Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

# КВАЛІФІКАЦІЙНА РОБОТА

*бакалавр*
Освітній рівень

Система виявлення вторгнень на основі нечіткого логічного висновку
Назва теми

*КвРКБ.180132.18.01.09 ПЗ*
Шифр

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *125 «Кібербезпека»*
Шифр, назва

Освітня програма *«Кібербезпека»*
Назва

Виконав: студент IV курсу, група КБ–18–1 _____ М.К. Огородник
Підпис, Ініціали, прізвище

Керівник _____ В.Ю. Тітова
Підпис, дата, Ініціали, прізвище

Нормоконтролер _____ С.В. Мостовий
Підпис, дата, Ініціали, прізвище

До захисту допускаю:
Зав. кафедри кібербезпеки _____ Ю.П. Кльоц
Підпис, Ініціали, прізвище

«16» червня 2022 р.

Хмельницький, 2022

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Кібербезпеки

Освітній рівень Бакалавр

Галузь знань 12 Інформаційні технології

Спеціальність 125 Кібербезпека

Освітня програма Освітньо-професійна програма підготовки бакалаврів

ЗАТВЕРДЖУЮ

Завідувач кафедри кібербезпеки

_____ Клюс О.Ч.

01 03 2022 р.

**ЗАВДАННЯ**
**НА КВАЛІФІКАЦІЙНУ РОБОТУ**

Огороднику Максиму Костянтиновичу

Прізвище, ім'я, по батькові студента

1 Тема роботи Система виявлення вторгнень на основі нечіткого логічного висновку

Керівник роботи к.т.н, доц. Тітова Віра Юріївна

Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджено наказом ректора університету від 01.03.2022 №18 додаток 10

2 Строк подання студентом роботи на кафедру: 3.06.2022

3 Вихідні дані до роботи визначити значних атрибутів, сформувати основи нечітких правил, розробити прототип програмного комплексу, тестування прототипу системи

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити) Виявлення вторгнень, нечіткі продукційні моделі, система виявлення вторгнень із використанням нечіткої логіки, результати роботи

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень) «», «Основні етапи роботи системи виявлення вторгнень», «Класифікація систем виявлень вторгнень», «Генетичний алгоритм».

6 Консультанти розділів курсового проекту (роботи)

| Розділ | Прізвище, ініціали та посада консультанта | Підпис, дата | |
|---|---|---|---|
| | | завдання видав | завдання прийняв |
| Нормоконтроль | Мостовий С.В., старший викладач кафедри кібербезпеки | | |
| Антиплагіат | Мостовий С.В., старший викладач кафедри кібербезпеки | | |

7 Дата видачі завдання _____ _____ 2022 р.

## КАЛЕНДАРНИЙ ПЛАН

| №з/п | Назва етапів (розділів) дипломного проекту (роботи) | Термін виконання етапів проекту (роботи) | Примітка |
|---|---|---|---|
| 1 | Вибір та затвердження теми кваліфікаційної роботи. | Січень | – |
| 2 | Аналіз об'єкта дослідження. | Січень–лютий | – |
| 3 | Проектування та розробка загальної архітектури і структури системи. | Лютий–березень | – |
| 4 | Програмна реалізація запропонованого рішення та тестування системи; аналіз результатів і оцінювання прийнятих рішень. | Квітень | – |
| 5 | Написання тексту пояснювальної записки та розробка графічних матеріалів. | Травень | – |
| 6 | Остаточне коригування кваліфікаційної роботи з урахуванням зауважень керівника. | | – |
| 7 | Оформлення кваліфікаційної роботи як документа відповідно до вимог. | | – |
| 8 | Отримання супровідних документів. Нормоконтроль. | Червень | – |
| 9 | Підготовка до захисту та захист кваліфікаційної роботи. | | – |

Студент _____ М.К. Огородник
Підпис Ініціали, прізвище

Керівник проекту (роботи) _____ В.Ю. Тітова
Підпис Ініціали, прізвище

# АНОТАЦІЯ

Тема кваліфікаційної роботи: Система виявлення вторгнень на основі нечіткого логічного висновку

Автор роботи: Огородник Максим Костянтинович

Керівник роботи: к.т.н., доц. Тітова Віра Юріївна

Пояснювальна записка: 59 с., 22 рис., 6 табл., 1 дод., 17 джерел.

Ключові слова: нечітка логіка, виявлення атак, кібербезпека, система виявлення вторгнень.

Метою кваліфікаційної роботи бакалавра є розробка системи виявлення вторгнень на основі нечіткого логічного висновку.

Представлені результати виявлення класів атак DoS, Probes, а також нормального трафіку. Крім того, в роботі запропоновано метод скорочення розмірності значущих атрибутів формування бази нечітких продукційних правил.

16.06.22

| № пилка | Формат | Позначення | Найменування | Кількість | № | Примітка |
|---|---|---|---|---|---|---|
| 1. | | | Текстові документи | | | |
| 2. | А4 | КвРКБ.180132.18.01.09 ПЗ | Пояснювальна записка | 1 | | |
| 3. | | | | | | |
| 4. | | | Графічні матеріали | | | |
| 5. | А2 | КвРКБ.180132.18.01.09 Е8 | Основні етапи роботи | 1 | | |
| 6. | | | системи виявлення | | | |
| 7. | | | вторгнень | | | |
| 8. | | | | | | |
| 9. | А2 | КвРКБ.180132.18.01.09 Е8 | Класифікація систем | 1 | | |
| 10 | | | виявлень вторгнень | | | |
| 11 | | | | | | |
| 12 | А2 | КвРКБ.180132.18.01.09 Е8 | Генетичний алгоритм | 1 | | |
| 13 | | | | | | |
| 14 | | | | | | |
| 15 | | | | | | |
| 16 | | | | | | |
| 17 | | | | | | |
| 18 | | | | | | |
| 19 | | | | | | |
| 20 | | | | | | |
| 21 | | | | | | |
| 22 | | | | | | |
| 23 | | | | | | |
| 24 | | | | | | |
| 25 | | | | | | |
| 26 | | | | | | |

КвРКБ.180132.18.01.09 ВП

| Зм. | Аркуш | № докум. | Підп. | Дата |
|---|---|---|---|---|
| Розроб. | | Огородник М.К. | | |
| Перевір. | | Тітова В.Ю. | | |
| Н. Контр. | | Мостовий С.В. | | |
| Затверд. | | Кльон Ю.П. | | |

Система виявлення вторгнень на основі нечіткого логічного висновку
Відомість проекту

| Літера | Аркуш | Аркушів |
|---|---|---|
| Н | | 1 |

ХНУ, КБ-18-1

**ЗМІСТ**

## ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

СВВ - Система виявлення вторгнень

СЗВ - Системи запобігання вторгнень

МЦФ - Моніторинг цілісності файлів

ГА - Генетичні алгоритми

КРБ - Кваліфікаційна робота бакалавра

КБ - Кібербезпека

ПЗ - Програмне забезпечення

НМ - Нейронна мережа

ПЗ - Пояснювальна записка

ПП - Програмний продукт

IDS - Intrusion Detection System

HIDS - Host-Based Intrusion Detection System

NIDS - Network intrusion detection systems

CIDF - Common Intrusion Detection Framework

SVN - Support Vector Networks

# ВСТУП

Засоби виявлення вторгнень чим далі частіше стають ключовим елементом систем захисту виявлення аномальної активності в інформаційної системі. В основному, традиційні системи виявлення вторгнень базуються на знаннях експертів, зокрема на їх знаннях, щодо системи, що захищається. Для того щоб зменшувати цю залежність, застосовуються різні методи аналізу даних, а також методи машинного навчання.

Одним із методів виявлення вторгнень є застосування системи, заснованої на нечіткій логіці, яка дозволяє ефективно виявляти випадки вторгнення.

Ця кваліфікаційна робота спрямована на розробку системи на основі нечіткої логіки, яка дозволить виявляти вторгнення за рахунок застосування правил із нечітким висновком. Експерименти та оцінки системи виявлення пропонується проводити з урахуванням наборів даних KDD Cup 99.

Метою роботи є виявлення мережевих атак із застосуванням нечіткого виводу.

Завдання:

1. Визначення значних атрибутів;
2. Формування основи нечітких правил;
3. Розробка прототипу програмного комплексу;
4. Тестування прототипу системи.

# 1 ВИЯВЛЕННЯ ВТОРГНЕНЬ

## 1.1 Основні положення функціонування систем виявлення вторгнень

Система виявлення вторгнень (СВВ) - програмний або апаратний засіб, призначений для виявлення фактів несанкціонованого доступу до комп'ютерної системи чи мережі або несанкціонованого управління ними в основному через Інтернет. Відповідний англійський термін - Intrusion Detection System (IDS). Системи виявлення вторгнень забезпечують додатковий рівень захисту комп'ютерних систем. Системи виявлення вторгнень використовуються для виявлення деяких типів шкідливої активності, яка може негативно вплинути на безпечність комп'ютерної системи. До такої активності відносяться мережеві атаки проти вразливих сервісів, атаки, неавторизований доступ до важливих файлів, а також дії шкідливого програмного забезпечення (комп'ютерних вірусів, троянів і черв'яків). В кінці 90-х років Агентство просунутих дослідницьких проектів зробило спробу структурувати складові елементи СВВ.

Свою схему вони назвали Загальною системою виявлення вторгнень (Common Intrusion Detection Framework (cidf)).

За цією схемою архітектура СВВ включає (Рисунок 1.1):

-сенсорну підсистему, призначену для збору подій, пов'язаних з безпекою системи що захищається (E-boxes);

-підсистему аналізу, призначену для виявлення атак і підозрілих дій на основі даних сенсорної системи (A-boxes);

-сховище, що забезпечує накопичення подій і результатів аналізу цих подій (D-boxes);

-засоби протидії, які реєструють виявлені потенційно небезпечні події та сигналізують про це або виконують певні маніпуляції щоб зупинити атаку (C-boxes);

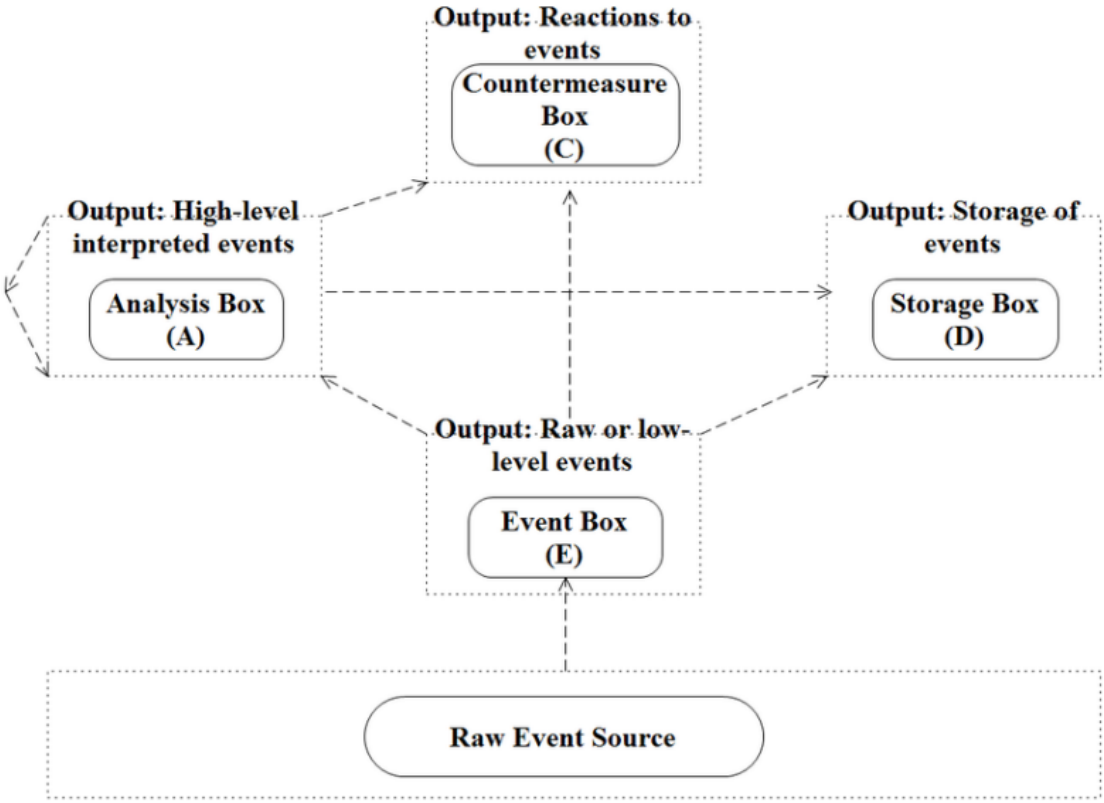



Рисунок 1.1 – Загальна схема роботи системи виявлень вторгнень

## 1.2 Класифікація систем виявлення вторгнень

Системи виявлення вторгнень можуть бути класифіковані різними способами. Ця класифікація може ґрунтуватися на джерелі даних, поведінці системи, архітектурі, на тому, як система захищена і як виявляються проникнення (рисунок 1.2).

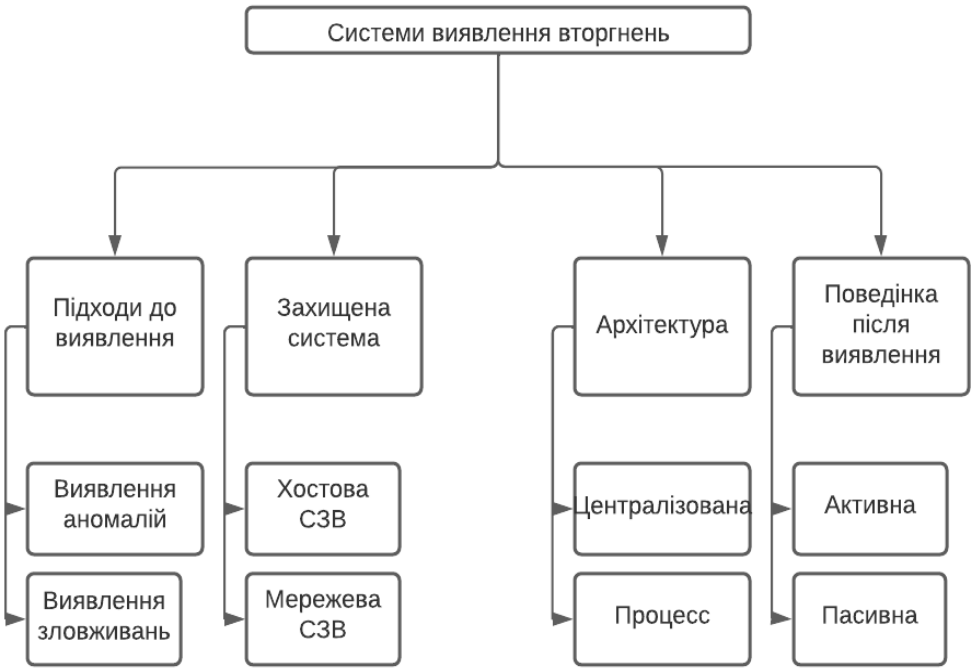



Рисунок 1.2 – Класифікація систем виявлень вторгнень

Відповідно до стратегії виявлення, СВВ, головним чином, можуть бути класифіковані, як системи виявлення аномалій і зловживань. Виявлення вторгнень на основі аномалій також відомо як поведінковий аналіз. Метод базується на виявленні трафіку, який сильно відрізняються від нормальної поведінки системи. Застосування цього підходу дозволяє виявляти раніше невідомі атаки. Другий підходи - це виявлення зловживань, також відоме як сигнальний аналіз. Основою методу є пошук атак заздалегідь складеним шаблонам, таким чином, система може виявити тільки відомі атаки. Кожен з цих методів має свої переваги і недоліки [2].

