Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет комп'ютерної інженерії та управління
(повна назва)

Кафедра електронних обчислювальних машин
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА

Пояснювальна записка

Рівень вищої освіти другий (магістерський)

Модель мультиагентної системи виявлення вторгнень

(тема)

Виконав:
здобувач 2 року навчання,
групи СПм-23-4
Вадим ДУБИХВІСТ
(власне ім'я, прізвище)

Спеціальність
123 «Комп'ютерна інженерія»
(код і повна назва спеціальності)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма
Системне програмування
(повна назва освітньої програми)

Керівник: доц. Олексій ЛЯШЕНКО
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри ЕОМ _____ Андрій КОВАЛЕНКО
(підпис) (власне ім'я, прізвище)

2025 р.

Харківський національний університет радіоелектроніки

Факультет комп’ютерної інженерії та управління

Кафедра електронних обчислювальних машин

Рівень вищої освіти другий (магістерський)

Спеціальність 123 «Комп’ютерна інженерія»
(код і повна назва)

Тип програми освітньо-наукова
(освітньо-професійна або освітньо-наукова)

Освітня програма Системне програмування
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
“____” _____ 20__ р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві Дубихвісту Вадиму Вячеславовичу
(прізвище, ім’я, по батькові)

1. Тема роботи Модель мультиагентної системи виявлення вторгнень

затверджена наказом по університету від “ 21 ” квітня 2025 р. № 296 Ст

2. Термін подання здобувачем роботи до екзаменаційної комісії 16 червня 2025 р.

3. Вхідні дані до роботи
Інтернет речей, безпека, система виявлення вторгнень, глибоке навчання, мультиагентна блокчейн, транспортний рівень, nsl-kdd, аналіз даних, виявлення атак

4. Перелік питань, що потрібно опрацювати у роботі
Теоретичні основи дослідження
Технології IDS
Розумна та безпечна система
Експерементальні дослдження та тестування
Висновки

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Слайд-презентація – 13 слайдів

6. Консультанти розділів роботи (заповнюється за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів роботи	Примітка
1	Отримання теми кваліфікаційної роботи	21.04.25	
2	Аналіз літератури	22.04.25-1.05.25	
3	Побудова гібридної моделі	2.05.25-15.05.25	
4	Тестування системи та отримання результатів	16.05.25-30.05.25	
5	Формування пояснювальної записки	31.05.25-04.06.25	
6	Перевірка на плагіат	05.06.25-06.06.25	
7	Рецензування роботи	07.06.25-12.06.25	
8	Подача роботи в ЕК	16.06.25	

Дата видачі завдання " 21 " квітня 2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ доц. Олексій ЛЯШЕНКО
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи: 61 с., 8 рис., 8 табл., 1 дод., 35 джерел.

ІНТЕРНЕТ РЕЧЕЙ, БЕЗПЕКА, СИСТЕМА ВИЯВЛЕННЯ ВТОРГНЕНЬ, ГЛИБОКЕ НАВЧАННЯ, МУЛЬТИАГЕНТНА СИСТЕМА, БЛОКЧЕЙН, ТРАНСПОРТНИЙ РІВЕНЬ, NSL-KDD, АНАЛІЗ ДАНИХ, ВИЯВЛЕННЯ АТАК.

Метою кваліфікаційної роботи розробка моделІ мультиагентної системи виявлення вторгнень.

У ході виконання кваліфікаційної роботи проведено аналіз функціонування систем виявлення вторгнень. Запропоновано модель мультиагентної системи виявлення вторгнень та розроблено модулі запропонованої системи. Проведено тестування створеної системи використовуючи набір даних NSL-KDD.

ABSTRACT

Master’s thesis: 61 pages, 6 figures, 6 tables, 1 appendices, 35 sources.

INTERNET OF THINGS, SECURITY, INTRUSION DETECTION SYSTEM, DEEP LEARNING, MULTI-AGENT SYSTEM, BLOCKCHAIN, TRANSPORT LAYER, NSL-KDD, DATA ANALYSIS, ATTACK DETECTION.

The major goal of this thesis is to develop a model of a multi-agent intrusion detection system.

During the qualification work, an analysis of the functioning of intrusion detection systems was carried out. A model of a multi-agent intrusion detection system was proposed and modules of the proposed system were developed. The created system was tested using the NSL-KDD dataset.

ЗМІСТ

СКОРОЧЕННЯ ТА УМОВНІ ПОЗНАКИ

ШНМ – штучні нейронні мережі

AI – штучний інтелект

DDoS – розподілена відмова в обслуговуванні

DL – глибоке навчання

DoS – відмова в обслуговуванні

IDS – система виявлення вторгнень

IoE – Інтернет усього

IoT – Інтернет речей

ВСТУП

З популярністю технології Інтернету речей (IoT) безпека мережі IoT стала важливою проблемою. Традиційні системи виявлення вторгнень мають свої обмеження при застосуванні до мережі IoT через обмеження ресурсів і складність. Це дослідження зосереджено на розробці, реалізації та тестуванні системи виявлення вторгнень, яка використовує гібридну стратегію розміщення на основі мультиагентної системи, блокчейну та алгоритмів глибокого навчання. Система складається з наступних модулів: збір даних, керування даними, аналіз і відповідь. Для тестування системи використовується набір даних NSL-KDD – виявлення знань і аналізу даних. Результати демонструють ефективність алгоритмів глибокого навчання при виявленні атак з транспортного рівня. Експеримент показує, що алгоритми глибокого навчання підходять для виявлення вторгнень у мережевому середовищі IoT.

1 ТЕОРЕТИЧНІ ОСНОВИ ДОСЛІДЖЕННЯ

1.1 Огляд теоретичних передумов безпеки IoT

Інтернет речей (IoT) можна розглядати як взаємопов’язану систему на основі затверджених протоколів, які обмінюються інформацією [1] між пристроями, що працюють через Інтернет. Останні досягнення в IoT вводять концепцію інтелектуальності в пристрої, датчики, будинки, вулиці і навіть міста. IoT є однією з провідних і зростаючих сфер сучасних обчислювальних і комунікаційних технологій, яка зробила значний внесок у різні сфери, від сільськогосподарського сектора до автоматизації транспортних засобів. Сьогодні IoT також називають [2] Інтернетом усього (IoE), оскільки він має справу з будь-яким типом підключених пристроїв у повсякденному житті. Очікується, що в 2025 році [2] кількість підключених пристроїв може досягти 21,5 млрд.

IoT – це комбінація [3] кількох рівнів, включаючи мережевий рівень. Конструкція мережевого рівня базується на традиційних рівнях Інтернет-зв’язку та головним чином відповідає за передачу пакетів даних між хостами. Крім того, мережевий рівень є складною та вразливою частиною архітектури IoT, що призводить до різноманітних проблем із безпекою. Тим не менш, існує кілька систем безпеки для вирішення проблем безпеки [4]. Ці структури вимагають інсталяції в архітектурі IoT та/або на пристроях, щоб пристрої працювали ефективно проти загроз безпеці. На жаль, більшість систем безпеки вимагають значної обчислювальної потужності, а також пам’яті [5]. Однак деякі підходи, такі як полегшене шифрування, механізми автентифікації, можуть бути використані для подолання обмежень [6].

Велика кількість вузлів, тобто хостів або пристроїв, підключених до IoT, є основною причиною проблем із безпекою, оскільки порушення безпеки, що відбувається в одному вузлі, може призвести до збою всієї

системи. Найпоширенішими загрозами безпеці, з якими стикається система IoT, є ботнети, атаки розподіленої відмови в обслуговуванні (DDoS), програми-вимагачі, віддалений запис, атаки маршрутизації та витік даних [1]. Для боротьби з атаками на пристрої IoT використання брандмауера вважається першою лінією захисту, але це не є ефективним рішенням через різноманітність і складність архітектури IoT.

Системи виявлення вторгнень (IDS) за останні десятиліття перетворилися на важливий інструмент кібербезпеки. Вперше концепцію IDS було запропоновано в 1980 році [7], де було сформульовано її базове визначення як системи для ідентифікації несанкціонованої активності в мережі. В контексті IoT під несанкціонованою активністю розуміють спроби неавторизованого доступу до вузлів мережі.

Сучасна архітектура IDS включає три ключові компоненти:

- агент збору даних – відповідає за моніторинг мережевого трафіку та збір інформації;
- аналітичний модуль – виявляє ознаки кібератак та генерує сповіщення;
- модуль реагування – виконує дії на основі результатів аналізу.

Незважаючи на значний прогрес у розвитку IDS, зростаюча складність кібератак, особливо в IoT-середовищах [8], вимагає нових підходів. Традиційні IDS демонструють обмежену ефективність при роботі з багаторівневими IoT-мережами.

Сучасні дослідження спрямовані на інтеграцію IDS з:

- розподіленими архітектурами;
- методами машинного навчання (ШНМ, глибоке навчання, навчання з підкріпленням);
- блокчейн-технологіями.

Особливу увагу приділяється подоланню обмежень традиційних нейромережних підходів. У цій роботі запропоновано нову методику, що

поєднує багатоагентну систему з блокчейн-технологією, з подальшою оцінкою її ефективності на відомих наборах даних.

Результати показують, що більшість атак генеруються з мережевого рівня або транспортного рівня [9, 10]. Основною метою цього дослідження є вивчення можливості застосування концепції блокчейну для мультиагентної системи (MAS).

Основним завданням цього дослідження є розробка рішення, створення інтелектуальної IDS, яка може виявляти зловмисників і запобігати атакам у середовищах IoT. Для досягнення цієї мети, по-перше, розглядається поточний стан моделей IDS на основі техніки машинного навчання. Щоб виявити потенційні проблеми з безпекою та конфіденційністю, пов’язані з атаками на системи IoT, які вже відбулися, розглядаються. Крім того, оцінюються недоліки існуючих пристроїв IoT. Нарешті, пропонується рішення, яке використовує потужність механізмів машинного навчання для боротьби з раптовими атаками несанкціонованих зловмисників на мережевому та транспортному рівнях. Щоб підвищити продуктивність IDS, вивчаються різні методи оптимізації для підвищення продуктивності IDS.

1.2 Огляд літературних джерел

Було проведено ретельний огляд літератури, щоб дослідити поточні дослідження в області продуктивності IDS для пристроїв IoT. Проведено детальний аналіз можливих загроз у просторі IoT та засобів їх протидії за допомогою IDS. Також були враховані протоколи зв’язку системи IoT.

1.2.1 Історія системи виявлення вторгнень

Як правило, IDS включає як програмні, так і апаратні механізми, і IDS відповідає за виявлення зловмисних дій шляхом моніторингу мережевих середовищ і систем. Іншими словами, IDS використовується для виявлення

кібератак і надання негайних сповіщень. Загалом IDS діє як захист мереж і систем. IDS зазвичай розгортається після брандмауера та використовується з системою запобігання вторгненням.

IDS – це не новий термін у галузі досліджень Інтернету речей щодо безпеки та конфіденційності. За останні роки з'явилася значна кількість публікацій. Експерти з кібербезпеки вже деякий час стурбовані безпекою та конфіденційністю середовищ IoT. Це призвело до введення концепції вбудовування IDS в архітектури та пристрої IoT для боротьби з кібератаками [11, 12, 13]. Дослідники здебільшого зацікавлені у винайденні нових механізмів і моделей протидії зловмисникам у звичайних мережевих протоколах. Однак традиційні механізми IDS несумісні з пристроями IoT, підключеними через IPv6 та інші складні мережеві структури [14]. Більш повне дослідження використання методів машинного навчання має важливе значення для IDS для забезпечення безпеки та захисту конфіденційності в IoT.

