Хмельницький національний університет
Факультет інформаційних технологій
Кафедра кібербезпеки

КВАЛІФІКАЦІЙНА РОБОТА

Козарєзова Олександра Андріївна

на здобуття ступеня вищої освіти Бакалавра

Система виявлення аномалій в IoT за допомогою Honeypots

Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека
Освітня програма Кібербезпека

Шифр КРБКБ.2102149.21.02.12 ПЗ

Виконала студентка 4 курсу група КБ-21-2 Олександра КОЗАРЄЗОВА
Керівник докт. тех. наук, професор Михайло КАСЯНЧУК
Нормоконтролер старший викладач Сергій МОСТОВИЙ

До захисту допускаю:
Завідувач кафедри кібербезпеки Юрій КЛЬОЦ

16 06 2025 р.

Хмельницький 2025

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Кібербезпеки

Рівень вищої освіти Бакалавр

Галузь знань 12 – Інформаційні технології

Спеціальність 125 – Кібербезпека

Освітня програма Кібербезпека

ЗАТВЕРДЖУЮ

Завідувач кафедри кібербезпеки

Юрій КЛЬОЦ

15 лютого 2025 р.

З А В Д А Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Козарєзова Олександра Андріївна

1 Тема роботи Система виявлення аномалій в IoT за допомогою Honeypots

Керівник роботи Михайло Касянчук

Затверджено наказом ректора університету від 7 лютого 2025 № 23

2 Строк подання студентом кваліфікаційної роботи на кафедру 10.06.2025

3 Вихідні дані до роботи Розробити ефективну систему виявлення аномальної активності в середовищі IoT-пристроїв з використанням технології Honeypots. Дослідити предметну область, що стосується загроз інформаційній безпеці в IoT, зокрема методів атак та вторгнень. Проаналізувати існуючі підходи до виявлення аномалій та роботу пасток Honeypot у контексті інтернету речей.

4 Зміст пояснювальної записки (перелік питань, які потрібно розробити)
У пояснювальній записці буде розглянуто аналіз предметної області та характеристику загроз для IoT-пристроїв, зокрема типові вектори атак та уразливості, що використовуються зловмисниками. Далі буде сформульовано постановку задачі та обґрунтовано вибір технології Honeypot як ефективного інструменту для виявлення аномальної активності в середовищі інтернету речей. Наступним етапом стане проєктування архітектури системи виявлення аномалій, що включає компоненти збору, обробки та аналізу даних.

5 Перелік графічного матеріалу (із зазначенням обов'язкових креслень) «Автоенкодер з лінійною функцією активації», «Сценарій атаки», «Схема структури модулів»

6 Консультанти розділів кваліфікаційної роботи

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

7 Дата видачі завдання 19 лютого 2025 р.

КАЛЕНДАРНИЙ ПЛАН

Назва етапів (розділів) кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
Вибір і затвердження теми кваліфікаційної роботи	Лютий	
Аналіз загроз та вразливостей IoT-пристроїв	Лютий	
Дослідження існуючих підходів до виявлення аномалій та Honeypot-систем	Лютий	
Постановка задачі та визначення вимог до системи виявлення аномалій	Березень	
Визначення загальних принципів побудови системи з Honeypot-компонентом	Березень	
Розробка архітектури системи виявлення аномалій з Honeypot	Квітень	
Реалізація механізмів збору даних, виявлення аномалій і логування	Квітень	
Тестування системи на виявлення аномалій у симульованому IoT-середовищі	Травень	
Оформлення пояснювальної записки згідно вимог	Травень	
Оформлення графічної частини	Червень	
Захист КР	Червень	

Студентка Олександра КОЗАРЄЗОВА

Керівник кваліфікаційної роботи Михайло КАСЯНЧУК

АНОТАЦІЯ

Тема кваліфікаційної роботи: Система виявлення аномалій в IoT за допомогою Honeypots.

Автор роботи: Козарєзова Олександра Андріївна.

Керівник роботи: Касянчук Михайло Миколайович.

Пояснювальна записка:59 с., 2 додатки, 24 рисунків, 41 джерел.

Графічна частина: 3 плакати.