Тип СВВ, заснований на способі захисту, може бути класифікований відповідно до джерела даних, з якого отримана інформація. Хостова СВВ аналізує дані, отримані від єдиного вузла або комп'ютерної системи. З іншого боку, мережеві системи виявлення вторгнень тримають під наглядом кожен вузол мережі. Однак, системи, доступні на ринку, здебільшого є гібридом IDS.

Система виявлення вторгнення може бути як розподіленою, так і централізованою.

У розподілених системах у мережі є кілька СВВ, пов'язаних один з одним, або підключених до централізованого сервера. Також слід розуміти, що IDS може бути автономною системою. Відповідно до поведінки, СВВ можна розділити на активні та пасивні. Активні системи дозволяють не тільки виявляти, але й негайно реагувати на можливі атаки. З іншого боку, пасивні СВВ лише виявляють проникнення. Отже, активні системи також відомі як системи запобігання вторгнень (СЗВ).

Перевага мережної системи виявлення вторгнень полягає в тому, що вона може захистити велику кількість пристроїв одного мережевого розташування. Для більшості підприємств це найпростіший у розгортанні та найменш дорогий варіант із двох. Навпаки, HIDS необхідно розгорнути та керувати ним для кожного хоста в мережі.

NIDS також швидше реагують на потенційні загрози, ніж HIDS, оскільки вони відстежують заголовки пакетів, що передаються мережею, в режимі реального часу. Це не означає, що HIDS неефективні; вони досягають успіху у виявленні внутрішніх загроз, таких як виявлення змін прав доступу до файлів.

HIDS забезпечить другу лінію захисту, виявляючи атаки, які NIDs могли виявити. Тому використання обох разом було б найбільш надійною стратегією IDS.

Виявлення вторгнення в мережу пропонує ряд варіантів безпеки, але, як і будь-яке інше рішення безпеки, воно має свої недоліки.

Основні переваги NIDS:

- NIDS можна легко розгорнути в існуючій мережі з невеликими порушеннями;
- вони можуть виявляти події в реальному часі, дозволяючи їм реєструвати докази атаки, яку зловмисник міг спробувати видалити;
- система мережевого вторгнення може аналізувати різні типи та кількість атак. Зібрані дані потім можуть бути використані для

забезпечення більш ефективного контролю безпеки та виявлення проблем із конфігурацією мережевого пристрою;

- покращена видимість мережі полегшує виконання конкретних вимог щодо IT-безпеки.

Основні проблеми NIDS:

- Система виявлення мережевих вторгнень допомагає лише виявити атаки, а не запобігти чи зупинити їх. З цієї причини NIDS необхідно поєднувати з іншими заходами безпеки, щоб забезпечити комплексну стратегію кібербезпеки;
- NIDS не може аналізувати зашифровані пакети;
- системи виявлення вторгнення в мережу не можуть легко розпізнати певні типи атак, наприклад, якщо використовуються фрагментовані пакети;
- NIDS потрібен досвідчений системний адміністратор для нагляду та моніторингу, який має необхідне розуміння, щоб вжити заходів щодо будь-якої загрози. Вони також повинні бути під рукою, щоб реагувати на часті помилкові спрацьовування.

Система виявлення вторгнень на базі хоста відстежує та надсилає сповіщення, якщо виявлено підозрілу активність на одному хості, наприклад, на комп’ютері, сервері чи іншому кінцевому пристрої. Більшість HIDS розгортають програмне забезпечення, відоме як агент, на хості, який буде відстежувати діяльність та звітувати про неї. Деякі приклади того, що буде відстежувати HIDS, — це мережевий трафік для цього конкретного хоста, доступ до файлів, модифікації файлів, зміни конфігурації, запущені процеси та події, журнали програм і системи.

HIDS зазвичай встановлюються на критичних хостах, таких як сервери, які містять конфіденційні дані або доступні для громадськості. Але оскільки агенти HIDS можуть бути розгорнуті на будь-якому окремому хосту, якщо потрібно.

Вони доступні для використання на більшості серверів і комп'ютерів, які використовуються бізнесом.

Host-Based Intrusion Detection пропонує широкий спектр можливостей безпеки, але, як і будь-яке інше рішення для забезпечення безпеки, має недоліки.

Основні переваги HIDS:

- Однією з переваг рішень на основі хоста є те, що вони можуть перевіряти дані, які можуть бути зашифровані, коли вони проходять через мережу, що ускладнює перевірку трафіку мережевими рішеннями. Хост-рішення може перевіряти дані в системній пам'яті в тих точках, де вони не зашифровані;
- система виявлення вторгнень на основі хоста також є корисним інструментом виявлення внутрішніх загроз, оскільки вона може виявляти підозрілі запити клієнт-сервер і зміни прав доступу до файлів;
- перевага HIDS у тому, що може відслідковувати певні дії, тому надає набагато більше деталей, ніж мережеві системи. Наприклад, він може відстежувати всі дії користувача під час підключення до мережі. Він буде знати, чи були змінені будь-які облікові записи користувачів, як тільки зміни будуть виконані, і відстежувати, коли користувач увійшов до системи або вийшов із неї. Оскільки HIDS може відслідковувати будь-які зміни в системних файлах, якщо намагаються перезаписати їх або встановити бекдори, це можна ідентифікувати. Цей тип діяльності часто не беруть до уваги NIDS;
- іншою важливою функцією HIDS є моніторинг цілісності файлів (МЦФ). Він може надати контрольний журнал для відстеження того, чи доступ до важливих файлів або їх зміна. МЦФ часто потрібний для відповідності вимогам, таким як Вимога 11.5 PCI DSS (Стандарт безпеки даних індустрії платіжних карток), якщо ви обробляєте кредитні картки;

– HIDS також має нижчий поріг входу, ніж NIDS, що робить їх рентабельнішими для малого бізнесу. Крім того, додаткове обладнання не потрібне, що дозволяє заощадити на управлінні та обслуговуванні.

Основні переваги HIDS:

– HIDS складніші в управлінні, ніж установка NIDS аналогічного розміру, особливо для більшої організації, де необхідно встановити, налаштувати та керувати кожним хостом, що потребує моніторингу. Це може бути складним процесом для великої мережі з тисячами хостів. Логічним рішенням у цьому випадку було б використати їхню комбінацію. Мережева система для більшої частини мережі та хост-система виявлення вторгнень для критично важливих машин;
– шкідливе ПЗ, що встановилося на хості, може отримати доступ до привілеїв і підвищити їх. Це може дозволити йому, наприклад, відключити антивірусне програмне забезпечення та ведення журналу під час атаки або навіть відключити HIDS після зламування;
– для адміністрування HIDS потрібен досвідчений адміністратор.

## 1.3 Методи виявлень вторгнень

Виявлення вторгнень є значною областю досліджень, оскільки неможливо створити системи без вразливостей. Однією з основних складнощів у виявленні вторгнень є необхідність виділення прихованих атак із величезного обсягу трафіку. Існують такі алгоритми машинного навчання, наприклад, нейронні мережі [3], метод опорних векторів [4], генетичні алгоритми [5], нечітка логіка [6], та методи аналізу даних (Data Mining) [8] , які використовуються для виявлення аномальних активності у великому обсязі складних та динамічних наборів даних. Генерація правил – дуже важливий аспект IDS. У системі необхідно відрізняти стандартну поведінку від нестандартної. Для виявлення раніше зустрічалися вторгнень проведено безліч різних досліджень. Метод

аналізу даних, що дозволяє визначати взаємозв'язки між подіями та виявляти приховані шаблони даних, які інакше пройшли б непоміченими, відомий як Data Mining. Відповідно до стратегії виявлення, системи Data Mining можна поділити на дві основні категорії [8]. Це виявлення зловживань, де для ідентифікації проникнень, використовуються шаблони відомих вторгнень або вразливостей системи, а також виявлення аномалій, що дозволяє виявити відхилення від стандартної поведінки.

Нейронна мережа — це набір алгоритмів, які намагаються визначити основні зв’язки в наборі даних за допомогою механізму, який імітує роботу мозку людини. Нейронні мережі можуть реагувати на вхідні дані, що розвиваються, тому мережа виробляє найкращий можливий вихід без переробки критеріїв продуктивності. Характеристики нейронної мережі – це комп’ютерна система із взаємопов’язаними вузлами, які функціонують як нейрони в мозку людини. Вони використовують нейронні мережі для виявлення подібності та прихованих тенденцій в необроблених даних, а також для групування та ідентифікації необроблених даних, а також для навчання та постійного вдосконалення з часом.

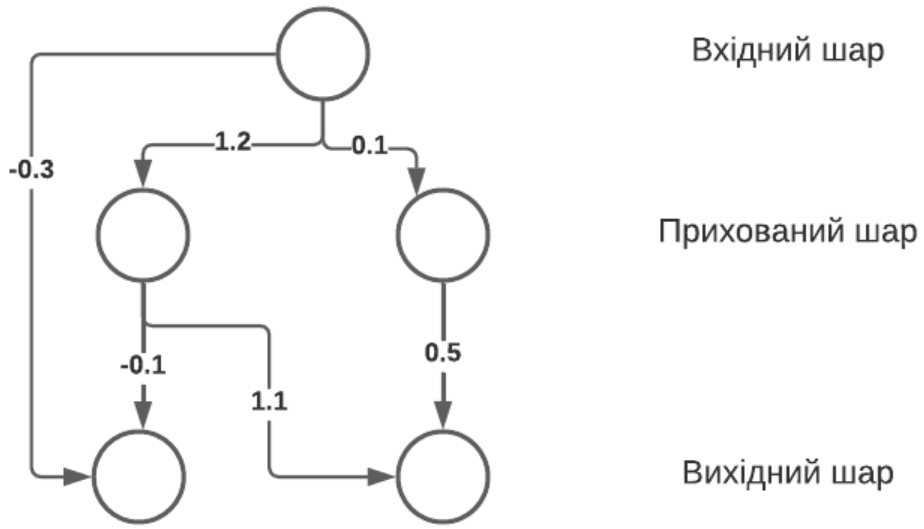



Рисунок 1.3 – Нейронна мережа

Дана мережа складається з п'яти вузлів (нейронів) і шести зв'язків

(з'єднань). Число поруч із кожним із зв'язків називається вагою – він визначає силу зв'язку або важливість вхідного сигналу. нелінійний елемент. При проходженні через нейрон всі вхідні сигнали множаться на відповідні їм вагові коефіцієнти, далі отримане значення подається на вхід функції активації нейрона, яка в свою чергу визначає вихідний сигнал. Ця властивість полягає у можливості зміни синаптичних зв'язків між нейронами, в разі машинного навчання - це коригування значень вагових коефіцієнтів на вхідних сигналах. Недоліком даного методу при виявленні вторгнень є сильна залежність результату роботи алгоритму від навчальної множини.

Переваги нейронних мереж:

- Нейронні мережі мають здатність вчитися самостійно та генерувати вихідні дані, які не обмежуються вхідними даними, які вони надають;
- вхідні дані зберігаються у власних мережах замість бази даних. Отже, втрата даних не впливає на спосіб їх роботи;
- нейронна мережа буде вчитися на екземплярах і адаптувати їх, коли відбувається подібна подія, що дозволить їм функціонувати через подію в режимі реального часу;
- навіть якщо нейрон не відповідає або інформація втрачена, мережа все одно здатна виявити несправність і створити вихід;
- нейронні мережі виконують кілька завдань паралельно, не впливаючи на продуктивність системи;
- зберігання інформації по всій мережі;
- вміння працювати з неповними знаннями;
- наявність відмово-стійкості;
- наявність розподіленої пам’яті;
- можливість робити машинне навчання;
- можливість паралельної обробки.

Недоліки нейронної мережі:

- Основними недоліками нейронних мереж є їхня «чорна скринька»;

– іноді вам потрібен більший контроль над деталями алгоритму, хоча є бібліотеки, такі як Keras, які роблять розробку нейронних мереж досить простою;
– нейронні мережі зазвичай вимагають набагато більше даних, ніж традиційні алгоритми, як принаймні тисячі, якщо не мільйони позначених зразків;
– нейронні мережі також складніші з точки зору обчислень, ніж традиційні алгоритми;
– тривалість нейронної мережі невідома;
– апаратна залежність;
– незрозуміла поведінка мережі;
– визначення належної структури мережі;
– складність показу проблеми мережі;

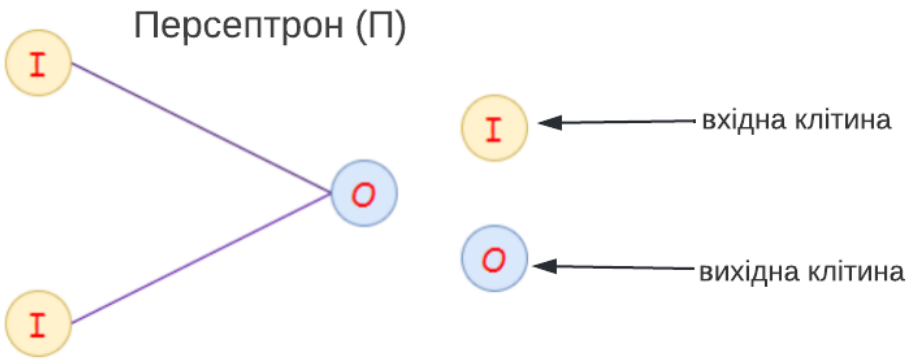



Рисунок 1.4 - Представлення персептрона

Модель Perceptron, запропонована Мінським-Папертом, є однією з найпростіших і найстаріших моделей Neuron. Це найменша одиниця нейронної мережі, яка виконує певні обчислення для виявлення функцій або бізнес-аналітики у вхідних даних. Він приймає зважені вхідні дані та застосовує функцію активації, щоб отримати вихід як кінцевий результат. Персептрон

також відомий як TLU (порогова логічна одиниця)

Perceptron — це контрольований алгоритм навчання, який класифікує дані на дві категорії, таким чином, це двійковий класифікатор.

Переваги персептрона

– Персептрони можуть реалізовувати логічні елементи, такі як І, АБО або NAND.

Недоліки Персептрона

– Персептрони можуть вивчати лише лінійно окремі задачі, такі як логічна задача І. Для нелінійних задач, таких як логічна задача XOR, це не працює.

Програми:

– Класифікація;
– кодування бази даних (багатошаровий персептрон);
– монітор даних доступу (багатошаровий персептрон).

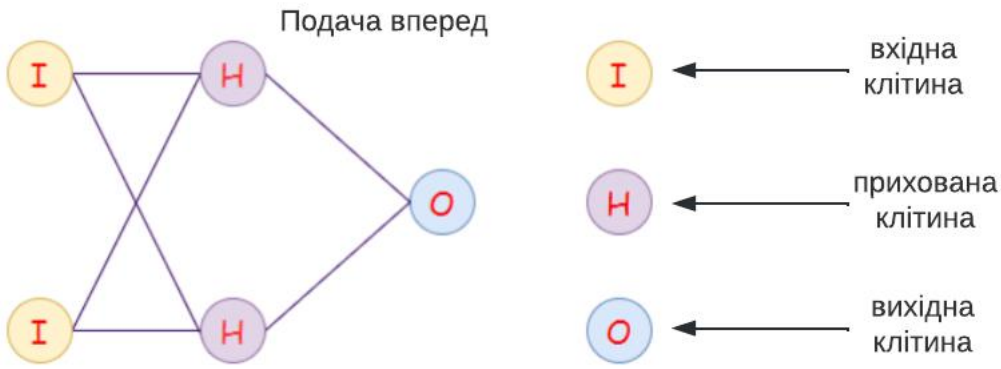



Рисунок 1.5 - Нейронна мережа із прямим зв'язком.