1.2.2 Класи IDS

IDS поділяють на дві основні категорії:

- IDS на основі хоста;
- мережевий IDS.

IDS на основі хосту виявляє вторгнення шляхом сканування журналу та записів аудиту. Цей тип IDS зазвичай використовується на важливих хостах для захисту безпеки хоста з усіх боків. Перевага IDS на основі хоста полягає в тому, що він надає більш детальну інформацію, знижує частоту помилкових тривог і має меншу складність, ніж IDS на основі мережі. Однак це знижує ефективність системи додатків і надмірно покладається на дані журналу та можливості моніторингу хоста [15].

Через характеристики IoT і оскільки багато пристроїв IoT можна підключити до мережі, мережеві IDS потребують уваги. Мережевий IDS

може виявляти ненормальну поведінку та потік даних у мережі, щоб знаходити потенційні вторгнення. Це не змінює конфігурацію хоста та не впливає на продуктивність бізнес-системи. Навіть якщо мережевий IDS вийде з ладу, це не вплине на нормальну роботу бізнесу. Проблема полягає в тому, що IDS на основі мережі перевіряє лише пряме підключення до сегмента мережі, не дивлячись на інші сегменти мережі. Також важко обробляти зашифровані сеанси за допомогою мережевих IDS [16].

1.2.3 Методи виявлення

Виходячи з природи різних атак вторгнень (рисунок 1.1), методи виявлення вторгнень класифікуються на чотири основні категорії: методи на основі сигнатур, методи на основі специфікацій, методи на основі аномалій і гібридні методи [16].

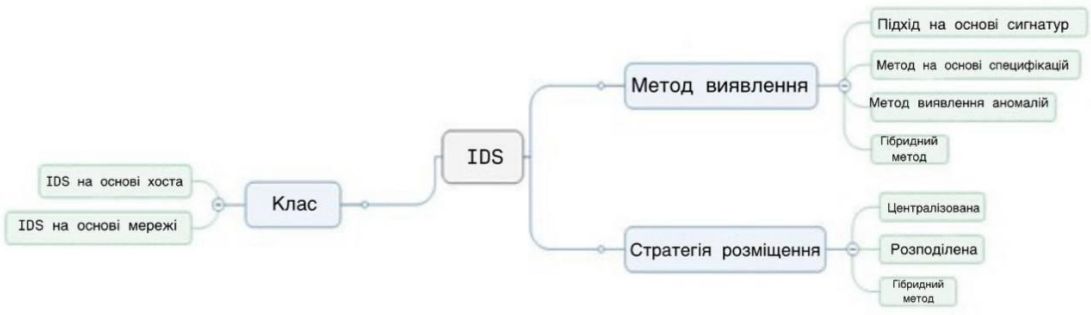

Рисунок 1.1 – Типи та розміщення систем виявлення вторгнень

Методи на основі сигнатур спочатку сканують дані в мережі та порівнюють їх із базою даних функцій. Якщо буде виявлено, що відскановані дані відповідають функціям бази даних сигнатур, ці дані розглядатимуться як вторгнення. Перевагою є те, що він може точно визначити тип атаки. Його відносно зручно використовувати, а попит на ресурси порівняно невеликий.

Методи, засновані на специфікаціях, вимагають від системних адміністраторів заздалегідь встановити правила та порогові значення. IDS визначає стан поточної системи та мережі згідно з правилами та пороговими

значеннями, встановленими адміністраторами [17]. У разі перевищення граничного значення або порушення правил IDS виявить нестандартну ситуацію та вживе відповідних заходів.

Методи на основі аномалій залежать від виявлення аномальних моделей і порівняння моделей трафіку. Перевага використання цього методу полягає в тому, що він дозволяє виявляти нові та невідомі вторгнення. Однак основним обмеженням є те, що метод, як правило, призводить до високого рівня хибнопозитивних результатів. Зараз дослідження зосереджені на застосуванні алгоритмів машинного навчання в методах виявлення вторгнень на основі аномалій, щоб підвищити надійність такого методу. Використовуючи алгоритми машинного навчання, методи виявлення вторгнень на основі аномалій можуть відстежувати поточні сліди вторгнень і порівнювати їх із наявними наборами даних, щоб бути попередженими щодо потенційних майбутніх атак.

Гібридні методи стосуються використання будь-якої комбінації вищезазначених методів виявлення в одній IDS. Цей підхід може допомогти подолати недоліки одного методу, тим самим підвищивши надійність усієї системи IoT. Однак очевидним недоліком є те, що вся IDS стане дуже великою та складною. Це ускладнить роботу всієї системи та зажадає більше ресурсів. Особливо, коли в системі IoT задіяно багато протоколів, процес виявлення вторгнень потребуватиме великих ресурсів і часу.

1.2.4 Стратегія розміщення

Централізована IDS складається з кількох вузлів моніторингу для розуміння поведінки хоста або мережі. Потім дані (або сповіщення, або системні журнали, або журнали мережі) будуть передані головному серверу або вузлу для подальшого аналізу. Коли система стає складнішою, навантаження на центральний сервер зростатиме, знижуючи продуктивність системи та потенційно навіть створюючи ризики для безпеки. У середовищі

IoT більша частина взаємодії даних відбувається на мережевому рівні та рівні сприйняття, а розподіл засобів IoT є складним і розосередженим. Тому централізована IDS не може ефективно виявляти вторгнення пристроїв IoT [18].

Розподілена IDS (DIDS) складається з кількох модулів, кожен з яких відповідає за різні завдання. Зазвичай вони розподіляють завдання моніторингу. Наприклад, у середовищі IoT загрози, з якими стикаються мережевий рівень, рівень сприйняття та рівень додатків, не однакові, тому відповідні модулі виявлення вторгнень також мають відрізнятися. Це може значно підвищити безпеку системи без необхідності надмірного зберігання та обміну даними між модулями. Основною перевагою є висока масштабованість, що означає, що його можна адаптувати до майбутніх середовищ IoT. Недоліком є те, що споживання ресурсів і витрати на зв'язок можуть бути високими [18].

Деякі гібридні стратегії розміщення були представлені в останніх дослідженнях. Перша стратегія гібридного розміщення розділяє мережу на різні регіони. Буде вузол, який має функцію виявлення вторгнень і стежить за вузлами в тому самому кластері. Відповідальністю цих вузлів також є моніторинг пакетів даних, що надходять від їхніх сусідніх вузлів. Якщо виявлено будь-яку аномальну поведінку, робиться висновок, що сусідні вузли були скомпрометовані зловмисниками. Прикордонний маршрутизатор збирає інформацію з вузлів і приймає остаточне рішення про вторгнення. Також слід зазначити, що гібридна IDS з розподіленими компонентами більш точна, ніж централізована IDS. Однак гібридна система має великі вимоги до ресурсів [19].

1.3 Попередні дослідження IDS для IoT

Удосконалення IDS IoT [17] є головною темою цього дослідження. У цьому розділі представлено огляд удосконалень і розробок IDS для IoT. Їх

можна класифікувати за цільовими загрозами, стратегією розміщення та методом виявлення.

У 2017 році в роботі [20] структуру неконтрольованого та гібридного виявлення вторгнень для розпізнавання селективного перенаправлення та атак із порожнечею для IoT. Він використовує метод напіврозрядної ситуації, переслідуючи проміжного агента на основі аномалій, розділеного декількома специфікаційними компонентами, які передають свою власну інформацію про місцезнаходження централізованому модулю розташування переривань, щоб передбачити будь-які невідповідності. Він застосовує неконтрольоване обчислення оптимальної траєкторії за допомогою технології MapReduce.

Також запропонувано використовувати сучасну техніку виявлення переривань для середовища IoT, що погоджується з демонстрацією автоматизації [21]. Ця стратегія розрізняє три типи IoT-атак: помилкові атаки, джем-атаки та атаки-відповіді. Це розширення іменованих структур переміщення. Організація цієї IDS може бути централізованим підходом, оскільки інформація, яка накопичується концентраторами організації, надсилається до модуля виявлення переривань і допомагає у створенні баз даних випадків. IDS використовує аналізатор випадків, заснований на стратегії на основі специфікації, для виявлення переривань. Порівнюючи попередньо зайняті потоки активності, цей IDS може вміло ідентифікувати джем-атаки, помилкові атаки та атаки-відповіді в мережах IoT.

В [22] також у 2017 році запропонував метод, який збирає багатоагентні системи та технологію блокчейну для посилення зв’язку між різними автономними агентами, задіяними в безпілотних літальних апаратах (БПЛА). Вони запропонували архітектурний протокол для включення зв’язку через технологію блокчейну Ethereum у однорангових підключених мережах. Результатом є протокол, який потенційно сумісний з автономними агентами. Запропонований протокол забезпечує безпеку процесу зв’язку та допомагає передбачити надійність змінних робочих станів. Це призвело до подальших досліджень поєднання технологій блокчейну з мультиагентними системами.

Дослідження [23] демонструє перспективний підхід до інтеграції блокчейн-технологій (BCT) з багатоагентними системами (MAS), що дозволяє досягти децентралізації довіри та усунути залежність від довірених третіх сторін – традиційних точок відмови в системах. Автори запропонували інноваційну модель прозорого управління репутацією на основі BCT, яка підвищує довіру до MAS.

Система інтегрує:

- агентну платформу JADE (Java Agent Development Framework);
- блокчейн-рішення на базі Hyperledger Fabric.

Ключові особливості реалізації: механізм управління ідентифікацією агентів через службу членства Hyperledger Fabric; прозорі методи обчислення та зберігання репутації агентів на основі:

- моделей агентної комунікації;
- оцінок взаємодії між агентами.

Використання смарт-контрактів для:

- моніторингу поведінки агентів;
- обчислення репутаційних показників.

Крім того, агенту асоціації пропонуються послуги з використанням книги. Впроваджено механізми для прозорого обчислення та зберігання репутацій агентів на основі моделей спілкування між агентами та оцінок взаємодії. Система оцінюється за допомогою різних тестів. Поведінка агента складається з автономної та залежної від користувача поведінки. Механізми розумних контрактів поєднуються з обчисленням і моніторингом репутації агентів на основі поведінки агентів. Завдяки доступній реалізації Hyperledger Fabric система є надійною та масштабованою. Тестування можна проводити легко та швидко завдяки класичному інтерфейсу командного рядка (графічний інтерфейс полегшив взаємодію). Хоча прототип інтеграції BCT і MAS був реалізований, технічні обмеження все ще існують. Це включає запобігання поодиноких точок збою системи шляхом окреслення розумного відображення між реальними об’єктами та розподіленими компонентами

базової технології блокчейну, застосування криптографічних рішень для підвищення безпеки та конфіденційності, перевірки точності впровадження смарт-контракту та затвердження технології блокчейну та агента в системах реального світу. Існують також етичні міркування для інтеграції ВСТ і MAS у реальних сценаріях. Існує прямий зв’язок між властивістю незмінності реєстру та прозорістю управління репутацією та досягненням області застосування, яка підвищує потужність користувачів системи та підтримує надійну взаємодію. Навмисна або випадкова несправність системи може поставити під загрозу конфіденційність користувачів, а фреймворк повністю покладається на розумні контракти. Незважаючи на те, що MAS із підтримкою ВСТ унеможливлює посередників у вирішенні конфліктів, питання надійності програмного забезпечення та процесу перевірки в середовищі ВСТ залишається відкритим.

