batch_size = 50
layer = 1
latent_dim = 32
dis_inter_layer_dim = 128

init_kernel = tf.compat.v1.keras.initializers.VarianceScaling(scale=1.0, mode="fan_avg", distribution="uniform")
#shape = 21

def encoder(x_inp, is_training=False, getter=None, reuse=False):
    """
    Encoder architecture in tensorflow
    Maps the data into the latent space
    """

    with tf.compat.v1.variable_scope('encoder', reuse=reuse, custom_getter=getter):

        name_net = 'layer_1'
        with tf.compat.v1.variable_scope(name_net):
            net = tf.compat.v1.layers.dense(x_inp,
                units=64,
                kernel_initializer=init_kernel,
                name='fc')
            net = leakyRelu(net)

        name_net = 'layer_2'
        with tf.compat.v1.variable_scope(name_net):
            net = tf.compat.v1.layers.dense(net,
                units=latent_dim,
                kernel_initializer=init_kernel,
                name='fc')

    return net

```

```

def decoder(z_inp, is_training=False, getter=None, reuse=False):
    """
    Decoder architecture in tensorflow
    Generates data from the latent space
    """
    with tf.compat.v1.variable_scope('generator', reuse=reuse, custom_getter=getter):
        name_net = 'layer_1'
        with tf.compat.v1.variable_scope(name_net):
            net = tf.compat.v1.layers.dense(z_inp,
                units=64,
                kernel_initializer=init_kernel,
                name='fc')
            net = tf.nn.relu(net)

        name_net = 'layer_2'
        with tf.compat.v1.variable_scope(name_net):
            net = tf.compat.v1.layers.dense(net,
                units=128,
                kernel_initializer=init_kernel,
                name='fc')
            net = tf.nn.relu(net)

        name_net = 'layer_3'
        with tf.compat.v1.variable_scope(name_net):
            net = tf.compat.v1.layers.dense(net,
                units=data.get_shape_input()[1],
                kernel_initializer=init_kernel,
                name='fc')

    return net

def discriminator(z_inp, x_inp, is_training=False, getter=None, reuse=False):
    """
    Discriminator architecture in tensorflow
    Discriminates between pairs (E(x), x) and (z, G(z))
    """
    with tf.compat.v1.variable_scope('discriminator', reuse=reuse, custom_getter=getter):
        # D(x)
        name_x = 'x_layer_1'
        with tf.compat.v1.variable_scope(name_x):
            x = tf.compat.v1.layers.dense(x_inp,
                units=128,
                kernel_initializer=init_kernel,
                name='fc')
        x = leakyRelu(x)
        x = tf.compat.v1.layers.dropout(x, rate=0.2, name='dropout', training=is_training)

```

```

# D(z)
name_z = 'z_fc_1'
with tf.compat.v1.variable_scope(name_z):
    z = tf.compat.v1.layers.dense(z_inp, 128, kernel_initializer=init_kernel)
    z = leakyRelu(z)
    z = tf.compat.v1.layers.dropout(z, rate=0.2, name='dropout', training=is_training)

# D(x,z)
y = tf.concat([x, z], axis=1)

name_y = 'y_fc_1'
with tf.compat.v1.variable_scope(name_y):
    y = tf.compat.v1.layers.dense(y,
    dis_inter_layer_dim,
    kernel_initializer=init_kernel)
    y = leakyRelu(y)
    y = tf.compat.v1.layers.dropout(y, rate=0.2, name='dropout', training=is_training)

intermediate_layer = y

name_y = 'y_fc_logits'
with tf.compat.v1.variable_scope(name_y):
    logits = tf.compat.v1.layers.dense(y,
    1,
    kernel_initializer=init_kernel)

return logits, intermediate_layer

def leakyRelu(x, alpha=0.1, name='leaky_relu'):
    if name:
        with tf.compat.v1.variable_scope(name):
            return _leakyRelu_impl(x, alpha)
    else:
        return _leakyRelu_impl(x, alpha)

def _leakyRelu_impl(x, alpha):

    return tf.nn.relu(x) - (alpha * tf.nn.relu(-x))

```