Розглянемо радіальну базову мережу (RBN). Мережі радіальних базисних функцій зазвичай використовуються для задач апроксимації функцій. Їх можна відрізнити від інших нейронних мереж завдяки швидшій швидкості навчання та універсальному наближенню. Основна відмінність між радіальними базовими

мережами та мережами з прямим зв'язком полягає в тому, що RBN використовують функцію радіального базису як функцію активації. Логістична функція (сигмовидна функція) дає вихід від 0 до 1, щоб визначити, чи є відповідь так чи ні. Проблема в тому, що якщо у нас є безперервні значення, то RBN не можна використовувати. RBIs визначає, наскільки далекий наш генерований результат від цільового результату. Вони можуть бути дуже корисними у випадку безперервних значень. Підсумовуючи, RBI поводяться як мережі FF, використовуючи різні функції активації.

Програми:

– Стиснення даних;
– розпізнавання образів;
– комп'ютерний зір;
– розпізнавання цілі сонара;
– розпізнавання мови;
– розпізнавання рукописних символів;

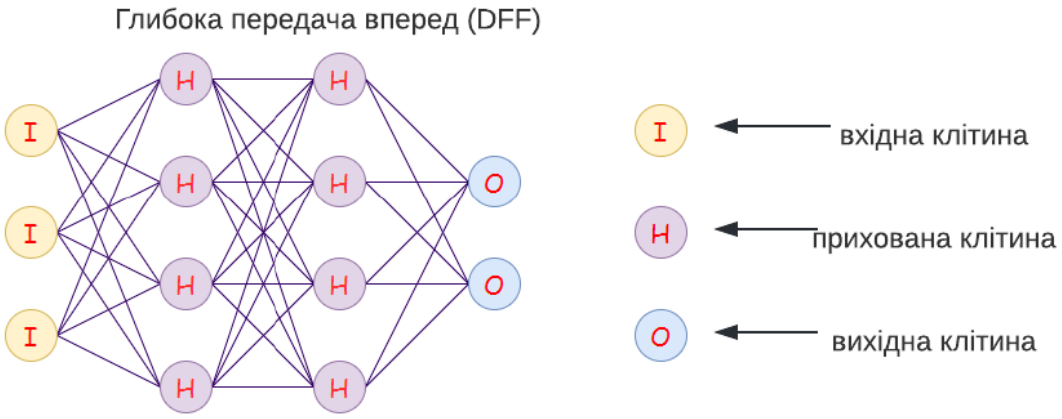



Рисунок 1.6 - Представлення глибокої нейронної мережі з прямим зв'язком.

Глибока мережа прямої подачі – це мережа з прямим зв’язком, яка використовує більше ніж один прихований шар. Основною проблемою

використання лише одного прихованого шару є переобладнання, тому, додаючи більше прихованих шарів, ми можемо досягти (не у всіх випадках) зменшення переобладнання та покращення узагальнення.

Програми:

– Розпізнавання мови;

– розпізнавання письма.

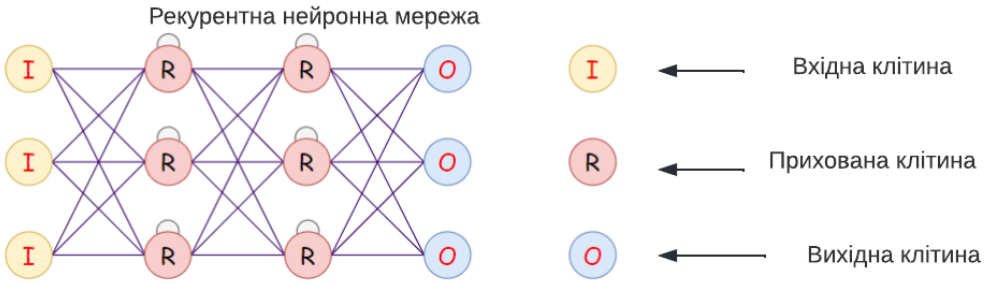



Рисунок 1.7 - Представлення рекурентної нейронної мережі (RNN)

Рекурентні нейронні мережі (RNN) є різновидом мереж прямої подачі (FF). У цьому типі кожен із нейронів у прихованих шарах отримує вхід із певною затримкою в часі. Ми використовуємо цей тип нейронної мережі, де нам потрібно отримати доступ до попередньої інформації в поточних ітераціях. RNN можуть обробляти вхідні дані та розподіляти будь-які довжини та ваги в часі. Розмір моделі не збільшується з розміром вхідних даних, і обчислення в цій моделі враховують історичну інформацію. Однак проблема цієї нейронної мережі полягає в повільній швидкості обчислень. Більше того, він не може враховувати будь-які майбутні внески для поточного стану. Він не може пам'ятати відомостей з давнього часу.

Програми:

– Машинний переклад;

– управління роботом;

– прогнозування часових рядів;
– розпізнавання мови;
– синтез мовлення;
– виявлення аномалій часових рядів;
– вивчення ритму;
– музична композиція.

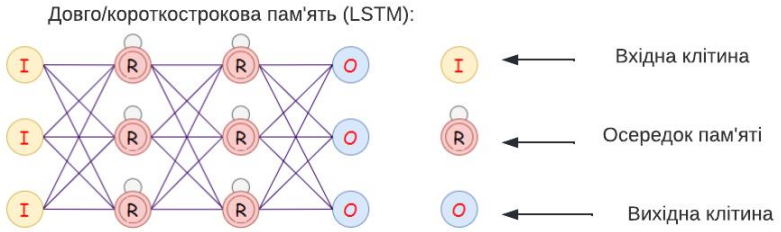



Рисунок 1.8 - Представлення мережі довгострокової пам'яті (LSTM)

Мережі LSTM представляють комірку пам'яті. Вони можуть обробляти дані з пробілами в пам'яті. Вище ми можемо помітити, що ми можемо враховувати часову затримку в RNN, але якщо наша RNN виходить з ладу, коли у нас є велика кількість релевантних даних, і ми хочемо з'ясувати відповідні дані з неї, тоді LSTM – це шлях. Крім того, на відміну від LSTM, RNN не можуть запам'ятати дані давнього часу.

Програми:

– Розпізнавання мови;
– розпізнавання письма.

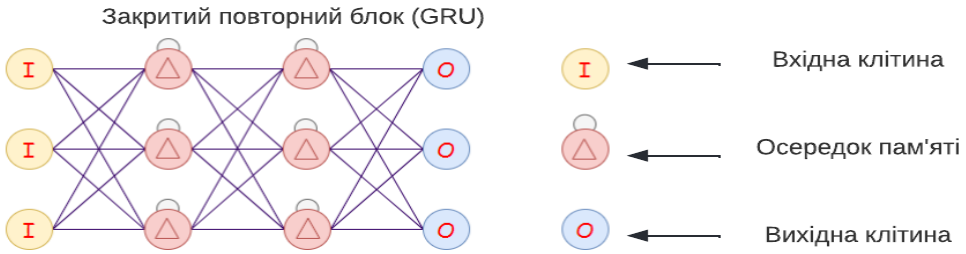



Рисунок 1.9 - Представлення мережі з рекурентним блокуванням (GRU)

Закриті повторювані блоки є різновидом LSTM, оскільки вони обидва мають подібну конструкцію і в основному дають однаково хороші результати. GRU мають лише три ворота, і вони не підтримують внутрішній стан клітини.

a. Update Gate: визначає, скільки минулих знань передати в майбутнє.

б. Reset Gate: визначає, скільки минулих знань потрібно забути.

c. Current Memory Gate: частина скидання долі.

Програми:

– Поліфонічне музичне моделювання;
– моделювання мовного сигналу;
– обробка природної мови.

Метод опорних векторів одна із способів розв'язання завдання класифікації. Підхід був запропонований В.Вапником для визначення того, до якого з двох заздалегідь визначених класів повинен належати зразок, що аналізується, заснований на принципі структурної мінімізації ризику. Імовірність помилки при класифікації оцінюється, як безперервна спадна функція, від відстані між вектором і площиною, що розділяє. Вона дорівнює 0,5 в нулі і прагне 0 на нескінченності. Пізніше Вапник та інших. поширив цей підхід до класифікації кілька класів [3]. На рисунку 3 наведено приклад для двовимірного випадку.

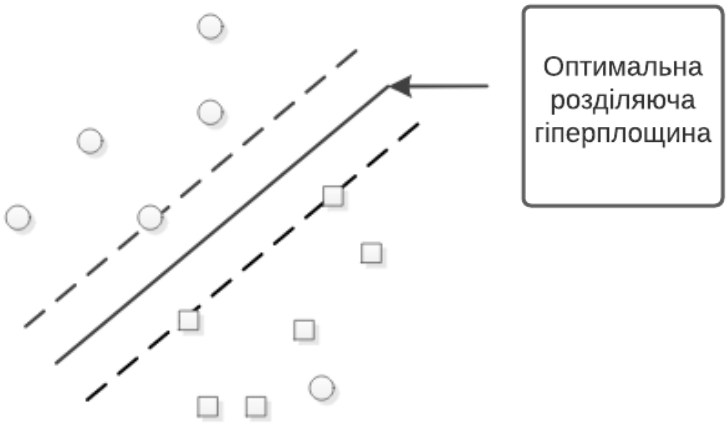



Рисунок 1.10 – Метод опорних векторів (випадок лінійної роздільності)

Переваги опорних векторів

- SVN дуже хороші, коли ми не маємо уявлення про дані;
- добре працює навіть з неструктурованими та напівструктурованими даними, такими як текст, зображення та дерева;
- трюк ядра – це справжня сила SVN. За допомогою відповідної функції ядра ми можемо вирішити будь-яку складну проблему;
- на відміну від нейронних мереж, SVN не вирішується для локальних оптимумів;
- він відносно добре масштабується до даних великої розмірності;
- моделі SVN мають узагальнення на практиці, ризик переобладнання менший у SVN;
- SVN завжди порівнюють з ANN. У порівнянні з моделями ANN, SVN дають кращі результати.

Недоліки опорних векторів

- якщо кількість функцій набагато більше, ніж кількість зразків, уникайте надмірного підбору функцій ядра, і термін регуляризації має вирішальне значення;

– SVN не надають безпосередньо оцінки ймовірності, вони розраховуються за допомогою дорогої п’ятикратної перехресної перевірки.

Генетичні алгоритми (ГА) були запропоновані у 1960-х рр. Дж. Холландом. Схема роботи алгоритму представлена на рисунку 1.11.

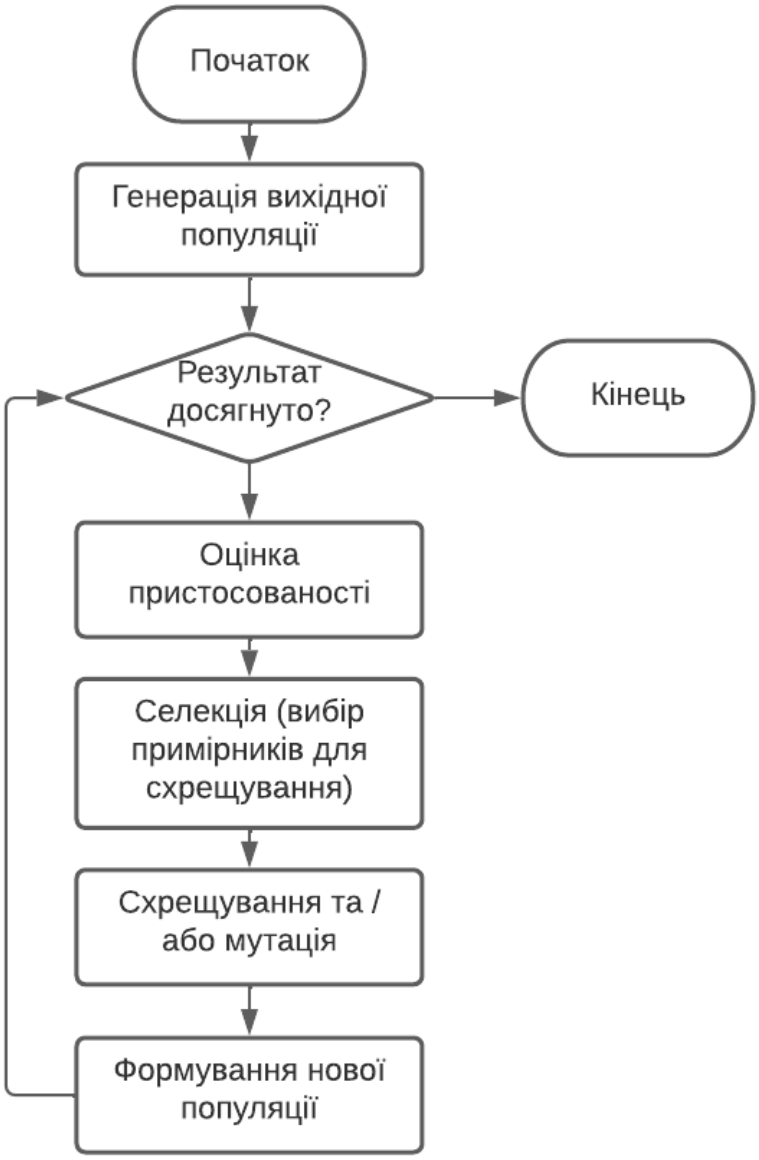



Рисунок 1.11 – Блок-схема роботи генетичного алгоритму

Генетичні алгоритми (ГА) — це адаптивні евристичні алгоритми пошуку, які належать до більшої частини еволюційних алгоритмів. Генетичні алгоритми засновані на ідеях природного відбору та генетики. Це інтелектуальне використання випадкового пошуку, наданого історичними даними, щоб спрямувати пошук у область кращої продуктивності в просторі рішень. Вони зазвичай використовуються для створення високоякісних рішень для задач оптимізації та пошуку.

Генетичні алгоритми імітують процес природного відбору, що означає, що ті види, які можуть адаптуватися до змін у своєму середовищі, здатні вижити, розмножуватися та перейти до наступного покоління. Простіше кажучи, вони моделюють «виживання найсильнішого» серед індивідів наступного покоління для вирішення проблеми. Кожне покоління складається з популяції індивідів, і кожна людина являє собою точку в просторі пошуку та можливе рішення. Кожна особа представлена у вигляді рядка символів/цілих чисел/бітів. Цей рядок аналогічний хромосомі.

Об'єктом у генетичному алгоритмі є хромосома, представлена як послідовності 0 і 1, тобто. двійкових чисел. Основні механізми ГА - це селекція (відбір) і репродукція (відтворення нової популяції). Функція пристосованості визначає міру пристосованості хромосомів популяції. З цієї функції проводиться відбір найбільш пристосованих особин. Постановка завдання зводиться до коректного завдання функції пристосованості.