HONEYPOTS, IoT, СИСТЕМА ВИЯВЛЕННЯ АНОМАЛІЙ, МЕРЕЖЕВА БЕЗПЕКА, АНАЛІЗ ТРАФІКУ, ВРАЗЛИВОСТІ IoT, КІБЕРЗАГРОЗИ.

Кваліфікаційна робота бакалавра присвячена розробці системи виявлення аномальної активності у мережах Інтернету речей IoT із використанням технології Honeypots. У роботі проаналізовано особливості безпеки IoT-пристроїв, типи атак та методи їх виявлення.

Розглянуто класифікацію Honeypots, їх архітектуру, способи інтеграції з IoT-середовищем та механізми збору й аналізу даних. Запропоновано модель виявлення аномалій, засновану на аналізі мережевого трафіку, що дозволяє ідентифікувати потенційні загрози в режимі реального часу.

У результаті розроблено систему, що включає технічне завдання, політику безпеки, модель загроз, план заходів, схему розгортання Honeypots та прототип засобу виявлення аномалій. Виконано підготовку до впровадження системи в інфраструктуру IoT з метою підвищення її стійкості до кібератак.

07.06.2025 _____

ABSTRACT

Qualification work topic: Anomaly detection system in IoT using Honeypots.

Author: Kozarezova Oleksandra Andriyivna.

Head of work: Kasyanchuk Mykhailo Mykolayovych.

Explanatory note:59 p., 2 appendices,24 figures, 41 sources

Graphic part: 3 posters.

HONEYPOTS, IoT, ANOMALY DETECTION SYSTEM, NETWORK SECURITY, TRAFFIC ANALYSIS, IoT VULNERABILITIES, CYBERTHREATS.

The bachelor's thesis is devoted to the development of a system for detecting anomalous activity in IoT networks using Honeypots technology. The paper analyses the security features of IoT devices, types of attacks, and methods for detecting them.

The article considers the classification of Honeypots, their architecture, methods of integration with the IoT environment, and mechanisms for data collection and analysis. An anomaly detection model based on the analysis of network traffic is proposed, which allows identifying potential threats in real time.

As a result, a system was developed that includes a technical specification, security policy, threat model, action plan, Honeypots deployment scheme, and a prototype anomaly detection tool. Preparations have been made to implement the system in the IoT infrastructure to increase its resilience to cyberattacks.

07.06.2025

ЗМІСТ

ВСТУП

Прогрес у сфері Інтернету речей IoT та розвиток підключення стали дедалі помітнішими за останнє десятиліття, і, за прогнозами, це зростання триватиме. Ця технологія знайшла широке застосування завдяки доступності недорогих сенсорних рішень, мініатюризації пристроїв та повсюдному поширенню інтернету. На сьогодні IoT впроваджено майже в усі сфери, що призвело до розвитку таких концепцій, як розумний транспорт, розумні міста, розумна медицина, розумне сільське господарство, розумні будинки, розумні університети тощо. Значення цього явища полягає в швидкій взаємодії та ефективній взаємодії між людьми, об'єктами та інфраструктурами.

Водночас важливо розуміти, що пристрій, який знаходиться вдома, за рівнем безпеки та вразливості практично не відрізняється від тих, що використовуються великими корпораціями.

Попри позитивні аспекти стрімкого розвитку IoT, слід також відзначити, що його різноманітність породжує серйозні виклики в сфері безпеки. Основною проблемою є зростання кількості загроз і атак у міру того, як технологія стає дедалі доступнішою для широкого кола користувачів і компаній. Пристрої IoT неодноразово ставали основою для ботнет-атак, у яких зловмисники захоплювали величезну кількість вразливих пристроїв для здійснення шкідливих дій, таких як атаки типу DDoS. Більшість підключених IoT-систем є більш вразливими до атак через великий обсяг даних, які генеруються і передаються між пристроями та користувачами. Це зумовлено тим, що як ландшафт загроз, так і методи атак стають дедалі складнішими.

Тому постійно виникає потреба в анонімному виявленні підозрілої та аномальної активності в середовищі IoT на основі даних, які генеруються цими пристроями наприклад, шляхом вивчення шаблонів атак та мотивів зловмисників за допомогою методів обману.