2 ТЕХНОЛОГІЇ IDS

У цьому розділі озглядаються технологія, що лежить в основі запропонованої нами IDS, включаючи мультиагентну систему та застосування технології блокчейн.

2.1 Мультиагентна система

Агент описується як інтелектуальна поведінка комп'ютерного програмного забезпечення в галузі штучного інтелекту (ШІ) та інформатики. Термін «мультиагентний» зазвичай відноситься до MAS або мультиагентної технології (MAT). MAS – це набір кількох агентів, який є розширенням і застосуванням розподіленого штучного інтелекту (DAI) [24].

Складні системи можна спростити та модульувати за допомогою MAS. Агенти відповідають за координацію та комунікацію в рамках своїх відповідних завдань. Будучи повністю автономним і незалежним від форми, кожен агент може бути індивідуумом і кластером у MAS [25, 26]. Хоча агенти розроблені різними мовами та мають різні шаблони проектування, вони повинні використовувати стандартні режими зв'язку, які викликають взаємне спілкування, якого немає в системі одного агента.

Кожен агент працює, маючи відповідний набір властивостей і правил роботи. На основі цих правил дій вони виконують завдання під час роботи всієї MAS. Співпраця між агентами в MAS допомагає людям вирішувати деякі складні явища. Агенти повинні володіти наступними чотирма основними характеристиками: автономією, чуйністю, ініціативністю та соціальністю. Це в основному проявляється в інтелекті та агентурній здатності системи. Інтелект відноситься до здатності прикладної системи використовувати міркування, навчання та інші методи для аналізу та інтерпретації всіх видів інформації та знань [27]. Здатність агента означає

здатність агента сприймати повідомлення із зовнішнього світу та реагувати автономно відповідно до власних знань.

В [28] провели опитування, повідомляючи про найбільш релевантні сучасні внески в IoT обчислень на основі агентів (ABC), щоб оцінити придатність ABC для розробки IoT. Хоча автори заявили, що з точки зору обчислень, зберігання та зв'язку немає технологічних обмежень, які б перешкоджали повній реалізації екосистеми IoT, її багатогранні проблеми розвитку все ще потребують додаткової уваги. Аналіз показав, що, застосовуючи ABC як SO, так і IoT системне моделювання та програмування сумісності, автоматизму та розподіленого інтелекту можна виконувати різною мірою. Це моделювання має переваги перед іншими підходами, такими як об'єктно-орієнтована, сервіс-орієнтована та компонентно-орієнтована обчислювальні парадигми. Крім того, перевірка кількох варіантів дизайну може бути виконана моделюванням на основі агента перед фазою розгортання. У синергії з іншими парадигмами, такими як хмарні та периферійні обчислення, це можна виконувати систематично та ефективно за допомогою агентних методологій. Опитування підтвердило, що підхід до розробки на основі агентів є ефективним вибором для багатьох SO та систем IoT.

У цьому дослідженні запропонована система використовує платформу MAS JADE [29], яка забезпечує просту, але потужну модель виконання завдань і композицію. Він використовує асинхронну передачу повідомлень для зв'язку між агентами. JADE заснований на Java, тому він не залежить від платформи. JADE можна використовувати в різних Java-орієнтованих середовищах, таких як пристрої Android і пристрої J2ME-CLDC MIDP 1.0. Крім того, JADE дозволяє конфігурувати мережі, що характеризуються частковим підключенням.

2.2 Машинне навчання

2.2.1 Глибоке навчання

В останні роки термін «глибоке навчання» набув популярності серед дослідників, які працюють із методами машинного навчання на основі штучної нейронної мережі. Як і ШНМ, глибоке навчання надихається загальною структурою та функціональністю мозку. Глибоке навчання передбачає набір кількох шарів ШНМ, які згруповані. Він складається як з вхідного, так і з вихідного рівнів, створюючи багаторівневу мережу потоку даних. Глибоке навчання також описується терміном глибока нейронна мережа (DNN) або глибоке структуроване навчання [30]. Ключова відмінність між навчанням ШНМ і глибоким навчанням полягає в прихованих шарах. Приховані шари розташовані ієрархічно між вхідним і вихідним шарами. Глибоке навчання надійніше, ніж звичайна ШМН, оскільки воно обробляє та обчислює задані вхідні дані більшою мірою, ніж ШНМ. У реальному світі обсяг даних зростає, що призводить до їх ускладнення. Внутрішній характер глибокого навчання полягає в здатності вчитися на попередніх рівнях або наборі великих даних. Ця функція робить глибоке навчання дуже сильним і потужним кандидатом у виборі моделей машинного навчання, особливо в класифікації даних немаркованих вибіркових наборів даних [31].

2.2.2 Multi-Agent Reinforcement Learning

Методи навчання з підкріпленням (RL) [32] дозволяють комп’ютеру завершити завдання шляхом постійного навчання та навчання з самого початку. В останні роки, як Alpha Go, в якому DeepMind розробив і досяг успіху в складних завданнях, RL продемонстрував великий дослідницький потенціал [33]. RL має довгу історію, як Sutton et al. вже почали вивчати RL у

1979 році. Спочатку цей метод використовувався в галузі інтелектуального керування, але з подальшим розвитком науки й техніки RL набув широкого застосування в автопілотах, голосовій взаємодії та багатьох інших областях. RL має велике значення в області машинного навчання та обчислювального інтелекту.

Послідовність дій кількох агентів регулює середовище багатоагентних систем. Якщо агент не в змозі сприйняти взаємозв'язок між своїми діями та змінами середовища, він може створити нестандартне марковське середовище. Оскільки підхід RL визначається шаблонами проектування MAS, одночасний ізольований RL (CIRL) може бути використаний, коли одиночний агент не взаємодіє з іншими агентами і покладається лише на техніку неконтрольованого навчання. Навчання взаємодії кількох агентів можливе шляхом застосування інтерактивного RL. RL з одним агентом повинен розглядати лише тимчасові проблеми розподілу кредитів, тоді як RL з кількома агентами повинен розглядати структурні призначення кредитів [34].

2.3 Блокчейн

Децентралізація є основною мотивацією технології блокчейн [35]. Розподілена та прозора функція реєстру блокчейну означає, що збій одного вузла не впливає на всю систему. Блокчейн змінив структуру транзакційної мережі з зіркової на точкову (P2P). Ця трансформована структура дозволяє двом сторонам мати справу безпосередньо одна з одною за допомогою шифрування та безпеки на основі захисту коду та алгоритму. Оскільки сторони, задіяні в системі транзакцій, повинні лише довіряти використаному алгоритму для встановлення взаємної довіри, немає необхідності знати про надійність сторін [36]. Крім того, фреймворк не потребує жодного схвалення безпеки сторонніми агентами, оскільки алгоритм повністю відповідає за всі види автентифікації.

Ethereum та Hyperledger Fabric є двома найпопулярнішими платформами розробки блокчейн-додатків. Їх базові технології однакові. Фундаментальна відмінність між Ethereum і Hyperledger полягає в тому, як вони розроблені, і в цільових користувачах. Віртуальна машина Ethereum (EVM) доступна в Ethereum. Розумні контракти та загальнодоступні блокчейни націлені на програми, якими користуються звичайні споживачі. Hyperledger Fabric має дуже модульну архітектуру, яка більше підходить для бізнес-додатків. Це забезпечує велику гнучкість і вільніше застосовує бізнес-логіку. Мета – спростити робочі та торгові процеси за допомогою технології блокчейн, тобто вирішити проблему міжфірмового кредитування.

В запропоговану систему можна використовувати в мережах IoT різних масштабів, включаючи навіть складніші мережі. Однак у цій галузі ще потрібно провести додаткові дослідження. Хоча рівень точності моделі DNN щодо розрізнення різних типів атак досить високий, деякі рідкісні типи атак не можуть бути точно виявлені. У майбутньому потрібно буде вирішити тривалий період навчання агентів і потребу у великих навчальних наборах. Треба додатково дослідити, як кілька агентів можуть належним чином співпрацювати у великій мережі IoT. Великі вимоги до обчислювальної потужності також є проблемою, яку необхідно вирішити в майбутніх проектах.

3 РОЗУМНА ТА БЕЗПЕЧНА СИСТЕМА

В рамках кваліфікаційної роботи використовуються методології науково-дослідних проектних та науково-інженерних досліджень. Поєднуючи наукові дослідження з інженерним проектуванням та впровадженням, можна дослідити доцільність та цінність цієї нової цільової структури системи виявлення вторгнень.

3.1 Проектування системи

3.1.1 Системна перспектива

Ця система буде використовуватися для виявлення атак з поточного мережевого трафіку та моніторингу деталей стану мережі Інтернету речей. Веб-портал буде використовуватися для візуалізації звітів про виявлення та налаштування правил реагування. Система складається з п'яти частин: модуля смарт-контрактів блокчейну, модуля виявлення та аналізу, модуля реагування, модуля обробки даних та модуля збору. Системі потрібно буде взаємодіяти з пристроями Інтернету речей та контролювати потік даних цих пристроїв. Оскільки системі потрібні дані для виявлення шкідливої поведінки мережі Інтернету речей, їй потрібна база даних для зберігання даних. Система додаватиме та змінюватиме дані, тоді як веб-портал може лише збирати дані. Весь зв'язок з базою даних відбуватиметься через Інтернет.

3.1.2 Системні функції

Розумна та безпечна система дозволить адміністратору мережі Інтернету речей контролювати пристрої Інтернету речей на основі даних мережевого трафіку. Область, яка буде контролюватися, визначається на

основі критеріїв, визначених адміністратором. Адміністратор має кілька варіантів налаштування процесу моніторингу модуля збору системних даних. Зібрані дані будуть оброблені модулем обробки даних, який виконує перше виявлення атаки на основі класифікації ознак. Потім ці дані будуть розділені на дві частини: неідентифікований набір даних та навчальний набір даних. Ці два набори даних будуть переміщені до модуля виявлення та аналізу. Навчальний набір даних буде використано для навчання агента виявлення. Неідентифікований набір даних буде використано для аналізу продуктивності моделі. Всі результати будуть надіслані до модуля відповіді, який діє на основі правил конфігурації, встановлених адміністратором. Всі процеси будуть зберігатися в реєстрах блокчейнів хостів, які мають модуль даної системи.

3.1.3 Характеристики користувача

Адміністратор мережі Інтернету речей може використовувати систему для виявлення вторгнень, додавати дані, налаштовувати процес обробки даних та критерії виявлення, а також за потреби зменшувати кількість агентів. Для великомасштабних мереж Інтернету речей системні модулі будуть встановлені та виконані на різних хостах. Це означає, що кожен модуль потребує власного адміністратора. Ці адміністратори повинні мати можливість налаштовувати модулі, якими вони керують.