Переваги генетичного алгоритму:

- концепцію легко зрозуміти;
- пошук ГА із сукупності точок, а не окремої точки;
- ГА використовує інформацію про виплату (об’єктивна функція), а не о похідних;
- ГА підтримує багатоцільову оптимізацію;
- ГА використовує імовірнісні правила переходу, а не детерміновані правила;
- ГА добре підходить для «шумних» середовищ;

– ГА є надійним щодо до локальних мінімумів/максимумів;
– ГА легко розпаралелити;
– ГА може оперувати різними уявленнями;
– ГА є стохастичним;
– ГА добре працює над змішаною дискретною/безперервною проблемою.

Недоліки генетичного алгоритму:

– Реалізація ГА залишається мистецтвом.
– ГА вимагає менше інформації про проблему, але розробка цільової функції та правильне представлення й оператори можуть бути складними.
– ГА є витратним у обчислювальному відношенні, тобто займає багато часу.

Засновником наукової теорії про нечітку логіку можна по праву вважати американського математика Лофті Заде, який у 1960-х роках. опублікував свої наукові труди. Для визначення нечіткої множини вводяться поняття нечіткої та лінгвістичної змінної.

Нечітка логіка використовується як техніка для прийняття людських рішень під час використання платформи машинного навчання або штучного інтелекту. У загальному вигляді це можна описати як врахування істинних значень змінних між 0 і 1. Представлення дійсних чисел між двійковими числами 0 і 1 реалізується за допомогою нечіткої логіки.

Вхідні змінні відображаються наборами функцій належності, відомими як нечіткі множини. Система керування може мати значення таблиці істинності, рівне 0 або 1. Контролер приймає вхідні дані системи, які потім надаються, і відображає їх у свої функції належності та значення істинності.

Якщо діапазон похибки становить від -1 до +1, а аналого-цифровий перетворювач має роздільну здатність 0,25, то нечіткий набір вхідних змінних описується дуже просто у вигляді таблиці.

Нечітка змінна описується:

$$<\alpha, X, A>$$

де $\alpha$ – назва нечіткої змінної; $X$ – область визначення нечіткої змінної; $A$ – нечітка множина на множині $X$.

$$<\beta, T, X, G, M>$$

$\beta$ де - назва лінгвістичної змінної; $T$ - безліч її значень (терм-множина), елементи якого являють собою назви нечітких змінних; $X$ - область визначення нечітких змінних; $G$ – синтаксичне правило генерації нових термів; $M$ – семантичне правило, що ставить у відповідність до значення лінгвістичної змінної, утвореної процедурою $G$, нечітке підмножина множини $X$.

Нечітким висловлюванням називають висловлювання наступного виду:

$$<\beta \epsilon \alpha >$$

де $\beta$ -ім'я лінгвістичної змінної; $\alpha$ - її значення, тобто. один з термів цієї змінної, якому відповідає нечітка підмножина множини $X$.

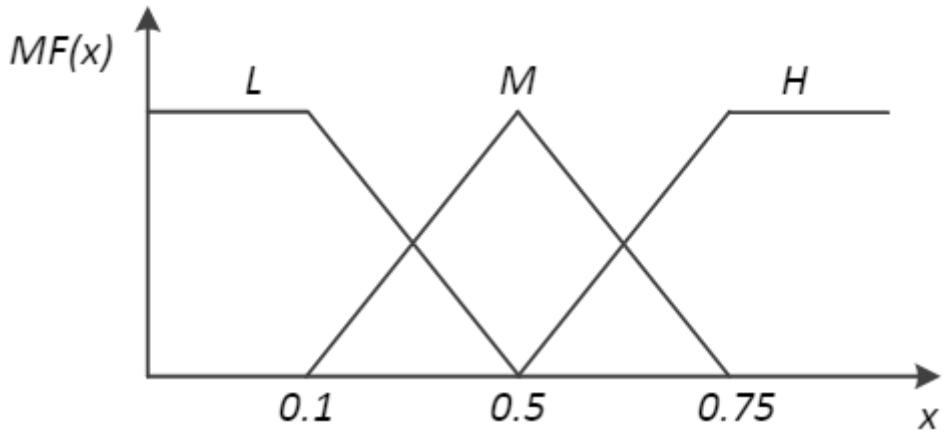



Рисунок 1.12 – Розбиття інтервалу за допомогою нечітких змінних з використанням трикутних функцій приладдя

Переваги нечіткої логіки

- Ця система гнучка і може змінюватися.
- системи Fuzzy_logic легко побудувати;
- ці системи надають рішення для комплексних рішень;
- логіка надійна, проста і може бути змінена відповідно до наших вимог;
- ця логіка може одночасно обробляти різні типи вхідних даних і приймати точні рішення, використовуючи точні функції;
- ці логічні системи мають просту структуру і їх легко побудувати;
- цей тип логіки вимагає менше місця і може бути закодований, використовуючи менше даних;
- рішення можна приймати легко, оскільки його логічна система нагадує людські міркування, що дозволяє легко вирішувати більш складні проблеми;
- якщо система зворотного зв'язку виходить з ладу в логічній системі, її можна легко перепрограмувати.

Недоліки Fuzzy Logic

- Потрібні регулярні оновлення системи керування нечіткою логікою;
- ці системи не зможуть розпізнавати машинне навчання та платформи нейронних мереж;
- основний недолік нечіткої логіки полягає в тому, що вона повністю залежить від людського інтелекту та досвіду;
- вони не використовуються широко через отримання неточних даних;
- ефективність системи невисока, оскільки вони в основному працюють на неточних входах;
- контролер нечіткої логіки повністю залежить від людських знань та досвіду. Ці контролери не можуть розпізнати машинне навчання або нейронні мережі.

# 2 НЕЧІТКІ ПРОДУКЦІЙНІ МОДЕЛІ

Нечіткі продукційні моделі (Rule-BasedFuzzyModels) застосовуються для опису, моделювання та аналізу складних слабоформалізованих систем та процесів. Під нечіткою продукційною моделлю розуміють безліч нечітких продукційних правил виду:

***ЯКЩО А, ТО В***

де ***А*** – передумова у вигляді нечіткого висловлювання; ***В*** – висновок у вигляді нечіткого висловлювання.

Компоненти, необхідні для побудови нечіткої продукційної моделі:

- схема нечіткого виведення висновків;
- база нечітких продукційних правил;
- процедура введення нечіткості;
- процедура агрегування ступеня істинності передумов у кожному з нечітких продукційних правил;
- процедура активізації висновків кожного з нечітких продукційних правил;
- процедура акумулювання активізованих висновків для кожної вихідної змінної;
- процедура дефазифікації (приведення до чіткості).

## 2.1 Способи нечіткого виведення

У нечітких продукційних системах зазвичай використовуються два способи нечіткого виведення висновків: прямий та зворотний. Прямий спосіб нечіткого виведення виходить з правила (fuzzymodusponens). Схема включає в себе такі етапи:

1.Завдання нечіткої імплікації R : A → B, яка визначає нечітке причинно-

наслідкове відношення між передумовою та висновком, яке подається у вигляді нечіткої продукції:

***ЯКЩО x є A ,ТО y є B***

де ***x*** –вхідна змінна, ***x ∈X***, ***X*** – область визначення передумови; ***A***–нечітка множина, визначена на ***X***; ***y*** –вихідна змінна, **y ∈ Y**, де ***Y*** – область визначення укладання; ***B*** – нечітка множина, визначена на ***Y***, з функцією приналежності ***M*** $F_B(y) \in [0,1]$. 2. Завдання нечіткого факту:

**<*x′ є A′*>**

де **x′**–реальне (фактичне) значення змінної ***x***; **A′**–нечітка множина, що відображає значення **x′**, визначене на множині ***X***, з функцією приналежності $MF_B'(x) \in [0,1]$.

3. Формування висновку:

**<*y′ є B′*>**

де ***y′***-отримане значення змінної ***y***; ***B′***–нечітка множина, що відображає значення y′, визначене на множині Y, з функцією приналежності $MF_B'(y) \in [0,1]$.

Зворотний спосіб нечіткого виведення виходить з правила нечіткий модус толленс (fuzzymodustollens). Схема включає в себе наступні етапи:

1. Завдання нечіткої імплікації ***R:A→B***, яка визначає нечітке пичинно-слідче відношення між передумовою і висновком, яке представляється у вигляді нечіткої продукції:

***ЯКЩО x є A ,ТО y є B***

Мета нечіткого зворотного висновку - це встановлення істинності причини.

2.2 Формування бази правил

База правил у системах нечіткого виведення призначена для представлення емпіричних знань або знань експертів у тій чи іншій предметній області. У системах нечіткого виведення використовуються нечіткі продукційні правила, у яких умови та висновки сформульовані у термінах нечітких лінгвістичних висловлювань. При визначенні простих нечітких висловлювань необхідно задати функції належності відповідних нечітких множин.

Зазвичай база правил представляється в такому вигляді:

$$R_1: \text{ЯКЩО "Умова}_1\text{", ТО "Висновок}_1\text{"} (F_n)$$
$$R_2: \text{ЯКЩО "Умова}_2\text{", ТО "Висновок}_2\text{"} (F_n)$$
$$R_n: \text{ЯКЩО "Умова}_n\text{", ТО "Висновок}_n\text{"} (F_n)$$

де, $F_i(i \in \{1,\ldots,n\})$ - вагові коефіцієнти, які позначає ступінь істинності і-ого правила. Якщо вагові коефіцієнти відсутні, то за замовчуванням їх значення набирають рівними одиниці. Залежно від кількості нечітких висловлювань у передумовах та висновках база правил може бути представлена однією з наступних структур:

«Один вхід - один вихід» (SISO-Single Input-Single Output);

«Багато Входів – один вихід» (MISO – Multi Inputs – Single Output);

«Багато Входів – багато виходів» (MIMO – Multi Inputs – Multi Output);

При складанні бази нечітких правил необхідно оцінити і забезпечити їхню повноту (достатність), їх несуперечність, а також можливості, усунути кореляції між окремими нечіткими правилами в базі.

2.3 Фазифікація вхідних змінних (введення нечіткості)

Під фазифікацією розуміється процес знаходження значень функцій належності нечітких множин вхідних змінних для всіх передумов нечітких продукційних правил. Фазифікацію часто називають запровадженням нечіткості.

Метою даного етапу є встановлення відповідності між конкретним, зазвичай чисельним, значенням вхідної змінної та значенням функції належності їй терму вхідної лінгвістичної змінної.

Отримане цьому етапі чисельне значення функції власності є результатом нечіткого висновку і характеризує рівень істинності причини у правилі.

## 2.4 Агрегування ступеня істинності передумов

З огляду на те, що у основі правил може бути правила складового виду, даний етап служить визначення ступеня істинності кожного правила. Наприклад, якщо у правилі є кілька передумов $\alpha_i$ $(i = 1, ..., n)$, то на етапі фазифікації будуть отримані значення функцій приналежності по кожній з них:

$$MF_{Ai1}(x'_1),..., MF_{Aij}(x'_j),..., MF_{Aim}(x'_m)$$

Далі ці значення агрегуються. Зазвичай використовується min-кон'юнкція ступенів істинності правил:

$$MF_{Aij \wedge Aim}(x)=min\{MF_{Aij}(x'_j),MF_{Aim}(x'_m)$$

## 2.5 Активація під заключень

На етапі активації перебуває ступінь істинності кожного з під заключень нечітких продукційних правил. Вважається, що на початок цього етапу вже відомі ступеня істинності всіх умов з бази нечітких правил та значення вагових коефіцієнтів $F_i$ для кожного правила. Далі визначається ступінь істинності кожного висновку, яка дорівнює твору алгебри відповідного значення ступеня істинності умови на ваговий коефіцієнт. Активація багато в чому дуже схожа на композицію, але не тотожна їй, у системах нечіткої логіки використовуються

лінгвістичні змінні. Як таку операцію зазвичай виступає так звана min-активація:

$$MF_{Bi}'=min\{\alpha_i,MF_{Bi}(y)\}$$

Далі результати коригуються рахунок їх алгебраїчного твори на вагові коефіцієнти. Якщо ж вагові коефіцієнти не задані, то вважаємо їх рівними 1. За наявності кількох під заключень у нечіткому продукційному правилі, ваговий коефіцієнт може бути заданий не тільки для правила загалом, але й індивідуально для кожного під заключення. Корекція бази нечітких правил рахунок множення їх у вагові коефіцієнти є альтернативою методу параметричної оптимізації з допомогою підстроювання параметрів функцій власності.

## 2.6 Акумулювання висновків

Акумулювання (акумуляція) - це процес знаходження функції приналежності для кожної з вихідних лінгвістичних змінних.

Метою акумуляції є об'єднання всіх ступенів істинності під заключень, щоб отримати ступінь істинності кожної з вихідних змінних.

У процесі нечіткого виведення цей етап необхідний, під складання, які стосуються однієї й тієї ж лінгвістичної змінної належать різним правилам системи нечіткого логічного висновку.

Об'єднанням двох нечітких множин є третя нечітка множина, з функцією приналежності, що обчислюється за формулою:

$$MF_i(x)=max\{MF_1(x),MF_2(x)\}$$

де, $MF_1(x)$ $MF_2(x)$ -функції приналежності функцій, що об'єднуються.

## 2.7 Дефаззифікація (приведення до чіткості)

У системах нечіткого виведення дефаззифікація є процес знаходження кількісного значення кожної вихідний лінгвістичної змінної(перетворення нечіткого значення на чітке).

Метою дефазифікації є отримання числових значень (crispvalue) на основі результатів акумуляції всіх лінгвістичних вихідних змінних.

Це перетворення необхідно для подальшої обробки даних зовнішніми пристроями. Передбачається, що до початку цього етапу вже відомі функції належності всіх вихідних лінгвістичних змінних у формі нечітких множин:

$$C_1', C_2', \dots C_s',$$

де s–число вихідних лінгвістичних змінних у основі нечітких правил. Після цього послідовно розглядається кожна з вихідних лінгвістичних змінних і що належить до неї безліч $C_j'$. Результатом дефазифікації є кількісне значення $y_i' \in \mathbf{R}$, отримане одним із способів, описаних нижче.

Деякі методи дефазифікації:

1.Метод центру тяжкості (COGS, Centre of Gravity)

Чітке значення вихідний змінної розраховується як центр тяжкості функції приналежності $MF_B'(y)$ і обчислюється за такою формулою:

$$y' = \frac{\int_{Min}^{Max} y * MF_B'(y)\,dy}{\int_{Min}^{Max} MF_B'(y)\,dy}$$

де $y'$-результат дефазифікації; y–змінна, відповідна вихідної лінгвістичної змінної; $MF(y)$–функція належності нечіткої множини, що відповідає вихідній змінній після етапу акумуляції. $Min$, $Max$–границя інтервалу носія нечіткої множини вихідної змінної y.

При дефазифікації методом центру тяжкості кількісне (чітке) значення

вихідний змінної приблизно дорівнює абсцисі центру тяжіння, обмеженою графіком кривої функції належності відповідної вихідної змінної.

2.Метод центра площі (CoA, COA, Centre of Area)

Чітке значення вихідний змінної y′=u, де значення u визначається з формули:

$$\int_{Min}^{u} MF_{B}'(y)dy=\int_{u}^{Max} MF_{B}'(y)dy.$$

Іншими словами, центр площі дорівнює абсцисі, яка ділить площу, обмежену графіком кривої функції належної вихідної змінної на дві рівні частини.