1 АНАЛІЗ АНОМАЛІЙ ТА МЕТОДІВ ЇХ ВИЯВЛЕННЯ

1.1 Аномалії та їх типи

У контексті Інтернету речей (IoT) аномалія — це будь-які спостереження чи набори даних, які суттєво відрізняються від очікуваних або типових характеристик роботи пристрою чи системи. Це можуть бути як поодинокі, так і регулярні події, які виникають несподівано або в нетиповий час і не відповідають заданим шаблонам поведінки системи.

Аномалії можуть бути спричинені різноманітними зовнішніми чинниками, такими як збої датчиків, некоректна робота програмного забезпечення, спроби несанкціонованого доступу чи цілеспрямовані кібератаки. Основною задачею алгоритмів виявлення аномалій є своєчасна ідентифікація таких подій та, за можливості, визначення їх першопричини. На рисунку 1.1 зображено приклад аномалії.

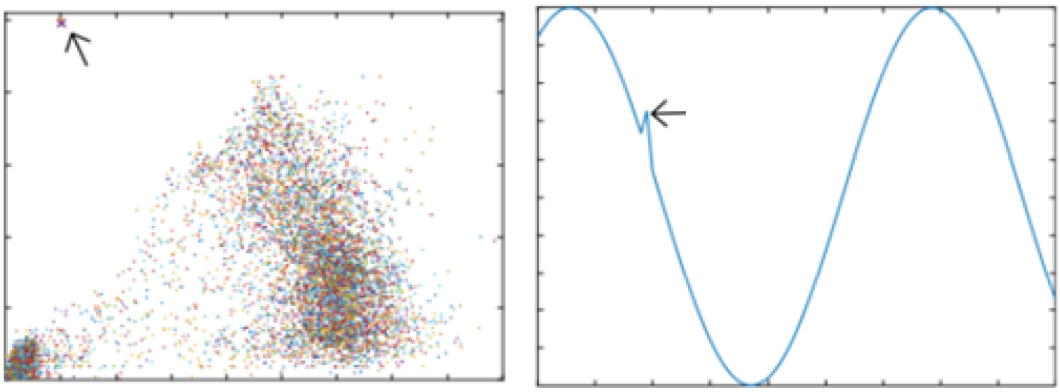


Рисунок 1.1 – Приклад аномалії

Залежно від способів вирішення проблеми, практичного застосування, типу методології та швидкості реагування, алгоритми виявлення аномалій умовно поділяються на кілька категорій. Враховуючи специфіку додатків IoT та розмаїття типів даних, необхідно застосовувати різноманітні методи і підходи до аналізу. Наприклад, алгоритми, які ефективно виявляють аномалії у показниках

датчиків, можуть бути менш придатними для аналізу мережевого трафіку, і навпаки. Таким чином, критично важливим є правильний вибір методології відповідно до конкретної ситуації.

Для забезпечення високого рівня безпеки та надійності систем IoT необхідно чітко розуміти різні типи аномалій, що можуть виникати у процесі їх експлуатації. Один з основних типів — це тимчасові аномалії, які відзначаються коротким періодом прояву та несподіваністю виникнення. Незважаючи на короткочасність, їх вплив на систему може бути значним. Наприклад, яскравим проявом тимчасової аномалії є атака типу Distributed Denial of Service (DDoS), коли значна кількість IoT-пристроїв одночасно направляє потік запитів на один сервер, що призводить до його перевантаження і відмови в обслуговуванні легітимних користувачів. На рисунку 1.2 зображено приклад DDoS-атаки.