Система виявлення вторгнень, описана [36] на рисунку 3.1, використовує мультиагентну технологію. Кожен агент є відносно незалежною одиницею. Агенти поділяються на чотири різні типи модулів. Вони взаємодіють один з одним через комунікаційні агенти, що знаходяться в кожному модулі. Таким чином, кожен модуль може працювати відносно незалежно, що зменшує залежності між модулями. Вся система складається з модуля збору, модуля обробки даних, модуля виявлення та аналізу та модуля реагування. Система використовує мову комунікації FIPA-ACL на основі інтелектуальних фізичних агентів та агентів як мову комунікації агентів,

оскільки FIPA-ACL підтримується багатьма спільнотами. Чотири комунікаційні агенти будуть модифіковані та вдосконалені за допомогою інтерактивного навчання з підкріпленням. Це означає, що кожен агент залежить від інших агентів.

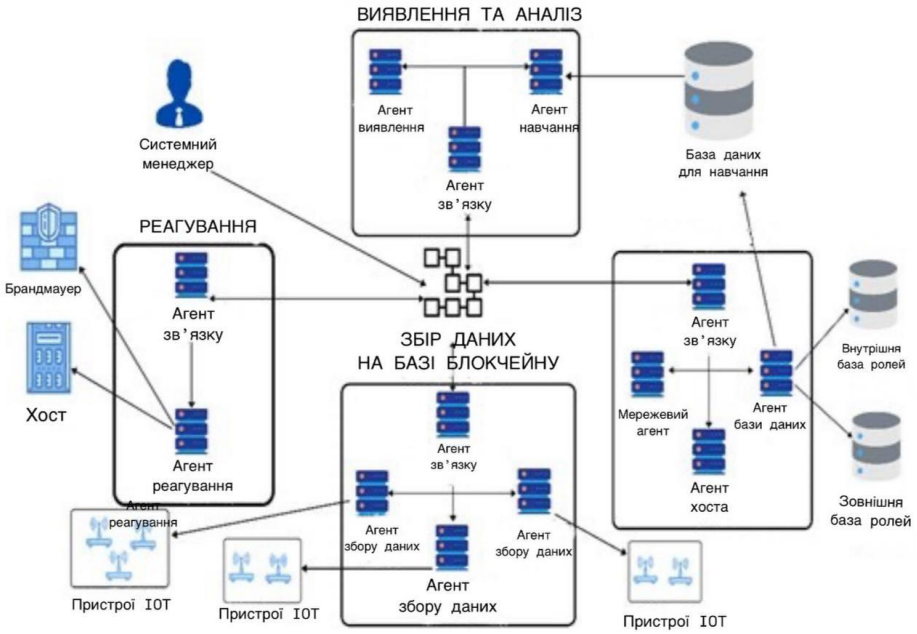

Рисунок 3.1 – Система виявлення вторгнень

Кожна успішна дія, виконана іншими агентами, створює зворотний зв'язок, який надсилається комунікаційному агенту в тому ж модулі. Комунікаційні агенти створюють звіт про зворотний зв'язок після збору відгуків від інших агентів у модулі. Звіти про зворотний зв'язок будуть використані для навчання комунікаційних агентів. Заслуги на зворотний зв'язок будуть присвоєні комунікаційним агентам на основі їхнього внеску. Кожна дія, яку виконують комунікаційні агенти, вважатиметься транзакцією. Оскільки комунікаційні агенти є єдиними агентами, які мають право видавати команди, більшість питань безпеки будуть порушені комунікаційними агентами. Транзакції будуть записані в блокчейні, і лише системний менеджер має доступ до смарт-контракту (ланцюжкового коду) цього блокчейну.

3.2 Розробка системи

Традиційні середовища Інтернету речей стикаються з багатьма проблемами, включаючи відсутність безпеки пристроїв, централізований контроль даних, жорстку архітектуру, відсутність сумісності комунікацій та труднощі у співпраці між кількома сторонами. Система виявлення вторгнень, що використовується для мереж Інтернету речей, повинна вирішувати ці проблеми.

Блокчейн може покращити безпеку пристроїв завдяки своєму децентралізованому способу збереження та передачі комунікаційних повідомлень, які є командами, що контролюють поведінку системи. Блокчейн також допомагає побудувати довіру між кількома сторонами, використовуючи узгоджений смарт-контракт для регулювання роботи системи та поведінки агентів. Агенти дозволяють системі масштабуватися зменшуватися та збільшуватися, що робить її більш придатною для роботи з різними типами структур середовища Інтернету речей та протоколами зв'язку. Оскільки масштаб мереж Інтернету речей швидко зростає, мережі Інтернету речей змінюються дуже швидко. Агенти можуть допомогти системі адаптуватися до цих змін, тоді як блокчейн може забезпечити безпеку великомасштабної мережі Інтернету речей.

3.2.1 Компонент блокчейну

У цій системі для забезпечення зв'язку між агентами використовується приватний ланцюг. Система інтегрує вбудовану базу даних та вузол блокчейну в одному агенті, запускаючи та зупиняючи їх одночасно. Останні взаємодії будуть додані до області кешу, що зменшить час пошуку та пришвидшить використання та зберігання даних на диску агентами. Зберігання копії бази даних (БД) у всіх агентах зменшить накладні витрати комунікаційного агента, виконуючи локальний пошук щойно оновленої

інформації. Комунікаційний агент модуля виявлення та аналізу є супервузлом для модуля блокчейну. Він містить весь реєстр блокчейну та використовується для того, щоб усі вузли в мережі могли отримати правильну копію та з'єднатися один з одним. Всі інші комунікаційні агенти будуть повними вузлами. Вони містять та розповсюджують весь реєстр блокчейну та є важливими для перевірки кожного запису зв'язку в блокчейні. Інші агенти в кожному модулі можуть бути активовані як легкі вузли та підключатися до комунікаційного агента, який є їхнім батьківським вузлом. Ці легкі вузли містять лише заголовки блоків і використовуються для перевірки, чи було внесено зміни до їх батьківського вузла. Модуль блокчейну використовується не лише для регулювання та перевірки поведінки комунікаційних агентів і забезпечення безпеки системи, але й для побудови довіри в мережах Інтернету речей, в яких задіяно кілька сторін, кожна з яких має своїх власних агентів. Крім того, дані з комунікаційних повідомлень можна використовувати для аналізу атак і покращення агентів шляхом застосування навчання з підкріпленням. Нижче описано класи Java, які ми використовували для реалізації блокчейну.

Обліковий запис агента. Об'єкт AgentAccount зберігатиме ім'я агента, його publicKey та privateKey. Ці ключі будуть згенеровані методом generate PublicAndPrivateKey. Ключі шифруються криптографічним алгоритмом ECDSA (Eliptic curve digital signature algorithm), який також використовується для шифрування облікових записів Bitcoin. Це гарантує, що лише правильний агент може генерувати певні комунікаційні повідомлення. Шляхом випадкової генерації відкритого та закритого ключів для кожного агента на основі криптографічного алгоритму ECDSA, агенти можуть захистити свою інформацію під час спілкування.

BlockChainUtil. Цей клас містить поширені методи, які необхідно використовувати в блокчейн-продуктах. Ці методи в основному використовуються для роботи з шифруванням та дешифруванням на основі криптографічних алгоритмів.

Клас Communication використовується як контейнер повідомлень між комунікаційними агентами. Оскільки комунікаційні агенти є адміністраторами системи, їхня комунікація має вирішальний вплив на продуктивність системи. Тому важливо забезпечити цілісність та безпеку інформації, яка передається між комунікаційними агентами. Об'єкти комунікації містять хеш, дані, хеш попереднього зв'язку, підпис privateKey агента відправника, publickKey відправника та publicKey агента одержувача. Аргумент хеш – це ідентифікатор цього комунікаційного об'єкта. Аргумент даних містить інформацію, яку агент відправника хоче надіслати агенту одержувачу. Аргумент prevCmHash – це хеш-значення попередніх комунікаційних агентів, яке використовується для відстеження порядку відповідних комунікаційних об'єктів. Аргумент підпису створюється на основі privateKey агента відправника та може бути перевірений за допомогою publicKey агента відправника. Він використовується, щоб переконатися, що інформація надіслана самим відправником. Аргументи відправника та одержувача – це publicKey агента відправника та агента одержувача.

Блокчейн – це ланцюжок, що складається з різних блоків. Блок містить хеш попереднього блоку, власне хеш-значення, час генерації, MerkleRoot та максимальну висоту блоку. MerkleRoot обчислюється рекурсивним способом. Перший рівень дерева обчислюється відповідно до хешу комунікацій у блоці, а кожен наступний рівень дерева обчислюється на основі попереднього рівня дерева, доки не буде обчислено корінь дерева. MerkleRoot – це ефективний спосіб відстеження цільового зв'язку та перевірки його цілісності. Попередній хеш використовується для ідентифікації позиції цього блоку, тоді як хеш використовується як ідентифікаційне значення. Як тільки розмір списку досягне MAX_BLOCK_HEIGHT, будуть обчислені MerkleRoot та хеш, і блок буде додано до блокчейну. Оновлений блокчейн потім буде збережено на локальному хості та передано іншим модулям. Приклад блоку, згенерованого за допомогою блокчейну, показано на наступній сторінці.

Клас контракту містить всю бізнес-логіку процесу комунікації. Визначено дозволи та перевірки. Вони пов'язані з певними комунікаційними повідомленнями, які комунікаційний агент може надсилати іншим конкретним комунікаційним агентам, і визначають, коли ця комунікаційна дія може бути дійсною. Комунікаційні агенти можуть запускати функцію комунікації лише за допомогою цього смарт-контракту. Оскільки цей контракт регулює лише комунікаційних агентів, інші агенти, які мають нові функції та перебувають під контролем комунікаційних агентів, все ще можуть бути додані до системи або видалені з неї. Контракт використовується лише для забезпечення зв'язку між комунікаційними агентами в межах рівня управління, щоб підтримувати як безпеку, так і масштабованість. Коли запускається модуль виявлення та аналізу, блокчейн буде ініціалізовано. Генезис блоків та зв'язок будуть збережені в блокчейні, а блокчейн буде збережено на локальному хості та трансльовано іншим модулям. Причина, чому цей блокчейн ініціалізовано модулем виявлення та аналізу, полягає в тому, що цей модуль є основним модулем системи. Тип зв'язку, який можна надсилати та зберігати, визначається контрактом. Смарт-контракти можуть редагуватися лише системними менеджерами перед ініціалізацією. Їх не можна змінити після ініціалізації системи.