3.Максимум функції власності. Чітке значення вихідний змінної y′ розраховується за формулою:

$$y'=argsupMF_{Bi}(y)$$

4.Перший максимум (first-of-maxima) Також даний метод називають лівим максимумом (leftmostmaximum). Чітке значення вихідний змінної y′ знаходиться як найменше значення, при якому досягається максимум підсумкової нечіткої множини:

$$y'=min\{y_{max}|\ MF_{Bi}(y_{max})=maxMF_{Bi}(y)\}$$

5. Найправіший максимум (rightmostmaximum)

Чітке значення вихідний змінної y′ знаходиться як найбільше значення, при якому досягається максимум підсумкової нечіткої множини:

$$y'=max\{y_{max}|\ MF_{Bi}(y_{max})=maxMF_{Bi}(y)\}$$

# 3 СИСТЕМА ВИЯВЛЕННЯ ВТОРГНЕНЬ З ВИКОРИСТАННЯМ НЕЧІТКОЇ ЛОГІКИ

Запропонована система виявлення вторгнень використовує нечітку логіку. Вхідними даними є безліч KDD Cup 99, яка розділена на два підмножини: навчальний набір даних і тестовий набір даних. Спочатку, навчальний набір даних класифікується в п'ять підмножин так, щоб виділити чотири типи атак (DoS (Відмова в обслуговуванні), R2L (Видалений до Локального), U2R (Користувач, з правами адміністратора), Probe), та нормальні дані. Далі з використанням програмного забезпечення Weka виявляються найважливіші атрибути, які згодом використовуються для складання певних та невизначених правил. Потім, відповідно до певного правила фазифікації, генеруються нечіткі правила, таким чином, щоб отримати ряд продукційних правил виду «ЯКЩО ... ТО», які дозволять визначити, до якого класу відноситься запис-нормальний дані або ж це аномалія. Для ефективного навчання нечіткої системи, сформовані правила перераховані на основі нечітких продукційних правил. У фазі тестування тестові дані зіставляються з нечіткими правилами з бази для того, щоб визначити, чи є дані нормальними або аномальними.

Побудова систем з нечітким висновком передбачає виконання наступних етапів:

1. Визначення входів і виходів створюваної системи;

2. Завдання для кожної з вхідних і вихідних змінних функції власності;

3. Розробка бази нечітких правил;

4.Вибір та реалізація алгоритму нечіткого логічного висновку;

5.Аналіз процесу управління створеної системи.

На рисунку 3.1 представлені основні етапи роботи системи виявлення вторгнень з чітким висновком.

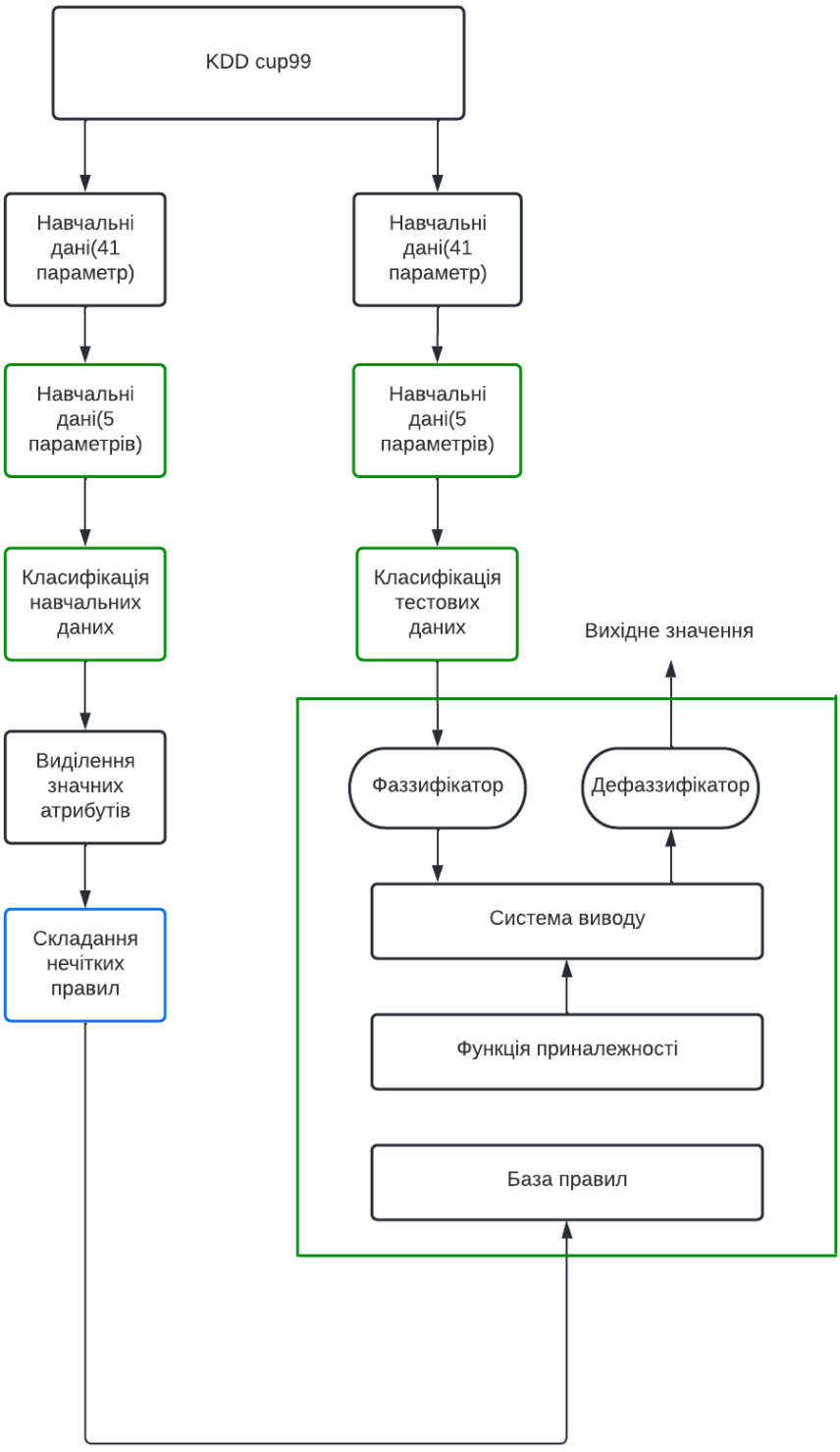



Рисунок 3.1 – Основні етапи роботи системи виявлення вторгнень

3.1 Алгоритм Мамдані у системах нечіткого виведення

На вхід алгоритму подаються кількісні значення, на виході формуються числове значення, на проміжних етапах застосовується апарат нечіткої логіки. Це і є основна перевага нечітких систем - можливість роботи зі звичними числовими даними і при цьому використовувати можливості, що надаються системами нечіткого висновку. На рисунку 3.2 представлені послідовно виконувані етапи алгоритму.


Рисунок 3.2 - Основні етапи нечіткого висновку

## 3.2 Опис бази KDD CUP 99 DATASET

У 1998 році DARPA спільно з Лабораторією Лінкольна Массачусетського технологічного інституту запустили програму DARPA 1998 по набору даних для оцінки IDS. Багато DARPA 1998 містить дані для навчання, отримані за сім тижнів, а також двотижневі дані для тестування. Загалом існує 38 типів атак у навчальних та тестових даних. Удосконаленою версією DARPA набору даних, який містить лише мережні дані (тобто дані Tcpdump), є набір даних, відомий як KDD [10]. Навчальний набір даних KDD складається приблизно з 4900000 одиничних векторів з'єднань, де кожен одиничний вектор з'єднання складається з 41 параметра і позначений як нормальний, або, як атака, з визначенням її типу. Ці параметри мають різні значення, як безперервні, таки символьні з широким діапазоном допустимих значень, що входять до чотирьох категорій:

- Перша категорія складається з властивих/характерних параметрів (intrinsic features), які включають основні риси кожного окремого TCP-з'єднання. Деякі з таких параметрів: тривалість з'єднання, тип протоколу (TCP, UDP і т.п.) та мережна служба (HTTP, Telnet, тощо). навантаження TCP-пакетів, таких як кількість невдалих спроб входу до системи.
- Параметри з'єднання, отримані на основі знання предметної області, використовуються для оцінки корисного навантаження TCP-пакетів, таких як кількість невдалих спроб входу до системи.
- Параметри хосту дозволяють контролювати встановлені з'єднання, які мають той самий хост призначення за останні дві секунди, а також оцінюються статистичні дані щодо поведінки протоколу, служби тощо.
- Аналогічно досліджуються параметри сервісів, які мають той самий сервіс як поточне з'єднання за останні дві секунди.

Безліч атак, включених у набір даних, потрапляє до наступних чотирьох основних категорій:

1.Атаки відмови в обслуговуванні (DoS): атаки відмови в обслуговуванні -

це атаки, в яких атакуючий займає деякий обчислення або ресурс пам'яті, тим самим роблячи його повністю недоступним для виконання законних вимог, та/або відхилення запитів на використання даного ресурсу від усіх законних користувачів.

2.User to Root Атаки: User to Root Атаки –категорія атак, де зловмисник, отримуючи доступ до облікового запису користувача системи (наприклад, в результаті підбору пароля) використовує деяку вразливість у системі для отримання прав адміністратора.

3. Remote to User Атаки: Remote to User Атаки мають місце, коли у атакуючого є можливість відправити пакети по мережі на віддалену машину, але немає облікового запису на даній машині. У такому випадку зловмисник використовує деяку вразливість для отримання локального доступу, як користувач віддаленої машини.

4.Probe: Зондування - категорія атак, де зловмисники досліджує мережу, щоб зібрати інформацію або виявити відомі вразливості, наприклад, сканування портів. Ці мережеві дослідження є особливо важливими для атакуючого, який збирається реалізувати атаку в майбутньому. Якщо у атакуючого є інформація про те, які пристрої та служби використовуються в даній мережі, він може використовувати цю інформацію для пошуку вразливостей.

У таблиці 3.1 представлені атаки, що у чотири основні категорії. У таблиці 3.2 представлений повний перелік параметрів, притаманних TCP-з’єднання.

Таблиця 3.1 – Типи атак, розподілені за чотирма основними категоріями

| Категорія | Приклади атак |
|---|---|
| 1 | 2 |
| Denial of Service Attacks | Back, land, neptune, pod, smurf, teardrop |
| User to Root Attacks | Buffer_overflow, loadmodule, perl, rootkit |

Кінець таблиці 3.1

| 1 | 2 |
|---|---|
| Remote to Local Attacks | Ftp_write, guess_passwd, imap, multihop, phf, spy, warezclient,warezmaster |
| Probes | Satan, ipsweep, nmap, portsweep |

Таблиця 3.2 - Список параметрів, представлений в множині KDDcup99

| № | Назва параметра | Опис | Тип |
|---|---|---|---|
| 1 | 2 | 3 | 4 |
| 1 | duration | Тривалість з'єднання (в секундах) | Безперервний |
| 2 | protocol_type | Тип протоколу (наприклад, tcp, udp тощо) | Символьний |
| 3 | service | Сервіс прикладного рівня (http, telnet тощо) | Символьний |
| 4 | flag | Статус з'єднання (normal або error) | Символьний |
| 5 | src_bytes | Число байт від джерела до одержувача (вхідний потік) | Безперервний |
| 6 | dst_bytes | Число байт від одержувача до джерела (вихідний потік) | Безперервний |
| 7 | Land | Якщо адреси збігаються 1(з'єднання з/на цей хост/порт); 0 інакше | Символьний |
| 8 | wrong_fragment | Кількість пошкоджених фрагментів | Безперервний |
| 9 | urgent | Кількість термінових TCP-пакетів, що містять важливі дані | Безперервний |

Продовження таблиці 3.2

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| 10 | hot | Кількість «гарячих» індикаторів | Безперервний |
| 11 | num_failed_logins | Кількість невдалих спроб входу до системи | Безперервний |
| 12 | logged_in | 1, якщо успішно увійшли до системи; 0 інакше | Символьний |
| 13 | num_compromised | Кількість «зламаних» умов | Безперервний |
| № | Назва параметра | Опис | Тип |
| 14 | root_shell | 1, якщо отримано права адміністратора; 0 інакше | Безперервний |
| 15 | su_attempted | 1, якщо була спроба введення команди su root; 0 інакше | Безперервний |
| 16 | num_root | Кількість «root» звернень | Безперервний |
| 17 | num_file_creations | Кількість операцій створення файлу | Безперервний |
| 18 | num_shells | Число спроб використання командного рядка | Безперервний |
| 19 | num_access_files | Число операцій з файлами керування доступом | Безперервний |
| 20 | num_outbound_cmds | Кількість вихідних команд у сеансі ftp | Безперервний |
| 21 | is_hot_login | 1, якщо вхід до системи належить "гарячому" списку; 0 інакше | Символьний |
| 22 | is_guest_login | 1, якщо вхід до системи - "гостьовий"; 0 інакше | Символьний |
| 23 | count | кількість з'єднань з тим же вузлом за останні дві секунди | Безперервний |

Кінець таблиці 3.2

| 1 | 2 | 3 | 4 |
|---|---|---|---|
| 24 | srv_count | кількість з'єднань з тим самим сервісом за останні дві секунди | Безперервний |
| 25 | serror_rate | % з'єднань, які мають помилки "SYN" | Безперервний |
| 26 | srv_serror_rate | % з'єднань, які мають помилки "SYN" | Безперервний |
| 27 | rerror_rate | % з'єднань, які мають помилки "REJ" | Безперервний |
| 28 | srv_rerror_rate | % з'єднань, які мають помилки "REJ" | Безперервний |
| 29 | same_srv_rate | % з'єднань з одним сервісом | Безперервний |
| 30 | diff_srv_rate | % з'єднань із різними сервісами | Безперервний |
| 31 | srv_diff_host_rate | % з'єднань з різними вузлами | Безперервний |
| 32 | dst_host_count | count для вузла призначення | Безперервний |
| 33 | dst_host_srv_count | srv_count вузла призначення | Безперервний |
| 34 | dst_host_same_srv_rate | same_srv_rate вузла призначення | Безперервний |
| 35 | dst_host_diff_srv_rate | diff_srv_rate вузла призначення | Безперервний |
| 36 | dst_host_same_src_ port_rate | same_src_port_rate вузла призначення | Безперервний |
| 37 | dst_host_srv_diff_ host_rate | diff_host_rate вузла призначення | Безперервний |
| 38 | dst_host_serror_rate | serror_rate вузла призначення | Безперервний |
| 39 | dst_host_srv_serror _rate | srv_serror_rate вузла призначення | Безперервний |
| 40 | dst_host_rerror_rate | rerror_rate вузла призначення | Безперервний |
| 41 | dst_host_srv_rerror _rate | srv_rerror_rate вузла призначення | Безперервний |

## 3.3 Класифікація навчальних даних

Перший компонент запропонованої системи відповідає за поділ вхідних даних на кілька класів, зважаючи на різні атаки, включені в набір вихідних даних для виявлення вторгнень. Набір даних для аналізу поведінки системи виявлення вторгнень є безліч KDD-Cup 99. Детальний аналіз даних KDD-Cup 99 наведено в розділі 3.2. Дані KDD-Cup 99 складаються з чотирьох різних типів атак і нормальних даних, кожна із записів має 41 параметр, які можуть бути представлені як безперервними, так і символьними значеннями. Потім, набір даних (D) ділиться на два під множення на основі мітки класу, встановленої у вихідній множині $D=\{D_i;1\leq i\leq 2\}$. Мітка класу описує належність запису до однієї з чотирьох основних атак (відмова в обслуговуванні, R2L, U2Ri Probe) або до множини нормальних даних. У розробленій системі навчальні дані розбираються на два класи: normal і anomaly. Потім ці два підмножини даних використовуються для генерації дерева виведення, яке згодом буде використано як основу нечітких правил, так щоб нечітка система могла ефективно застосовувати правила.