Кожен комунікаційний агент має AgentAccount, який зберігатиме ім'я агента, його публічний ключ (publicKey) та приватний ключ (privateKey). Ці ключі шифруються криптографічним алгоритмом ECDSA, щоб гарантувати, що лише правильний агент може генерувати та отримувати доступ до певних комунікаційних повідомлень. Облікові записи агентів генеруються під час ініціалізації агентів у системі. Коли комунікаційним агентам потрібно надіслати комунікаційне повідомлення (як команду, так і зворотний зв'язок), ці комунікаційні повідомлення можуть оброблятися лише методами, визначеними в смарт-контракті. Отримувачі можуть перевірити повідомлення лише після перевірки цілісності даних за допомогою відкритого ключа відправника, даних та підпису. Смарт-контракти також

вирішують, яке повідомлення можуть надсилати або отримувати які комунікаційні агенти. Кожен комунікаційний агент містить інформацію про prevCmHash, хеш, підпис, дані, відправника та одержувача. Підпис обчислюється за допомогою закритого ключа відправника та необроблених даних. Кожен блок містить хеш попереднього блоку, власне хеш-значення, час генерації, merkleRoot, максимальну висоту блоку та комунікації. Комунікації містять деталі кожного комунікаційного повідомлення, виданого комунікаційними агентами. MerkleRoot обчислюється рекурсивним способом. Перший рівень дерева обчислюється відповідно до хешу комунікацій у блоці, а потім кожен наступний рівень дерева обчислюється на основі попереднього рівня дерева, доки не буде обчислено корінь дерева. MerkleRoot – це ефективний спосіб відстеження цільового зв'язку та перевірки його цілісності. Попередній хеш використовується для ідентифікації позиції цього блоку, тоді як хеш, згенерований Sha256, використовується як ідентифікаційне значення. Як тільки розмір списку комунікацій досягне MAX_BLOCK_HEIGHT, будуть обчислені MerkleRoot та хеш, а блок буде додано до блокчейну. Оновлений блокчейн потім буде збережено локально на хості та трансльовано іншим модулям.

Система інтегрує вбудовану базу даних та вузол блокчейну в одному агенті, запускаючи та зупиняючи роботу одночасно. Взаємодії будуть додані до кешу, що скоротить час пошуку та пришвидшить використання та зберігання даних в агентах.

3.2.2 Модуль виявлення та аналізу

Цей модуль відповідає за виявлення та аналіз шкідливого трафіку/шаблонів. Нижче описано класи Java, які ми використовували для реалізації цього модуля.

Агент DACommunication – це комунікаційний агент модуля виявлення та аналізу. Цей процес пояснюється на рисунку 3.2. Після того, як модуль

обробки даних надсилає повідомлення про необхідність виявлення нового пакета даних, агент DACommunication надсилає команду агенту виявлення. Коли агент виявлення завершить виявлення, агент DACommunication отримає звіт про виявлення від агента виявлення. Звіт про виявлення буде інкапсульовано в комунікаційний об'єкт і надіслано агенту RCommunication. Якщо агент DACommunication отримає комунікаційний об'єкт, який повідомляє про оновлення навчального набору, агент DACommunication надішле команду навчальному агенту. Після того, як навчальний агент завершить навчання моделі, звіт про навчання буде надіслано агенту DACommunication, а також агенту RCommunication.

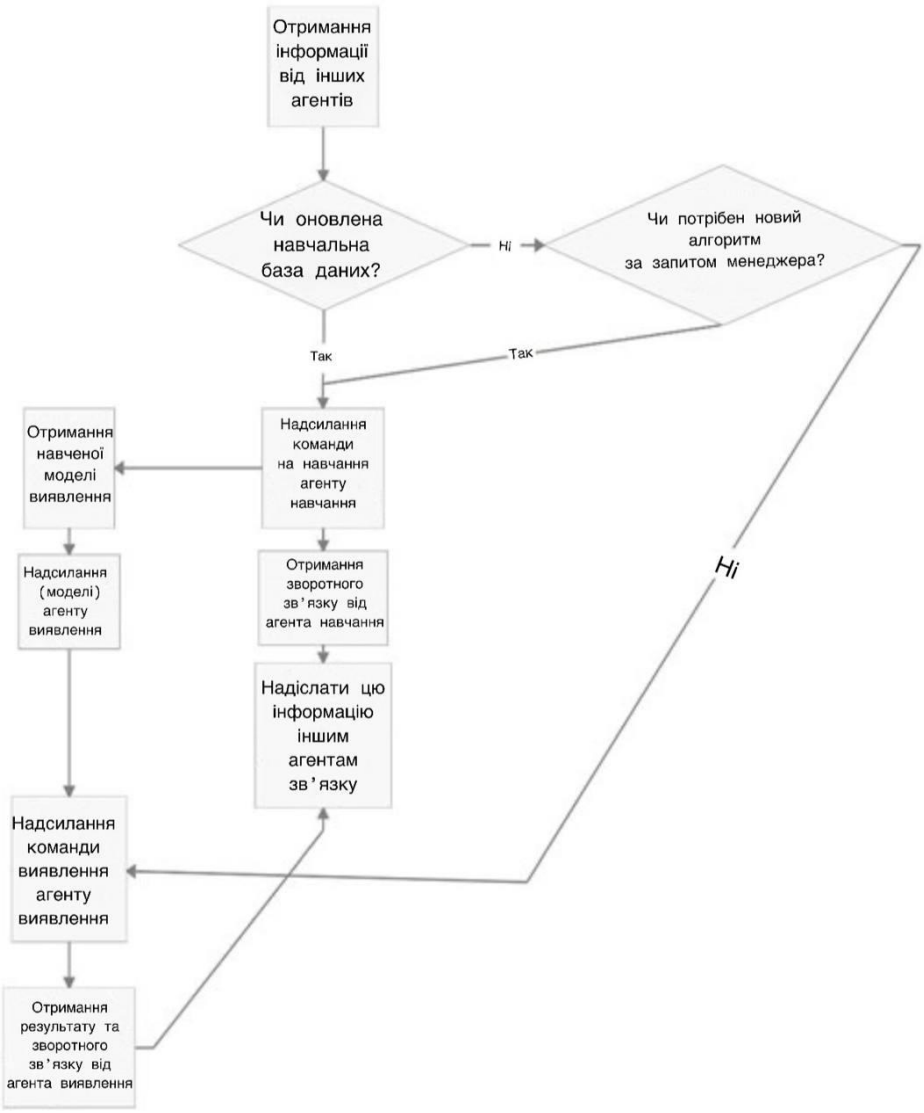

Рисунок 3.2 – Процес виявлення та аналізу.

Detection Agent. Коли агент виявлення отримує команди від агента DACommunication, він починає використовувати модель DNN для аналізу нового пакета даних та виявлення атак. Після завершення процесу буде згенеровано звіт про виявлення, який буде надіслано агенту DACommunication.

Training Agent. Коли навчальний агент отримує команди від агента DACommunication, він розпочинає навчання моделі DNN, використовуючи оновлений набір навчальних даних. Після завершення процесу буде згенеровано звіт про навчання, який буде надіслано агенту DACommunication.

DNNUtil. Цей клас містить методи, що стосуються управління моделлю DNN. Новий необроблений набір даних буде стандартизовано та нормалізовано за допомогою методів нормалізації. Трансформовану структуру можна переглянути та скоригувати за допомогою функції transformProcess та файлу.

3.2.3 Модуль реагування

Агент RCommunication – це комунікаційний агент модуля відповіді. Щойно агент RCommunication отримає повідомлення Communication, яке містить звіт про виявлення або звіт про навчання, він надішле команду агенту відповіді. Ці звіти будуть збережені на локальному хості.

Response Agent. Після того, як агент відповіді отримає команду від агента RCommunication, він почне генерувати діаграми візуалізації даних для користувачів на основі плану реагування адміністратора та вносити зміни до брандмауера та безпеки хоста на основі команд агента RCommunication.

Візуалізація даних. Класи BarChartUtil та LineChartUtil використовуються для керування візуалізацією даних. Параметри, такі як розмір діаграм або тема діаграм, можна налаштувати.

Послідовність дій системи виявлення вторгнень наступна.

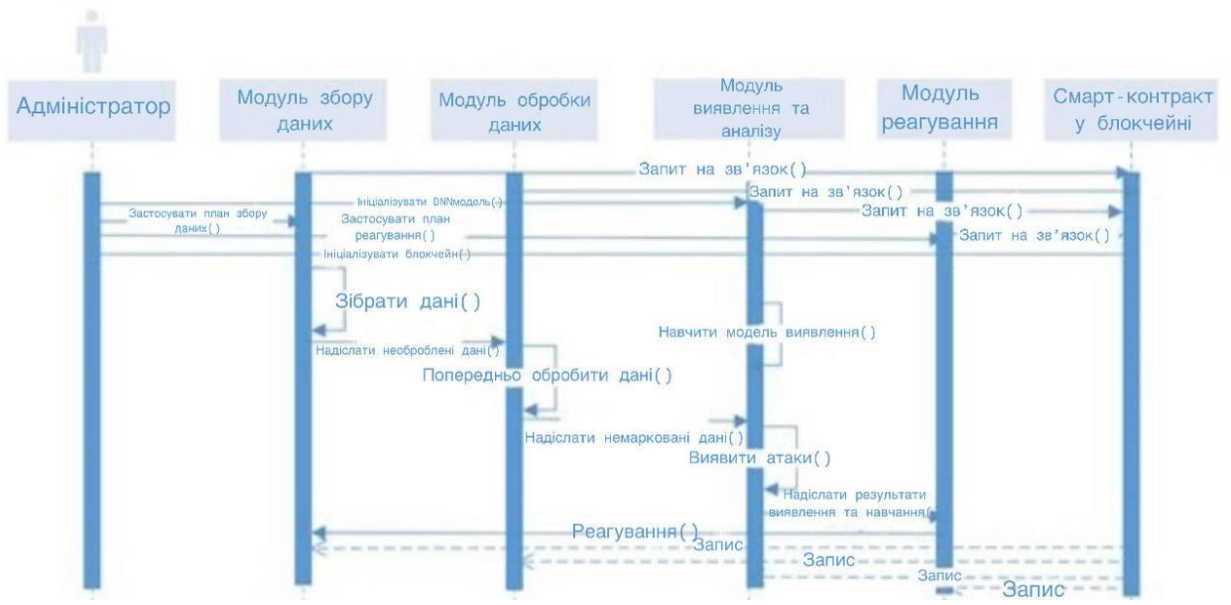

Рисунок 3.3 – Діаграма послідовності системи

3.2.4 Модуль обробки даних

На рисунку 3.4 відображено основну цього модуля, що є обробка даних для мережі, хоста та інших агентів.

Агент керування мережевими даними та агент керування даними хоста обробляють дані, що надходять відповідно з мережі та хоста. Обидва агенти повинні виконувати попередню обробку даних, включаючи обробку відсутніх значень, інтеграцію даних та стандартизацію даних. Після етапу попередньої обробки дані будуть скановані на основі правил виявлення вторгнень, згенерованих іншими спільнотами відкритого коду IDS та/або правил системного менеджера. Крім того, попередньо оброблені дані будуть упаковані та стиснуті. Виявлені поведінкові процеси вторгнення та відповідні заходи обробки записуються в пакет виявлення вторгнень. Після порівняння цього з результатом агента виявлення, пакет виявлення вторгнень надсилається агенту відповіді. Позначені дані об'єднуються в пакет навчального набору, який використовується для підвищення точності агента виявлення.