## 3.4 Стратегія формування нечітких правил

Зазвичай, нечіткі правила, що використовуються в нечіткій системі, складаються вручну або із залученням експертів, які формулюють їх на основі емпіричних даних з аналізу вторгнень або власних знань. У випадку з TCP-трафіком дуже трудомістко складати нечіткі правила вручну через те, що обсяг вхідних даних величезний і необхідно брати до уваги багато параметрів, а також їх усілякі комбінації. На жаль, зараз є кілька досліджень з прикладами автоматичної генерацією нечітких продукційних правил. Керуючись цим фактом, необхідно визначити метод вибору найкращого набору правил.

### 3.4.1 Стратегія автоматичної генерації правил

Розглянемо стратегію автоматичної генерації нечітких правил задля забезпечення ефективності процесу навчання [11]. Пропонована система призначена лише безперервних атрибутів. Основні атрибути KDD-Cup є безперервними. Таким чином, будуть використовуватися лише 34 параметри із вхідного набору даних. Даний метод включає в себе наступні етапи: 1.Виділення елементів одиничної довжини, що часто зустрічаються На підставі часто зустрічаються атрибутів, виявлених в обох класах вхідних даних, для безлічі KDD-cup 99 виділяються істотно-значущі атрибути. Зазвичай значущі параметри визначаються за допомогою різних стандартних алгоритмів інтелектуального аналізу, таких як Apriori [12] та FP-Growth [13]. Ці алгоритми підходять для виявлення часто зустрічаються значень зі змінною довжиною тільки для двійкової бази даних, що містить виключно двійкові значення. Але вхідний набір даних (KDD cup-99) містить безперервні змінні для кожного з параметрів, так що, звичайні алгоритми не підходять виявлення значущих елементів. Відповідно до цієї властивості, для кожного атрибуту необхідно зіставити атомарний елемент, шляхом визначення значень атрибуту, що найчастіше зустрічаються, а також задати мінімальне відхилення для цього значення. Такі параметри, що часто зустрічаються, необхідно визначити для двох класів, а саме, для нормального трафіку і аномального (об'єднання чотирьох типів атак).

2.Виділення відповідних атрибутів для генерації правил. На цьому кроці вибираються найвідповідніші атрибути, які дозволять визначити, до якого з класів варто віднести запис. Причиною для цього кроку є те, що вхідні дані містять 34 атрибути, не всі з яких можуть бути досить ефективними при виявленні вторгнення. Для виділення відповідного атрибута застосовується спосіб відхилення. Спочатку виділені на минулому етапі атомарні елементи зберігаються у вектор так, щоб для кожного класу були отримані 34 вектори (клас 1 і клас 2), представлені як, $C_i=[V_1,V_2,...,V_j,..., V_{34}]$, де $i=1$(належать до нормальної поведінки), $i=2$(належать до атаки). Кожен вектор ($V_j$) містить атрибути, що часто зустрічаються, частота появи яких перевищує мінімальний поріг. $V_j=\{f_i;1\leq i\leq m\};support(f_i)\geq min\_sup$. Потім для кожного атрибута

визначається діапазон відхилення елементів, шляхом порівняння частоти {max,min} для кожного вектора.

$$D_{v(j)}=\{f_{max},f_{min}\};$$

де $f_{max}=Max(f_j)$; $f_{min}=Min(f_j)$. Потім виконується порівняння один до одного між двома класами векторів для визначення значущих атрибутів. Атрибути, які не містять ідентичні ***{max.min}***діапазони, вибираються як значущі атрибути, вони гарантують найвищий рівень виявлення, порівняно з використанням всіх атрибутів для класифікації. Ефективні атрибути, вибрані для генерації правил надаються таким чином:

$$C_j=[V^{(1)},V^{(2)},\ldots,V^{(j)},\ldots,V^{(k)}]$$

де $k \leq 34$.

3.Генерація правил. На наступному етапі значущі атрибути, вибрані на попередньому кроці з множини ***{max,min}***, використовується для генерації правил. Порівнюючи відхилення значущих ознак між нормальними та аномальними даними, для значних атрибутів визначаються точки перетину. Завдяки використанню цих двох точок перетину, генеруються певні та невизначені правила. Наприклад, ***{max,min}***відхилення діапазону для нормальних даних, що відносяться до атрибут1 {1, 5} та ***{max,min}***відхилення для атаки, що відповідає атрибут 1 {2, 8}. Потім, сформулюється правило, "ЯКЩО атрибут 1 більше, ніж 5, ТО це атака", ЯКЩО атрибут 1 знаходиться між 2 і 5, ТО це або нормальні дані АБО атака" і "ЯКЩО атрибут1 менше, ніж 2, ТО це нормальні дані". Деякі дані містять лише одну точку перетину, на основі якої сформулюються лише два правила.

4.Фільтрування правил. Для того щоб підвищити ефективність застосування нечітких правил і розробити компактну та інтерпретовану систему класифікації, слід звернути увагу на два критерії, наведені в [14, 15]:

– Кількість нечітких правил має бути зменшена, наскільки це можливо;

– умовна частина нечітких правил («ЯКЩО» ...) має бути короткою.

Ґрунтуючись на цих двох критеріях, необхідно відфільтрувати правила таким чином, щоб вибирати максимально короткі і якомога меншу кількість правил. Правила, сформовані на попередньому етапі, можуть бути визначеними та невизначеними. Певні правила, містять лише один критерій у «ТО» частини правила, а невизначені правила - два критерії в «ТО» частини правила. Запропонована методика фільтрації правил відфільтрує невизначені правила, і вибирає лише певні правила для навчання нечіткої системи.

5.Генерація нечітких правил.Зазвичай, нечіткі правила, визначені в нечіткій системі складаються вручну або це можуть бути експертні дані. Але, в запропонованій системі, пропонується автоматично виділяти нечіткі правила, ґрунтуючись на виділених часто зустрічаються елементах. Нечіткі правила генеруються з певних правил, де умовна «ЯКЩО» частина цього правила є числовою змінною, а «ТО» частина – це мітка класу, пов'язана або з атакою, або з нормальним трафіком. Але, нечіткі правила повинні включати лише лінгвістичні змінні. Для того, щоб сформувати нечіткі правила з певних, слід фазифікувати чисельну змінну певних правил і «ТО» частина нечітких правил залишити такою ж, як наслідок певного правила. Наприклад, «ЯКЩО атрибут1 є H, ТО це атака» і «IF атрибут1 є VL, ТО це нормальні дані». Ці нечіткі правила використовуються для навчання нечіткої системи, що дозволить підвищити ефективність запропонованої системи в порівнянні з використанням нечітких правил без будь-яких власних методів.

### 3.4.2 Формування правил із використанням ПС Weka

В даній роботі було вирішено не використовувати описану розділ 3.4.1 стратегію генерації правил, вона виключає з правил усі рядкові змінні, такі як тип протоколу, назва служби, прапори та ін. У результаті проведених досліджень було виявлено, що використання атрибуту «протокол» значно скорочує кількість правил у базі.

Методика формування правил: 1.Виділення значущих атрибутів Спочатку для вибору атрибутів використовувався кореляційний метод, представлений у програмному продукті Weka модулем CFSsubsetEvall. В результаті з 41 атрибута були обрані наступні 9:

- src_bytes;
- dst_bytes;
- logged_in;
- serror_rate;
- srv_serror_rate;
- diff_srv_rate;
- srv_diff_host_rate;
- dst_host_srv_diff_host_rate;
- dst_host_srv_rerror_rate.

Найбільш суттєвими є показники класифікації «Коректно класифіковані записи» та «Невірно класифіковані записи», а так само «Розмір дерева виводу», «Кількість листя в деревині». Таблиці 3.3 представлені отримані значення.

Таблиця 3.3 - Результати для 9 атрибутів

| Показник | Значення |
|---|---|
| 1 | 2 |
| Коректно класифіковані записи | 99,56 % |
| Невірно класифіковані записи | 0.04 % |
| Розмір дерева виводу | 267 |
| Кількість листя у дереві виводу | 134 |

В результаті подальшої роботи від описаного більшості атрибутів довелося відмовитися, воно включає такі значення, як кількість переданих і отриманих байт. Дані атрибути є специфічними для конкретної тестової множини.

Виходячи із загального становища та спираючись на сукупність усіх раніше перерахованих вище і згаданих факторів , що значення для цих атрибутів практично рівномірно розподілені по області розподілу, виходить велика кількість нечітких множин і, як наслідок, база правил сильно збільшується. При використанні вищеописаного списку значущих атрибутів база правил містить близько двохсот правил. Було вирішено не використовувати метод CFSsubsetEvall, а натомість виділити значущі атрибути на основі інформації наведеної в [16], а також атрибутів визначених модулем CFSsubsetEvall, але без урахування src. Було проведено серію експериментів, їх результати представлені у Таблиці 3.4.

Таблиця 3.4 - Виділення значних атрибутів

| № | Порядкові номери атрибутів | Розмір дерева виводу | Кількість листя у дереві виводу | %, коректно класифікованих записів |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| 1 | 1,23,29,33 | 287 | 144 | 92,7 |
| 2 | 1, 2, 23, 29, 33 | 251 | 127 | 96,7 |
| 3 | 1, 2, 12, 23, 29, 33 | 201 | 102 | 97,1 |
| 4 | 1, 2, 12, 23, 26, 29, 33 | 211 | 107 | 97,4 |
| 5 | 1, 2, 12, 23, 25, 26, 29, 33 | 281 | 143 | 97,3 |
| 6 | 1, 2, 12, 23, 29, 33, 39 | 235 | 119 | 97,4 |
| 7 | 1, 2, 12, 23, 29, 33, 36, 41 | 605 | 309 | 98,6 |

Кінець таблиці 3.4

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| 8 | 1,2,12,23,29,33,41 | 349 | 176 | 97,6 |
| 9 | 1,2,4,12,23,29,33 | 302 | 175 | 97,6 |
| 10 | 1, 4, 12, 23, 29, 33 | 391 | 232 | 95,5 |
| 11 | 2, 12, 29, 33, 41 | 213 | 108 | 96,3 |
| 12 | 2, 12, 23, 29, 33, 41 | 313 | 158 | 97,5 |
| 13 | 2, 4, 12, 23, 29, 33,41 | 300 | 174 | 97,8 |
| 14 | 2, 12, 23, 29, 30, 33, 41 | 283 | 143 | 97,8 |
| 15 | 2, 12, 23, 29, 31, 33, 37, 41 | 445 | 226 | 98,2 |

Для виявлення набору значних атрибутів було необхідно, щоб їх число було мінімальним, зменшити кількість листя в дереві виведення і при цьому мати досить високий відсоток записів, що правильно класифікуються. Експериментальним шляхом було виділено множину №13, що складається з п'яти атрибутів:

- protocol type (тип протоколу);
- logged_in (чи успішно виконаний вхід до системи);
- count (число з'єднань з одним і тим же вузлом за останні 2 секунди);
- same_srv_rate (% з'єднань з одним і тим же сервісом);
- dst_host_srv_count (кількість з'єднань з тим же сервісом вузла призначення за останні 2 секунди).

2.Аналіз дерева рішень. Наступний етап у формуванні основи правил –це виділення найбільш «значних» листя з дерева виведення. На рисунку 3.3 представлений фрагмент дерева. У кожного аркуша дерева в дужках зазначено значення правильно класифікованих тестових записів та число помилкових

спрацьовувань. Для подальшого формування правил було обрано листя, у яких співвідношення помилково класифікованих записів до вірно розпізнаним не перевищує значення 0,15 і число записів, що під це правило більше 500.

```
dst_host_srv_count <= 254
|   same_srv_rate <= 0.33
|   |   protocol_type = tcp: anomaly (5549.0/18.0)
|   |   protocol_type = udp
|   |   |   dst_host_srv_count <= 1
|   |   |   |   same_srv_rate <= 0.06: anomaly (287.0/1.0)
|   |   |   |   same_srv_rate > 0.06
|   |   |   |   |   same_srv_rate <= 0.1: anomaly (7.0)
|   |   |   |   |   same_srv_rate > 0.1
|   |   |   |   |   |   count <= 7: anomaly (4.0)
|   |   |   |   |   |   count > 7: normal (5.0)
|   |   |   dst_host_srv_count > 1
|   |   |   |   count <= 12: anomaly (12.0/5.0)
|   |   |   |   count > 12: normal (111.0/9.0)
|   |   protocol_type = icmp: normal (3.0/1.0)
|   same_srv_rate > 0.33
|   |   protocol_type = tcp
|   |   |   logged_in = 0
|   |   |   |   dst_host_srv_count <= 82
|   |   |   |   |   same_srv_rate <= 0.65: anomaly (287.0/11.0)
|   |   |   |   |   same_srv_rate > 0.65
|   |   |   |   |   |   count <= 2
|   |   |   |   |   |   |   count <= 1: anomaly (1229.0/231.0)
|   |   |   |   |   |   |   count > 1
|   |   |   |   |   |   |   |   dst_host_srv_count <= 38: anomaly (279.0/51.0)
|   |   |   |   |   |   |   |   dst_host_srv_count > 38: normal (71.0/28.0)
|   |   |   |   |   |   count > 2
```

Рисунок 3.3 – Фрагмент дерева виводу у Weka

В результаті з дерева виводу було обрано 17 листків.

3. Формування певних правил (з цілими значеннями). Для кожного з листя було складено правило - в результаті проходження від листа до кореня дерева. Правило може включати як всі п'ять атрибутів, так і тільки деякі з них. На даному етапі було складено 17 правил.

4.Визначення нечітких множин кожного з атрибутів. Область визначення кожного атрибута розбирається на відрізки відповідно до значень, що є у правилах. На рисунку 3.4 наведено приклад розбиття області визначення змінної count на нечіткі множини. Залежно від числа записів, що потрапляють під

конкретне правило, деякі інтервали були об'єднані.

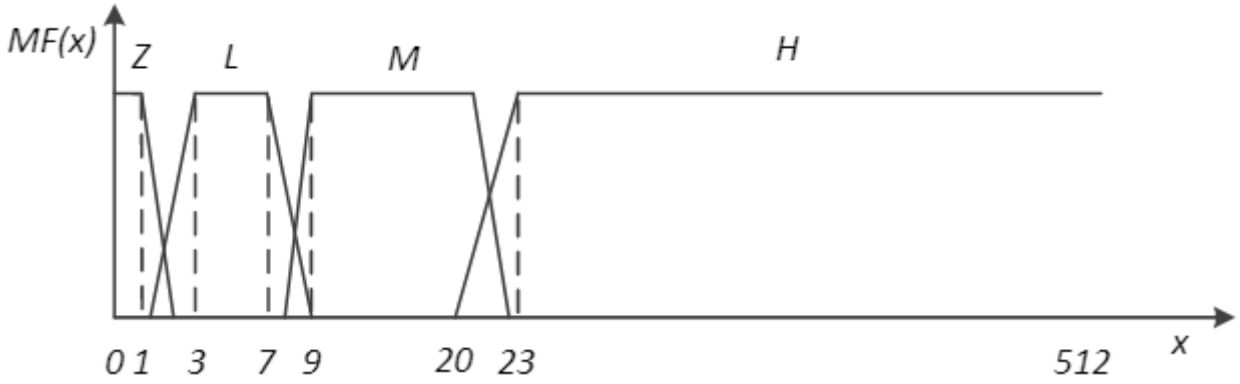



Рисунок 3.4 - Визначення нечітких множин атрибута count

В результаті введення нечіткості у правила у об'єктів з'являється можливість належати відразу кільком множинам, але з різним ступенем приналежності. Ця властивість є основною особливістю застосування нечіткої логіки.

5. Формування бази нечітких продукційних правил. Заключним етапом є процес формування бази нечітких продукційних правил. Внаслідок чого, кожному точному правилу ставиться у відповідність одне або кілька нечітких. Якщо правило з нечіткими змінними дотримується логічного оператора АБО - таке правило слід розбити на два. В результаті розбиття правил відповідним чином було отримано 55 нечітких продукційних правил. На рисунку 3.5 наведено приклад одного з них.

```
//--------------Rule 1 ----------------//
Rule rule1 = new Rule();
rule1.setConditions(
        new Condition(tp.getTermByName("prot_tcp"), vp.getVariableById(0, false)),
        new Condition(tp.getTermByName("login_zero"), vp.getVariableById(1, false)),
        new Condition(tp.getTermByName("count_low"), vp.getVariableById(2, false)),
        new Condition(tp.getTermByName("ssr_low"), vp.getVariableById(3, false)),
        new Condition(tp.getTermByName("dhs_low"), vp.getVariableById(4, false))
);
rule1.setConclusions(new Conclusion(tp.getTermByName("class anomaly"), vp.getVariableById(0, true)));
```

Рисунок 3.5 - Приклад нечіткого продукційного правила

## 3.5 Нечіткий модуль прийняття рішень

У цьому розділі описується система нечіткої логіки для знаходження відповідної мітки класу тестового набору даних. Лотфі Заде наприкінці 1960-х [15] ввів поняття нечіткої логіки.

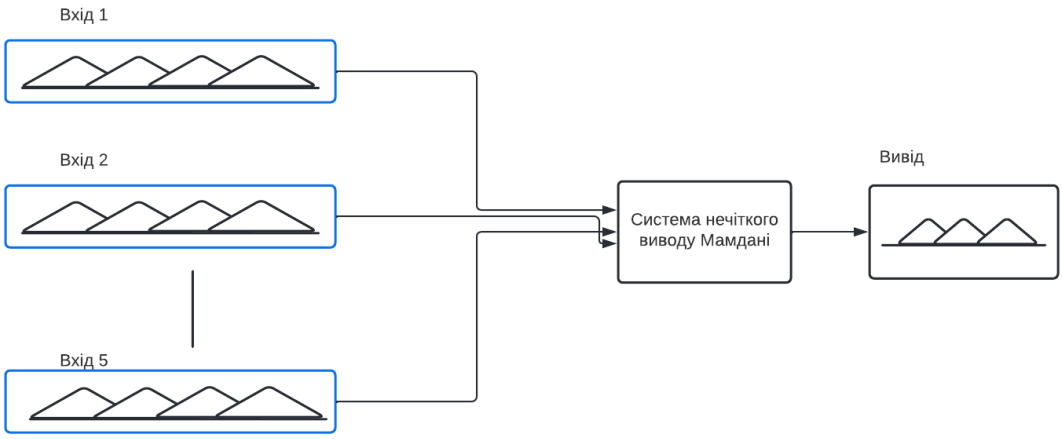



Рисунок 3.6 - Система нечіткого виводу

Нечітка система, зображена на рисунку 3.6, містить п'ять входів один вихід, де входи пов'язані з п'ятьма атрибутами, а вихід, пов'язаний з міткою класу (атака або нормальні дані). Застосовується система нечіткого виведення Мамдані з дефаззифікацією методом центру тяжкості (рисунок 3.7). Тут, кожне вхідне нечітке безліч, визначене в нечіткої системі, включає кілька функцій власності (залежно від різнорідності даних), і вихідне нечітка множина містить дві функції власності.

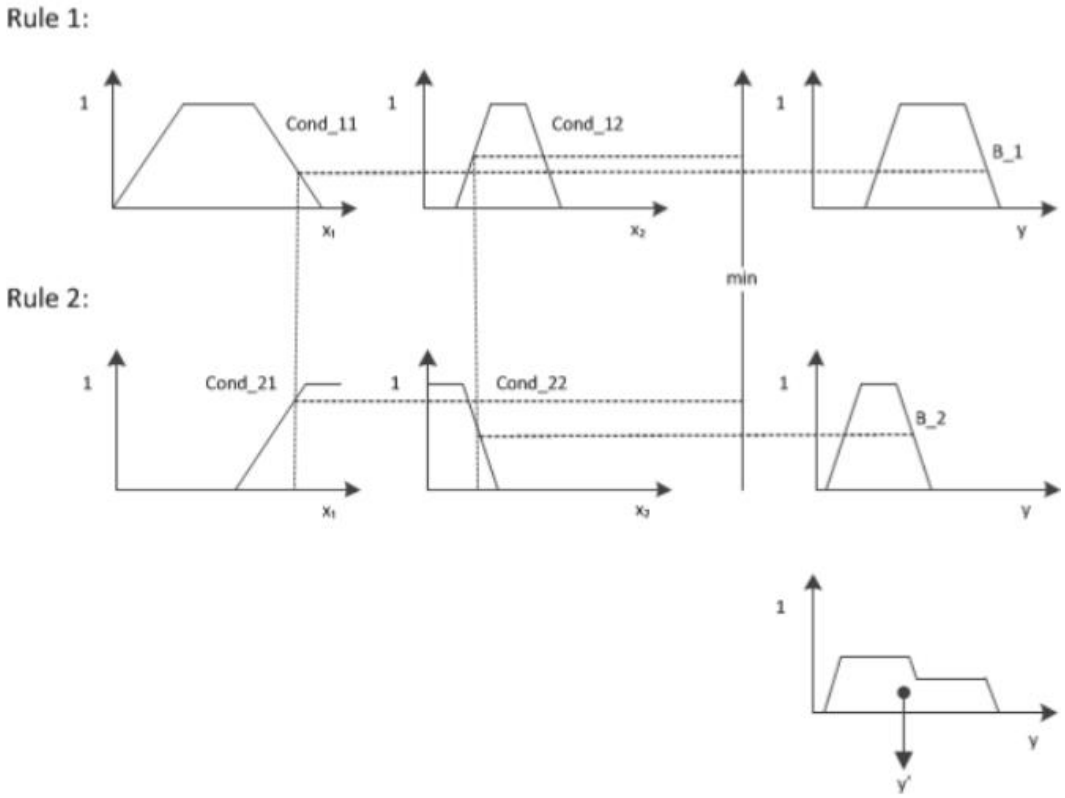



Рисунок 3.7 – Приклад нечіткого висновку Мамдані

Для фазифікації (розмивання) даних у системі застосовуються функції приналежності різних видів: трикутні, трапецієподібні та лінійні (вибір форми функції залежить від типу та специфіки даних, а також від діапазону значень, який необхідно покрити). Нечіткі правила, отримані на етапі, описаному в розділі 3.2.2, використовуються як база нечітких продукційних правил.

# 4 РЕЗУЛЬТАТИ РОБОТИ

## 4.1 Результати роботи алгоритму

Для тестування, на вхід системі нечіткої логіки, описаної в розділі 3.5, подаються тестові дані з множини KDD-cup 99. По-перше, тестові вхідні дані подаються на вхід фазифікатору, який перетворює 5 атрибутів (числові змінні) в лінгвістичні змінні, використовуючи відповідність функцію власності. Вихідні дані від фазифікатора подаються на вхід механізму логічного висновку, який, у свою чергу, порівнює цей конкретний вхід з базою правил. Виходом логічного висновку є одне із значень лінгвістичної змінної з наступної множини {Low and High}, потім дефаззифікатор перетворюється його до чіткого значення. Чітке значення, отримане від механізму нечіткого виведення, змінюється в діапазоні від 0 до 1, де «0» означає, що дані є повністю нормальними, а «1» вказує на те, що це однозначно атака.

Класи атак UsertoRoot і RemotetoLocal в оцінці результатів роботи системи не розглядалися, кількість записів їм у тестовому безлічі дуже мало(порядку 30-100 прим). У силу малої кількості записів для цих класів рекомендується створювати окремі підмножини правил. Тестування системи проводилося на випадкових наборах записів з кроком 300 штук (від 300 до 1500). У таблиці 5 представлені узагальнені дані кожного з класів.

Таблиця 5 - Зведена таблиця результатів тестування системи

| Клас | Ефективність, % | Хибні спрацьовування, % |
|---|---|---|
| 1 | 2 | 3 |
| Normal | 93.47 | 2.83 |
| DoS | 92.09 | 7.56 |
| Probes | 96.2 | 0.23 |

На рисунку 4.1 представлена діаграма, що ілюструє кількість правильно класифікованих записів.

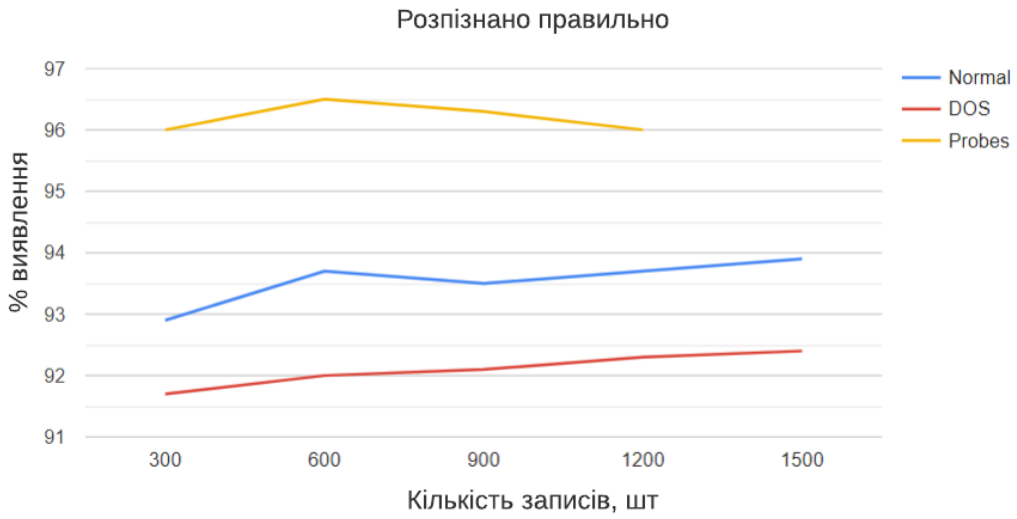



Рисунок 4.1 – Діаграма розпізнавання різних класів записів на тестовій вибірці

Рисунок 4.2 ілюструє відсоток помилкових спрацьовувань кожного з класів.

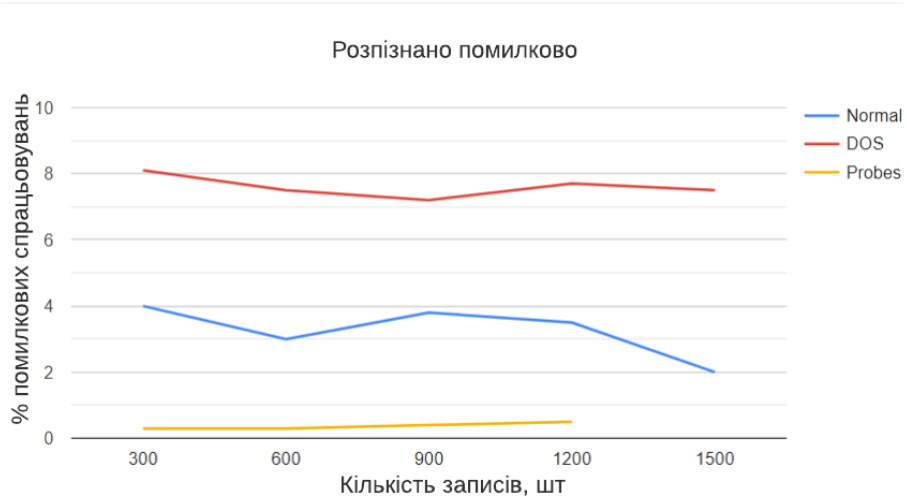



Рисунок 4.2 – Діаграма помилково класифікованих записів для кожного з класів на тестовій вибірці

На рисунку 4.3 представлено відсоткове співвідношення числа записів, котрим у основі нечітких продукційних правил був знайдено жодного відповідного. Відповідно, такі записи не були віднесені до жодного з двох класів атак або до нормального трафіку.

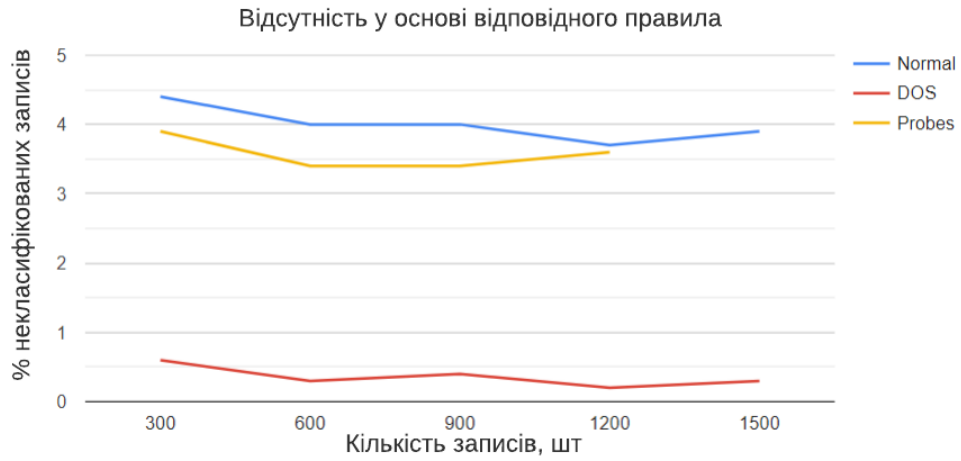



Рисунок 4.3 – Діаграма, що відображає відсоток записів тестової вибірки не підпадає під жодне правило з бази правил

Для оцінки результатів роботи прототипу були проаналізовані результати інших дослідників, які також використовували безліч KDD-cup99 для навчання та тестування СОР. У роботах [11, 17, 18] також розглядається можливість застосування нечіткої логіки виявлення мережевих атак. Для складання бази нечітких правил [17, 18] використовувався генетичний алгоритм. У таблиці 4.1 наведено результати виявлення.

Таблиця 4.1- Порівняння результатів правильно класифікованих записів, %

| Клас | Розроблений прототип | Робота[17] | Робота[18] | Робота[11] |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |
| Normal | 93.47% | 92,78 % | 69.5% | 90,3% |

Кінець таблиці 4.1

| 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|
| DoS | 92.09% | 98,91 % | 99.4% | 94,92% |
| Probes | 96.2% | 50,35 % | 71.1% | 90,9% |

4.2 Висновки за отриманими результатами

На основі даних, представлених у розділі 3.6, можна зробити висновок про те, що застосування нечіткої логіки в системах виявлення мережевих атак є досить ефективним. Отримані в результаті експериментів дані доводять це твердження.

Розроблений прототип показує результати не гірше, а для більшості класів атак краще, ніж існуючі аналоги.