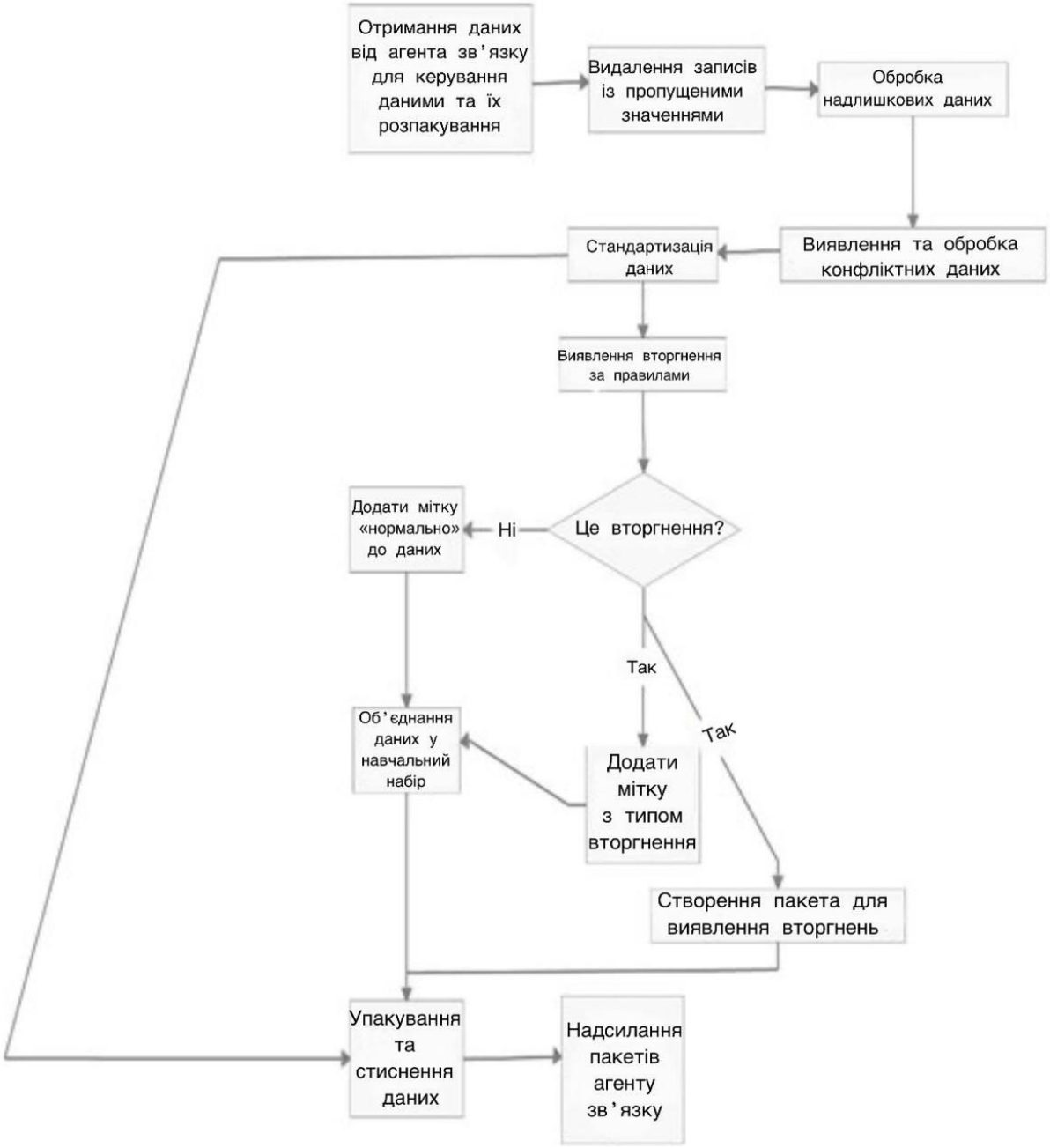

Рисунок 3.4 – Процес роботи агентів управління даними.

Агент бази даних – єдиний агент, який може змінювати базу даних, на рисунку 3.5. Він може змінювати базу даних лише за наявності команди від комунікаційного агента. Після того, як агент бази даних отримає команду оновлення від комунікаційного агента, він змінює базу даних на основі даних, наданих комунікаційним агентом. Після того, як агент бази даних змінив базу даних, він надсилає комунікаційному агенту повідомлення зворотного зв'язку.

Рисунок 3.5 – Процес агента бази даних

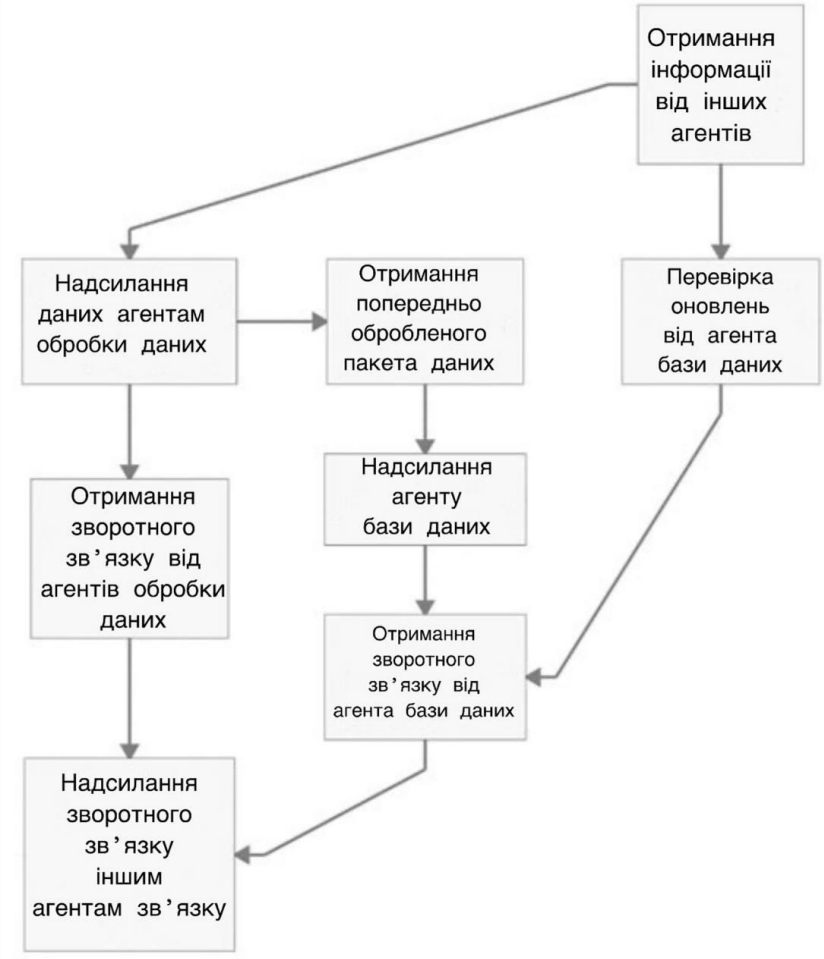

Рисунок 3.6 – Комунікаційний агент для процесу обробки даних

Комунікаційний агент для процесу обробки даних керує та контролює хост-агент обробки даних, мережевий агент обробки даних та агент бази даних, на рисунку 3.6. Він надсилає пакети даних з модуля збору агентам обробки даних, надає команди агенту обробки даних та отримує їхній зворотний зв'язок. Він запитує агент бази даних про регулярні перевірки оновлень, забезпечуючи актуальність системи. Цей агент також взаємодіє з іншими комунікаційними агентами та коригує свій робочий режим відповідно до стану системи.

4 ЕКСПЕРЕМЕНТАЛЬНІ ДОСЛДЖЕННЯ ТА ТЕСТУВАННЯ

4.1 Набір даних

Набір даних аудиту мережевої безпеки KDDCUP99, опублікований лабораторією виявлення вторгнень Колумбійського університету, був проаналізований та складений на основі набору даних IDS лабораторії Лінкольна Массачусетського технологічного інституту, але він містив лише дані мережевого трафіку.

Набір даних NSL-KDD вирішує деякі з властивих наборів даних KDD99. Набір даних NSL-KDD широко використовується в розробці систем виявлення вторгнень. Хоча цей набір даних все ще має деякі недоліки, його можна використовувати як ефективний набір даних для порівняння різних методів виявлення вторгнень. Розмір навчального та тестового наборів NSL-KDD є прийнятним, а результати оцінки різних дослідників будуть узгодженими та порівнянними. Дані набору даних NSL-KDD надходять з трьох різних протоколів (протокол керування передачею (TCP), протокол користувацьких дейтаграм (UDP) та протокол керування повідомленнями Інтернету (ICMP). Набір даних NSL-KDD містить чотири класи атак (DoS, Probe, R2L, U2R) та 39 різних типів атак. У цьому дослідженні набір даних NSL-KDD використовується як джерело даних для моделювання виявлення вторгнень.

Хоча дані NSL-KDD не є специфічними для тестування середовища Інтернету речей, протоколи та атаки, що використовуються в цьому наборі даних, є цінними для досліджень виявлення вторгнень у середовище Інтернету речей. Типи атак у цьому наборі даних також трапляються в реальних мережах Інтернету речей, і протоколи, такі як TCP, стають дедалі важливішими, оскільки як традиційний рівень додатків, так і з'єднання хмарної платформи є важливою частиною середовищ Інтернету речей. Тому

набір даних NSL-KDD використовується як набір даних для Інтернету речей. Зразки з набору даних NSL-KDD позначені як нормальні або аномальні. 70% підмножини з назвою «Набір даних Train + 20%» використовується як навчальний набір даних, тоді як 30% набору даних «Train + 20%» використовується як набір даних для валідації. Підмножина «Набір даних Test+», що містить 1200 зразків даних, використовується як тестовий набір даних в таблицях 4.1-4.2.

Таблиця 4.1 – Типи атак (Навчання +20%)

Основні типи атак	Типи атак
DOS	Back, Land, Neptune, Pod, smurf, teardrop
Probe	Satan, IPsweep, Nmap, portsweep
R2L	ftp_write, phf, multihop, warezmaster, guess_passwd, warezclient, imap, spy
U2R	buffer_overflow, loadmodule, rootkit

Таблиця 4.2 – Типи атак на піднабір даних Test+

Основні типи атак	Типи атак
DOS	Back, Land, Neptune, Pod, smurf, teardrop, mailbomb, processtable, udpstorm, apache2
Probe	Satan, IPsweep, Nmap, portsweep, mscan, saint
R2L	Warezmaster, xsnoop, snmpguess, snmpgetattack, httptunnel, sendmail, named, guess_passwd
U2R	buffer_overflow, rootkit

4.2 Заходи оцінювання

Продуктивність виявлення вторгнень буде вимірюватися на основі точності, прецизійності, повноти та оцінки F1. Точність – це відношення

кількості зразків у парі прогнозування до загальної кількості зразків. Точність базується на результатах прогнозування. Вона показує, скільки зразків позитивного прогнозування є істинно позитивними зразками. Повнота розраховується для оригінального зразка, що показує, скільки позитивних прикладів у вибірці передбачено правильно. Оцінка F1 всебічно враховує результати розрахунку коефіцієнта точності та коефіцієнта повноти моделі. Чим більша оцінка F1, тим краща якість моделі. Також необхідно враховувати перенавчання.

4.3 Попередня обробка

Усі необроблені дані для навчальних, валідаційних та тестових наборів даних стандартизовані за допомогою одноразового кодування та вимірювань z-оцінки, а нормалізація виконується з використанням однієї й тієї ж схеми. Для цього експериментального дизайну одноразове кодування використовує категоріальні змінні з усіх навчальних, валідаційних та тестових наборів даних, замість використання категоріальних змінних на основі окремих наборів даних. Одна категоріальна змінна в кожному параметрі, яку потрібно закодувати за допомогою одноразового кодування, видаляється, щоб запобігти потраплянню фіктивних змінних.