З рисунків 4.1-4.3 видно, що зі збільшенням числа записів у вибірці зростає відсоток вірно розпізнаних класів, зменшується кількість помилкових спрацьовувань, а також зменшується, або, принаймні, не зростає відсоток записів, для яких у базі правил не існує жодного правила, на основі якого можна було б віднести запис до того чи іншого класу.

Однак у деяких випадках спостерігаються зміни у характері графіка, що можна інтерпретувати неоднорідністю розподілу прикладів у тестовій множині. Цей факт може бути обумовлений двома причинами: нерівномірний розподіл тестових записів за класами та ситуаціями, у яких не було активовано жодного правила з бази.

# ВИСНОВКИ

У роботі проведено дослідження існуючих методів виявлення вторгнень, особливу увагу приділено системам з нечітким висновком. Однією з основних особливостей нечітких систем є їх прозорість, яка досягається рахунок їх лінгвістичної інтерпретації як основи нечітких продукційних правил.

Алгоритм нечіткого висновку дозволяє забезпечити гнучкість системи прийняття рішень рахунок можливості коригування критеріїв оцінки та правил виведення.

Основною перевагою нечіткої логіки є те, що нечіткі правила дозволяють об'єктам належати кільком класам одночасно з різним ступенем належності.

У зв'язку з тим, що при зростанні кількості атрибутів кількість нечітких правил експоненційно зростає, з погляду продуктивності системи, недоцільне використання великої кількості атрибутів. У зв'язку з цим у процесі роботи було виділено підмножину значущих атрибутів, яке дозволило скласти базу з мінімальним числом простих нечітких правил, що водночас має високу точність виявлення і невелику кількість помилкових спрацьовувань. В результаті роботи сформована база нечітких продукційних правил, кожне з яких включає у собі трохи більше п'яти атрибутів. Крім того розроблено архітектуру та реалізовано прототип системи виявлення мережевих атак з нечітким висновком. Як навчальних і тестових даних використовувалося безліч KDD-cup99. Експерименти показали, що запропонований підхід дозволяє ідентифікувати різні класи атак. Точність розробленої системи класифікації даних досить висока, і можна порівняти з результатами аналогічних досліджень.

З метою подальшого вдосконалення запропонованого підходу є перспективним розглянути можливість формування п'яти множин нечітких правил – для кожного класу, а також можливість автоматизації процесу генерації бази нечітких продукційних правил.

Зокрема, можливе застосування генетичних алгоритмів для формування бази нечітких правил та функцій приналежності нечіткої системи.

## ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ


1. Dr. Fengmin Gong, “Deciphering Detection Techniques: Part II Anomaly-Based Intrusion Detection”, White Paper from McAfee Network Security Technologies Group, 2003.

2. Shaik Akbar, Dr.K.Nageswara Rao, Dr.J.A.Chandulal, “Intrusion Detection System Methodologies Based on Data Analysis”. International Journal of Computer Applications (0975 –8887) Volume 5–No.2, August 2010.

3. Cannady J, “Artificial Neural Networks for Misuse Detection”, in Proceedings of the ’98 National Information System Security Conference (NISSC’98), pp. 443-456, 1998.

4. Shon T, Seo J, and Moon J, “SVM Approach with A Genetic Algorithm for Network Intrusion Detection”, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Vol. 3733, pp. 224-233, 2005.

5. Yu Y, and Huang Hao, “An Ensemble Approach to Intrusion Detection Based on Improved Multi-Objective Genetic Algorithm”, Journal of Software, Vol.18, No.6, pp.1369-1378, June 2007.

6. J. Luo, and S. M. Bridges, “Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection”, International Journal of Intelligent Systems, Vol. 15, No. 8, pp. 687-704, 2000.

7. W. Lee, S. Stolfo, and K. Mok, “A Data Mining Framework for Building Intrusion Detection Model”, In Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, pp. 120-132, 1999.

8. Honig, A., Howard, A., Eskin, E., and Stolfo, S. J., “Adaptive Model Generation: An Architecture for the Deployment of Data Mining-Based Intrusion Detection Systems, Applications of Data Mining in Computer Security, Kluwer Academic Publishers, Boston, MA, pp. 154-191, 2002.

9.KDDCup1999 Data.[Електронний ресурс]URL:http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

10. Network Intrusion Detection System Using Fuzzy Logic. [Електронний ресурс]URL:http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.300.7185&rep=rep1&type=pdf

11. R. Agrawal, T. Imielinski, A., Swami, “Mining association rulesbetween sets of items in large databases”, in Proceedings of 1993 ACM SIGMOD Intl. Conf. on Management of Data, Washington, DC, pp. 207–216, 1993.

12. Iawei Han, Jian Pei, Yiwen Yin, Runying Mao, "Mining Frequent Patterns without Candidate Generation: A Frequent-Pattern Tree Approach", Data Mining and Knowledge Discovery, Vol: 8, No: 1, pp: 53 -87, 2004.

13. M. Saniee Abadeh, J. Habib and C. Lucas, “Intrusion detection using a fuzzy genetics-based learning algorithm”, Journal of Network and Computer Applications,vol.30, no.1, pp. 414–428, 2007.

14. Zadeh, L.A., “Fuzzy sets”. [Електронний ресурс] URL:http://www.cs.berkeley.edu/~zadeh/papers/Fuzzy%20Sets-Information%20and%20Control-1965.pdf

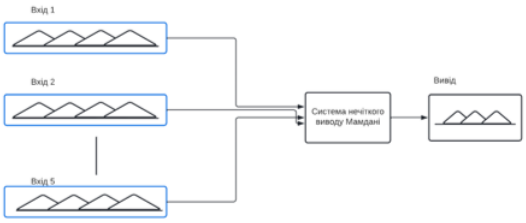
15. An Efficient Decision Tree Model for Classification of Attacks withFeatureSelection. [Електронний ресурс] URL:http://research.ijcaonline.org/volume84/number14/pxc3892967.pdf

16. Evolving Fuzzy Classifiers for Intrusion Detection. [Електронний ресурс] URL: http://www.docentes.unal.edu.co/jgomezpe/docs/papers/infassu2002.pdf

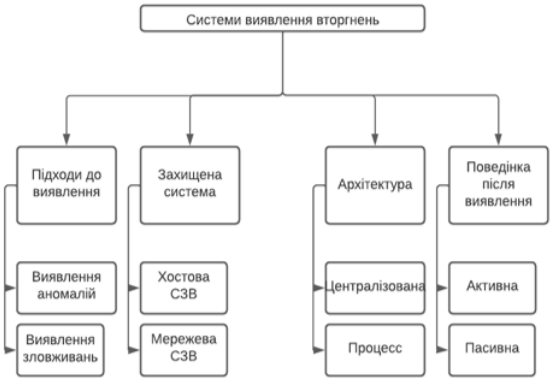
17. AnImplementation of Intrusion Detection System Using Genetic Algorithm. [Електронний ресурс] URL:http://airccse.org/journal/nsa/0312nsa08.pdf

КРКБ.180322.18.01.09 Е8

**Система нечіткого виводу**




**Класифікація систем виявлень вторгнень**




КРКБ.180132.18.01.09 Е8

Система виявлення вторгнень на основі нечіткого логічного висновку

Класифікація систем виявлень вторгнень

Система нечіткого виводу

Літера: Маса: Масштаб:

Аркуш 1

ХНУ, КБ-18-1

КРКБ.180322.18.01.09 Е8

**Основні етапи роботи системи виявлення вторгнень**


**Основні етапи нечіткого висновку**


КРКБ.180132.18.01.09 Е8

Системи виявлення вторгнень на основі нечіткого логічного висновку

Основні етапи роботи системи виявлення вторгнень

Основні етапи нечіткого висновку

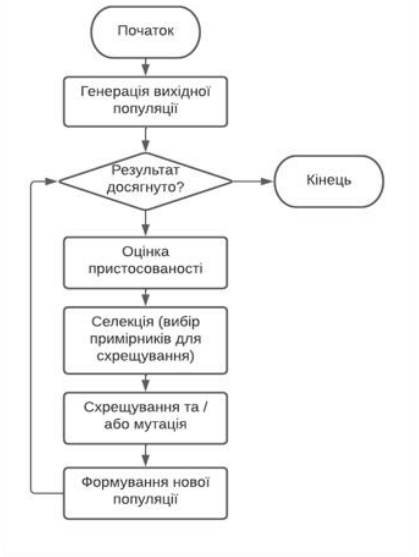
Літера: Маса: Масштаб:

Аркуш 1

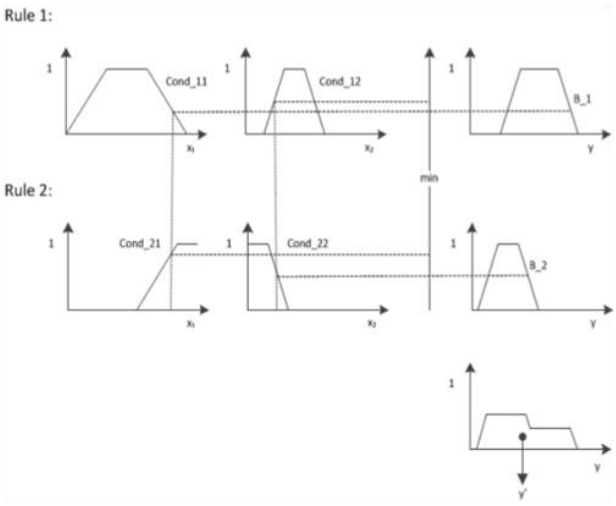
ХНУ, КБ-18-1

КРКБ.180322.18.01.09 ПЗ

**Блок-схема роботи генетичного алгоритму**




**Приклад нечіткого висновку Мамдані**




КРКБ.180132.18.01.09 ПЗ

Система виявлення вторгнень на основі нечіткого логічного висновку

Блок-схема роботи генетичного алгоритму
Приклад нечіткого висновку Мамдані

Літера: Маса: Масштаб:
Аркушів: 1

ХНУ, КБ-18-1

РІШЕННЯ ЕКСПЕРНОЇ КОМІСІЇ

КАФЕДРИ КІБЕРБЕЗПЕКИ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Система виявлення вторгнень на основі нечіткого логічного висновку

Автор: Огородник Максим Костянтинович

Спеціальність: 125 – Кібербезпека

Науковий керівник: Тітова Вера Юріївна, к.т.н., доцент

Після аналізу звіту подібності зроблено такий висновок:

| № | Висновок | Позначка про відповідність |
|---|---|---|
| 1 | Запозичення, виявлені в роботі, є законними і не є плагіатом (далі – зазначаються підстави віднесення запозичень до правомірних). Робота приймається до захисту. | відповідає |
| 2 | Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи (далі – зазначаються детальні та аргументовані підстави віднесення запозичень до правомірних). Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи. | |
| 3 | Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат. | |
| 4 | Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту. | |
| 5 | Інше: | |

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) Переважна більшість посилань на плагіат прийшлось на текстове наповнення рамок, яке є стандартним у відповідності до ДСТУ;
2) усі інші запозичення фрагментарні, є загально відомою інформацією або мають належним чином оформленні посилання.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 12,8%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи — В.Ю. Тітова

Гарант ОПП — В.М. Чешун

Завідувач кафедри Кб — Ю.П. Кльоц

Дата: 09.06.2022

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

**РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ**

освітньо-кваліфікаційного рівня «бакалавр»

Бакалавр Огородник Максим Костянтинович

Тема Система виявлення вторгнень на основі нечіткого логічного висновку

Спеціальність 125 – Кібербезпека

**Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «бакалавр»:** кількість листів креслень 3; кількість сторінок записки 64

1. Короткий зміст КР та прийнятих рішень Дана кваліфікаційна робота присвячена розробці системи виявлення вторгнення на основі нечіткого логічного висновку, визначенню значних атрибутів, формування нечітких правил, розробка прототипу програмного комплексу і тестування прототипу системи.

2. Висновок про відповідність КР завданню Кваліфікаційна робота у повній мірі відповідає поставленому завданню як в теоретичній, так і в практичній частині роботи

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У вступі подана загальна характеристика поставленої задачі, сформульована актуальність. Визначені задачі, які необхідно вирішити для досягнення поставленої мети, практична цінність отриманих результатів. У першому розділі проведено огляд систем виявлення вторгнень, проведено їх класифікацію. В другому розділі розглянуто нечіткі продукційні моделі класифікації атак в системах виявлення вторгнень. В третьому розділі реалізовано систему виявлення вторгнень із використанням нечіткої логіки. Четвертий розділ присвячено тестуванню розробленої системи та висновкам за отриманими результатами.

4. Позитивні сторони проекту Кваліфікаційна робота має практичну цінність, яка полягає у виявленні мережних атак із застосуванням нечіткої логіки, визначенні атрибутів атак, формування правил реалізації сценаріїв атак, розробці прототипу системи виявлення вторгнень та її тестування на реальних прикладах.

5. Негативні сторони проекту немає.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічне оформлення виконане відповідно до теми кваліфікаційної роботи з дотриманням стандартів. В загальному графічне оформлення виконане якісно, пояснювальна записка відповідає нормам щодо її оформлення.

7. Відгук про роботу в цілому В загальному кваліфікаційна робота заслуговує позитивної оцінки. Весь матеріал роботи структурований, чіткий та послідовний. Усі розділи роботи послідовні та логічні, що дозволяє чітко розуміти викладений матеріал в рамках тематики кваліфікаційної роботи. Графічний матеріал дозволяє наочно побачити доцільність та ефективність рішень, які були прийняті за основу для досягнення поставленої мети.

8. Інші зауваження немає

9. Оцінка кваліфікаційної роботи Враховуючи всі позитивні та негативні сторони представленої кваліфікаційної роботи, можна зробити висновок, що вона заслуговує оцінку «добре».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи) д.т.н., професор, професор кафедри комп'ютерної інженерії та інформаційних систем Савенко Олег Станіславович

« 15 » 06 2022. (підпис)

UNICHECK

Ім'я користувача:
Кафедра кібербезпеки

ID перевірки:
1011588313

Дата перевірки:
15.06.2022 16:56:42 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
15.06.2022 16:57:42 EEST

ID користувача:
100008300

Назва документа: Записка Огородник

Кількість сторінок: 66 Кількість слів: 12912 Кількість символів: 91590 Розмір файлу: 1.82 MB ID файлу: 1011457486

## 12.8% Схожість

Найбільша схожість: 7.13% з джерелом з Бібліотеки (ID файлу: 1011340401)

| | |
|---|---|
| 10.15% Джерела з Інтернету | 42 |
| 8.77% Джерела з Бібліотеки | 35 |

## 0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

## 0% Вилучень

Немає вилучених джерел

## Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

| | |
|---|---|
| Замінені символи | 22 |

# Anti-Plagiarism v-15.257

**Максимальное совпадение с одним документом 1.0%**

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 12%**

| ID: 105577<br>Название: Система виявлення вторгнень на основі нечіткого логічного висновку<br>Добавлено в БД: 2022-06-15<br>Авторы: Огородник Максим Костянтинович<br>Руководители: Тітова В.Ю.<br>Консультанты:<br>Опоненты: | Документ | | Суммарное совпадение по Базе Данных | |
|---|---|---|---|---|
| | Символы | Лексемы | Символы | Лексемы |
| | 67763 | 573 | 741 (1%) | 8 (1%) |

Источник плагиата

| ID | Описание | Наличие плагиата в документе | |
|---|---|---|---|
| | | Символы | Лексемы |