Останнє поле «class» – це вихідна мітка. Змінні, такі як protocol_type, service та flag, містять кілька значень, тому їх потрібно перевести в числові значення за допомогою одного гарячого кодування. Щоб уникнути пастки фіктивних змінних, одне категоріальне значення protocol_type, service та flag видаляється. Поле «class» також кодується в числові значення.

Для визначення продуктивності модуля виявлення та аналізу було порівняно продуктивність у різних ситуаціях, результати в таблицях 4.3-4.5.

Таблиця 4.3 – Точність виявлення вторгнень за різної кількості епох

Розмір пакету	Номер партії	Епоха	Точність на наборі валідації	Точність на тестовому наборі
32	300	50	98,46%	80,35%
32	300	100	98,51%	82,91%
32	300	150	98,47%	81,91%
32	300	200	98,52%	82,43%
32	300	250	98,44%	82,24%
32	300	300	98,45%	82,57%

Таблиця 4.4 – Точність виявлення вторгнень для різних номерів партій.

Розмір пакету	Номер партії	Епоха	Точність на наборі валідації	Точність на тестовому наборі
32	50	100	97,36%	77,58%
32	100	100	97,63%	78,83%
32	150	100	98,07%	80,25%
32	200	100	98,19%	81,00%
32	250	100	98,34%	80,17%
32	300	100	98,51%	82,92%
32	350	100	98,43%	80,67%
32	400	100	98,47%	80,92%

Тестування модуля виявлення та навчання з різними розмірами партій, номерами партій та епохами показало, що модель DNN має кращу продуктивність для розміру партії 48, номера партії 300 та кількості епох 100, ніж для інших протестованих значень. Коефіцієнт точності з використанням валідаційного набору становить 98,27%, тоді як точність тестового набору становить 83,17%. Точність на тестовому наборі становить 83,53%. Повчальність на тестовому наборі становить 84,14%, а показник F1 на тестовому наборі становить 83,94%. Це означає, що модель DNN у цій

системі має хорошу здатність відрізняти атаки від звичайного трафіку даних. Ці експерименти також показують, що вибір достатньої кількості епох може забезпечити хорошу продуктивність моделі, навіть якщо навчальний набір має менше вибірок.

Таблиця 4.5 – Коефіцієнт точності виявлення вторгнень різного розміру пакетів.

Розмір пакету	Номер партії	Епоха	Точність на наборі валідації	Точність на тестовому наборі
16	300	100	98,09%	81,25%
32	300	100	98,51%	82,92%
48	300	100	98,27%	83,17%
64	300	100	98,54%	80,42%
80	300	100	98,67%	79,83%

В таблиці 4.6 показано продуктивність вторгнень на основі різних наборів даних. Вони позначені наступним чином: точність на валідаційному наборі як AV, прецизійність на валідаційному наборі як PV, повнота на валідаційному наборі як RV, бал F1 на валідаційному наборі як FV, точність на тестовому наборі як AT, прецизійність на тестовому наборі як PT, повнота на тестовому наборі як RT та бал F1 на тестовому наборі як FT: В таблиці 4.7 підсумовано точність виявлення за використання різних умов розділення даних. Модель потенційно може бути оновлена для більш складних наборів даних.

Таблиця 4.6 – Продуктивність виявлення вторгнень з використанням різних навчальних наборів

Навчальний набір	AV	PV	RV	FV	AT	PT	RT	FT
Train+	98.27%	98.27%	98.24%	98.12%	83.17%	83.53%	84.14%	83.94%
Test+	97.48%	97.69%	97.22%	97.81%	91.50%	91.53%	91.77%	91.21%

Таблиця 4.7 – Коефіцієнт точності виявлення вторгнень за допомогою різних умов розділення даних

Умови розділення даних	Точність на валідаційному наборі AV%	Точність на тестовому наборі AT%
- 70% від навчання + 20% як навчальний набір - 30% від поїзда + 20% як набір для перевірки - підмножина набору даних Test+, що містить 1200 зразків даних як тестовий набір	98,27	83,17
- 70% Test+ як навчальний набір - 30% Test+ як набір для перевірки - підмножина набору даних Train + Test, що містить 1200 зразків.	97,48	91,50

В роботі досліджувались параметри даних, отриманих з NSL-KDD, разом з їхніми серійними номерами. Загалом використано 41 параметр. В таблиці 4.8 показано продуктивність модуля виявлення та аналізу за різною кількістю параметрів. Оскільки система виявлення вторгнень SESS розроблена для використання в мережах Інтернету речей різного масштабу, а

деякі параметри важко зібрати в деяких мережах Інтернету речей, важливо знати, як різна кількість параметрів впливає на продуктивність системи.

Таблиця 4.8 – Продуктивність виявлення вторгнень за допомогою різних параметрів.

Номер параметра	Вкл. параметри	AV	PV	RV	FV	AT	PT	RT	FT
41	[1–41]	98.27	98.27	98.24	98.12	83.17	83.53	84.14	83.94
40	[1–2], [4–41]	98.85	98.84	98.85	98.76	82.33	84.04	84.00	82.30
25	[1–2], [4–6], [12], [23–41]	97.82	97.84	97.77	97.63	80.75	82.56	82.45	80.64
17	[1–2], [4–6], [12], [23–24], [29], [32–39]	97.88	97.92	97.81	97.69	81.33	83.13	83.03	81.24
7	[1–2], [4–6], [23–24]	95.10	95.21	94.96	94.62	78.08	78.19	78.75	79.37
6	[1–2], [5–6], [23–24]	93.53	93.87	93.25	92.75	72.67	79.45	68.74	80.15
5	[2], [5–6], [23–24]	92.67	93.16	92.32	91.71	72.33	79.23	68.35	79.95
4	[2], [5–6], [23]	90.40	91.45	89.84	88.80	73.92	75.68	71.31	79.71
3	[2], [5–6]	58.15	69.92	54.95	20.87	71.75	78.83	67.67	79.62
2	[5–6]	84.27	86.25	85.16	85.12	68.50	71.95	64.68	76.89
1	[6]	53.72	53.72	50.00	0.00	43.00	43.00	50.00	0.00

У таблиці 4.8 використовується 48 розмірів пакетів, 300 номерів пакетів та 100 епох. Загальна кількість параметрів набору даних NSL-KDD становить 41.

Першим параметром, який видаляється, є сервіс, оскільки тип сервісу відрізняється для різних мереж Інтернету речей, що має великий вплив на процес нормалізації. Якщо виключити лише параметр сервісу,

використовуючи 40 параметрів, модель DNN має точність 82,33% на тестовому наборі, що на 0,84% менше, ніж при використанні повної кількості з 41 параметра.

Коли набори даних використовують 25 параметрів, модель DNN має точність 80,75% на тестовому наборі, а точність зростає до 81,33, якщо використовувати лише 17 параметрів.

Причина, чому точність для 17 параметрів вища, ніж за використання 25 параметрів, полягає в тому, що деякі параметри необхідно поєднувати з іншими параметрами для виявлення атак. Коли кількість параметрів зменшується з 40 до 25, деякі параметри втрачають свої комбіновані параметри, що негативно впливає на процес зважування. Після видалення цих комбінованих параметрів точність зростає.

Коли використовуються лише сім параметрів, а саме duration, protocol_type, flag, src_bytes, dst_bytes, count та srv_count, модель DNN має точність 78,08%. Це означає, що модель DNN все ще має пристойну здатність до виявлення, навіть якщо модуль збору не може отримати багато деталей з мережі Інтернету речей. Коли як параметри використовуються лише duration, protocol_type, src_bytes, dst_bytes, count та srv_count, модель DNN має точність 72,68% та повноту 68,74% на тестовому наборі. У цьому випадку видалення параметра flag знизило коефіцієнт точності. Це показує важливість параметра flag. Однак параметр flag також ускладнює процес нормалізації. Щоб дослідити продуктивність цієї системи в мережі Інтернету речей, яка не надає багато деталей, параметр flag слід видалити, оскільки його може бути важко отримати за певних обставин.

Якщо набори даних використовують лише п'ять параметрів, модель DNN має точність 72,33% на тестовому наборі. Якщо набори даних використовують protocol_type, src_bytes, dst_bytes та count як параметри, модель DNN має точність 73,92% на тестовому наборі. Це означає, що використання лише чотирьох параметрів призводить до кращої продуктивності, ніж використання п'яти або шести параметрів.

Якщо набори даних використовують три параметри, модель DNN має точність 71,75% на тестовому наборі, але вона має точність лише 58,15% на валідаційному наборі. Це пояснюється тим, що модель DNN використовує метод виключення (Dropout). Цей метод регуляризації випадковим чином видалятиме деякі слабкі класифікатори під час процесу навчання. У цій ситуації використання методу виключення вплине на точність при застосуванні до валідаційного набору. Збільшення розміру пакета або видалення методу виключення може вирішити цю проблему, хоча це, у свою чергу, може призвести до проблеми перенавчання. Якщо метод виключення видаляється з моделі DNN, модель DNN має точність 89,23% на валідаційному наборі та 70% на тестовому наборі. Якщо набори даних використовують src_bytes та dst_bytes лише як параметри, модель DNN має точність 68,5% на тестовому наборі.

Якщо використовувати dst_bytes як єдиний параметр, коефіцієнт точності становить 43% для тестового набору даних, але показник F1 як для валідаційного, так і для тестового набору даних дорівнює 0%. Це означає, що всі зразки в наборах даних класифікуються як один клас, що є проблемою перенавчання. Це пояснюється тим, що деякі гіперпараметри, такі як кількість епох, є занадто високими для наборів даних з одним параметром.

На рисунку 4.1 можна побачити, що ця модель добре працює на наборах даних TCP та ICMP (коефіцієнт точності становить 99,1% та 98,1% відповідно). Однак коефіцієнт точності для протоколу UDP становить лише 93,7%. Однією з причин є те, що набір даних містить дуже мало даних з протоколами UDP. Особливості протоколу UDP є ще однією причиною. Для набору даних, який використовує клас для різних типів атак як значення, DNN має хорошу продуктивність (коефіцієнт точності становить 97%). Однак деякі типи атак, які трапляються лише у відносно невеликій кількості даних, не можуть бути добре розрізнені.

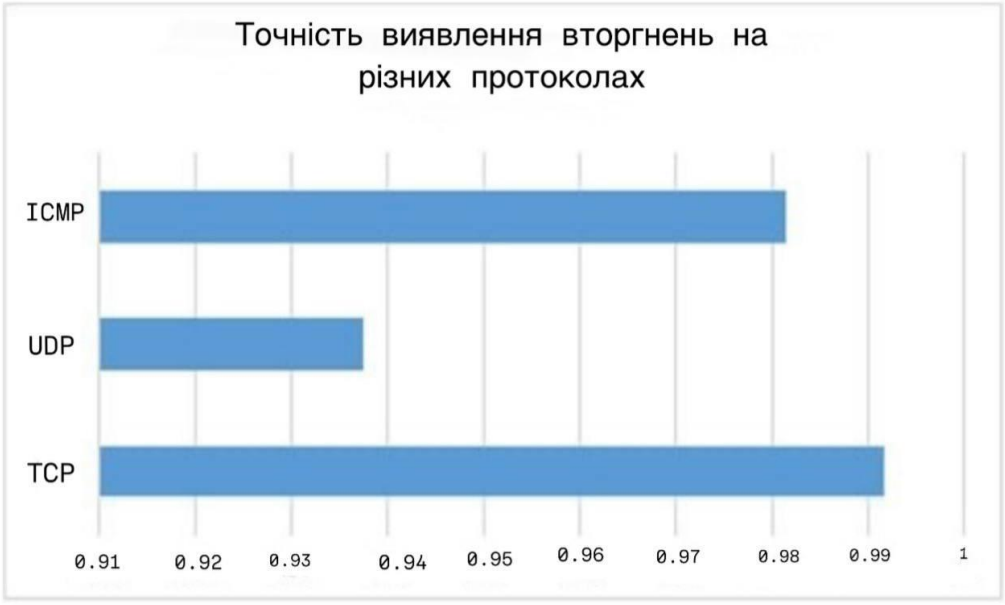

Рисунок 4.1 – Точність за різними протоколами.

На завершення, ці експерименти показують можливість використання цієї системи на різних масштабах мережі Інтернету речей. Під час експериментів час виявлення становив близько 0,18 с, що означає, що система може виявляти атаки та реагувати на них у режимі реального часу. Час записувався на основі різних етапів, включаючи ініціалізацію та завантаження моделі даних, час навчання та час виявлення. Цей час може змінюватися залежно від різних обставин, включаючи апаратне забезпечення, інші ресурси, що використовують сервер, потреби в пам'яті тощо. Фактично, ці вимірювання потребують подальшого дослідження та дослідження, щоб переконатися, що запропонована модель може працювати у високошвидкісному реальному середовищі. Результати цих експериментів також будуть використані для визначення статусів процесу багатоагентного навчання з підкріпленням.

ВИСНОВКИ

У рамках кваліфікаційної роботи розглянуто та запропоновано систему виявлення вторгнень на основі багатоагентної системи, блокчейну та глибокого навчання. Детально описали режим роботи кожного компонента моделі та роботу всієї системи. Гнучкість багатоагентних систем означає, що цю нову IDS можна застосовувати в середовищах IoT різного розміру. Усі дії, спричинені комунікаційними агентами, будуть записані на блокчейні, що робить систему захищеною від загроз, включаючи підробку інформації, розголошення інформації тощо. Використання багатоагентного алгоритму посилення може допомогти системі постійно покращувати її продуктивність.

Вивчається застосування нейронних мереж. Результати моделювання показують, що алгоритм глибокого навчання має кращу продуктивність, ніж традиційні методи в мережі IoT такого ж типу. Впровадження технології блокчейн гарантує, що цю систему можна поширювати на різні віддалені хости, оскільки зв’язок ACL захищений блокчейном, а зв’язок між агентами регулюється смарт-контрактом. Експерименти з використанням набору даних NSL-KDD демонструють високу точність виявлення вторгнень на транспортному рівні середовища IoT для DNN. Ефективність моделі DNN у розрізненні аномалії від норми краща, ніж продуктивність інших методів машинного навчання, таких як дерева рішень. Це демонструє потенціал використання алгоритмів глибокого навчання для IDS пристроїв IoT. Експерименти демонструють високу продуктивність системи в різних сценаріях, таких як мережі різної складності та різні типи атак.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Adat, V.; Gupta, B. Security in Internet of Things: Issues, challenges, taxonomy, and architecture. Model. Anal. Des. Manag. 2018, 67, 423–441.

2. Statista Research Department. IoT: Number of connected devices worldwide 2012–2025.

3. Tahaei, H.; Afifi, F.; Asemi, A.; Zaki, F.; Anuar, N.B. The rise of traffic classification in IoT networks: A survey. J. Netw. Comput. Appl. 2020, 154.

4. О.С. Ляшенко, І.А. Великодний, В.Г. Знайдюк, О.Д. Журило. Модель та методи виявлення широкомасштабної атаки в середовищі IoT / Системи управління, навігації та зв'язку. Том 1 № 75 (2024), С.127-132

5. Hajiheidari, S.; Wakil, K.; Badri, M.; Navimipour, N.J. Intrusion detection systems in the Internet of things: A comprehensive investigation. Comput. Netw. 2019, 160, 165–191.

6. Журило, О. і Ляшенко, О. (2024) «Архітектура та системи безпеки IoT на основі туманних обчислень», Сучасний стан наукових досліджень та технологій в промисловості, (1(27), с. 54–66. doi: 10.30837/ITSSI.2024.27.054.

7. Bhattarai, S.; Wang, Y. End-to-End Trust and Security for Internet of Things Applications. Computer 2018, 51, 20–27.

8. Spafford, E.; James, P. Anderson: An Information Security Pioneer. IEEE Secur. Priv. 2008, 6, 9.

9. Alnaghes, M.S.; Gebali, F. A Survey on Some Currently Existing Intrusion Detection Systems for Mobile Ad Hoc Networks. In Proceedings of the Second International Conference on Electrical and Electronics Engineering, Clean Energy and Green Computing (EEECEGC2015), Antalya, Turkey, 26–28 May 2015; Volume 12.

10. Hari, P.B.; Singh, S.N. Security Attacks at MAC and Network Layer in Wireless Sensor Networks. J. Adv. Res. Dyn. Control Syst. 2019, 11, 82–89.

11. Pacheco, J.; Hariri, S. IoT Security Framework for Smart Cyber Infrastructures. In Proceedings of the IEEE 1st International Workshops on Foundations and Applications of Self* Systems (FAS*W), Augsburg, Germany, 12–16 September 2016; pp. 242–247.

12. Liu, C.; Yang, J.; Chen, R.; Zhang, Y.; Zeng, J. Research on immunity-based intrusion detection technology for the Internet of Things. In Proceedings of the 2011 Seventh International Conference on Natural, Shanghai, China, 26–28 July 2011.

13. Roman, R.; Zhou, J.; Lopez, J. On the features and challenges of security and privacy in distributed internet of things. Comput. Netw. 2013, 57, 2266–2279.

14. Wallgren, L.; Raza, S.; Voigt, T. Routing Attacks and Countermeasures in the RPL-Based Internet of Things. Int. J. Distrib. Sens. Netw. 2013, 9, 794326. [Google Scholar]

15. Raza, S.; Wallgren, L.; Voigt, T. SVELTE: Real-time intrusion detection in the Internet of Things. Ad Hoc Netw. 2013, 11, 2661–2674.

16. Zegzhda, P.; Kort, S. Host-Based Intrusion Detection System: Model and Design Features. In Proceedings of the International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, St. Petersburg, Russia, 13–15 September 2007; pp. 340–345.

17. Chakravarthi, S.S.; Veluru, S. A Review on Intrusion Detection Techniques and Intrusion Detection Systems in MANETs. In Proceedings of the International Conference on Computational Intelligence and Communication Networks, Bhopal, India, 14–16 November 2014.

18. Zhurylo, O., Liashenko, O., & Avetisova, K. (2023). Hardware security overview of Fog computing end devices in the Internet of Things. Innovative technologies and scientific solutions for industries, (1 (23), 57–71. https://doi.org/10.30837/ITSSI.2023.23.057

19. Mohamed, A.B.; Idris, N.B.; Shanmugum, B. A Brief Introduction to Intrusion Detection System. In Trends in Intelligent Robotics, Automation, and Manufacturing, Proceedings of the IRAM 2012, Kuala Lumpur, Malaysia, 28–30

November 2012; Communications in Computer and Information Science; Ponnambalam, S.G., Parkkinen, J., Ramanathan, K.C., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 330.

20. Zarpelão, B.B.; Miani, R.S.; Kawakani, C.T.; Alvarenga, S.C. A survey of intrusion detection in Internet of Things. J. Netw. Comput. Appl. 2017, 84, 25–37.

21. Bostani, H.; Sheikhan, M. Hybrid of anomaly-based and specification-based IDS for Internet of Things using unsupervised OPF based on MapReduce approach. Comput. Commun. 2017, 98, 52–71.

22. Fu, Y.; Yan, Z.; Cao, J.; Koné, O.; Cao, X. An Automata Based Intrusion Detection Method for Internet of Things. Mob. Inf. Syst. 2017, 2017, 1750637.

23. Kapitonov, A.; Lonshakov, S.; Krupenkin, A.; Berman, I. Blockchain-based protocol of autonomous business activity for multi-agent systems consisting of UAVs. In Proceedings of the Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), Linkoping, Sweden, 3–5 October 2017; pp. 84–89.

24. Calvaresi, D.; Calbimonte, J.P.; Dubovitskaya, A.; Mattioli, V.; Piguet, J.G.; Schumacher, M. The Good, the Bad, and the Ethical Implications of Bridging Blockchain and Multi-Agent Systems. Information 2019, 10, 363.

25. Le, T.-T.-H.; Kim, Y.; Kim, H. Network Intrusion Detection Based on Novel Feature Selection Model and Various Recurrent Neural Networks. Appl. Sci. 2019, 9, 1392.

26. Arshad, J.; Azad, M.A.; Abdeltaif, M.M.; Salah, K. An intrusion detection framework for energy constrained IoT devices. Mech. Syst. Signal Process. 2020, 136, 106436.

27. Anthi, E.; Williams, L.; Slowinska, M.; Theodorakopoulos, G.; Burnap, P. A Supervised Intrusion Detection System for Smart Home IoT Devices. IEEE Internet Things J. 2019, 6, 9042–9053.

28. Chaabouni, N.; Mosbah, M.; Zemmari, A.; Sauvignac, C.; Faruki, P. Network Intrusion Detection for IoT Security Based on Learning

Techniques. IEEE Commun. Surv. Tutor. 2019, 21, 2671–2701.

29. Liang, C.; Shanmugam, B.; Azam, S.; Jonkman, M.; Boer, F.D.; Narayansamy, G. Intrusion Detection System for Internet of Things based on a Machine Learning approach. In Proceedings of the International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN), Vellore, India, 30–31 March 2019; pp. 1–6.

30. Savaglio, C.; Fortino, G.; Ganzha, M.; Paprzycki, M.; Badica, C.; Ivanovic, M. Agent-based Internet of Things: State-of-the-art and research challenges. Future Gener. Comput. Syst. 2019, 102.

31. Pipattanasomporn, M.; Feroze, H.; Rahman, S. Multi-agent systems in a distributed smart grid: Design and implementation. In Proceedings of the IEEE/PES Power Systems Conference & Exposition, Seattle, WA, USA, 15–18 March 2009.

32. Fortino, G.; Russo, W.; Savaglio, C. Agent-oriented modeling and simulation of IoT networks. In Proceedings of the Federated Conference on Computer Science and Information Systems (FedCSIS), Gdansk, Poland, 11–14 September 2016; pp. 1449–1452.

33. Wang, S.; Wan, J.; Zhang, D.; Li, D.; Zhang, C. Towards smart factory for industry 4.0: A self-organized multi-agent system with big data based feedback and coordination. Comput. Netw. 2016, 101, 158–168.

34. Bellifemine, F.L.; Caire, G.; Greenwood, D. Developing Multi-Agent Systems with JADE; Wiley: Hoboken, NJ, USA, 2007.

35. Ляшенко О.С., Журило О.Д, Дубихвіст В.В. Модель мультиагентної системи виявлення вторгнень // Вісник ВПІ, вип. 3.