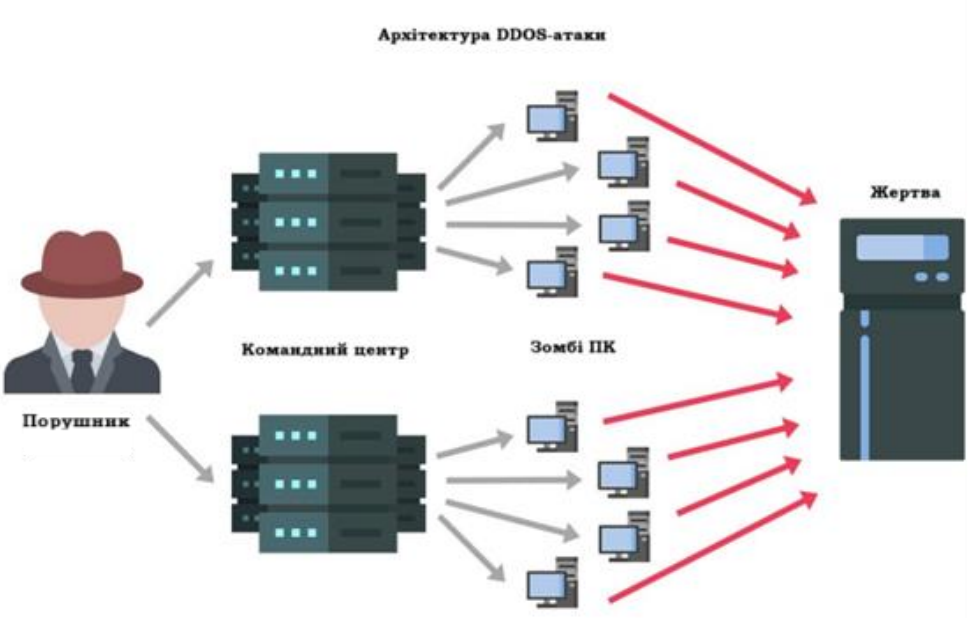


Рисунок 1.2 - Приклад DDoS-атаки

Такі ситуації часто трапляються через недостатню захищеність пристроїв IoT та масове їх зараження ботнетами [1]. Також до цього типу належать короткочасні різкі сплески використання ресурсів, наприклад, процесора чи оперативної пам'яті, що виникають через програмні помилки або

непередбачувані події в системі.

Для виявлення таких відхилень особливо ефективні алгоритми машинного навчання, які здатні швидко ідентифікувати та повідомляти про аномалії завдяки аналізу великих обсягів даних та формуванню базових моделей нормальної поведінки [2].

Важливим типом є постійні або хронічні аномалії, які характеризуються тривалістю існування та стійкістю. Цей тип аномалій свідчить про серйозні системні збої, неправильно налаштоване обладнання чи інші тривалі відхилення, які можуть залишатися непоміченими протягом тривалого часу.

Постійно відкритий мережевий порт на IoT-пристрої може використовуватися зловмисниками для тривалого несанкціонованого доступу, що становить серйозну загрозу безпеці системи [3]. До цього типу також відносяться ситуації, коли пристрої постійно демонструють підвищений рівень споживання ресурсів, що може свідчити про наявність шкідливого програмного забезпечення або фізичну несправність обладнання [4].

Виявлення таких аномалій часто потребує системного підходу та використання спеціалізованих методів кіберрозвідки, а також інтеграції honeypot-систем, що дозволяють відслідковувати підозрілу активність та збирати інформацію про потенційних зловмисників [5].

Останнім важливим типом є контекстуальні аномалії, визначення яких значною мірою залежить від ситуації або контексту їх виникнення. Тобто одні й ті самі події можуть вважатися нормальними або аномальними залежно від контексту, в якому вони відбуваються. Розуміння цих типів аномалій є ключовим для розробки ефективних систем виявлення та реагування на загрози. Аналізуючи контекст, тривалість та характер аномалій, можна більш точно визначити їх причини та вжити відповідних заходів для забезпечення безпеки та надійності системи.

Активне використання мережі є абсолютно нормальним явищем в робочий час, коли більшість пристроїв та користувачів системи активно обмінюються

інформацією, але такий самий рівень активності вночі або у вихідні може бути ознакою стороннього втручання чи несанкціонованого доступу [6].

Так само температурні показники датчика можуть бути абсолютно нормальними влітку, але ті самі значення можуть свідчити про несправність або злом у зимовий період. Для ефективного виявлення таких аномалій особливо важливо враховувати контекст, що досягається за допомогою спеціалізованих алгоритмів, здатних аналізувати поведінку пристроїв залежно від часу доби, пори року чи місця розташування [7]. Тут активно використовуються системи honeypot, які імітують реальні пристрої та дозволяють виявити спроби несанкціонованого доступу та атаки в конкретних контекстах, що значно покращує ефективність захисту IoT-систем [8].

Варто також зазначити, що для створення ефективних механізмів виявлення аномалій необхідно поєднувати різні методології та підходи. Комплексний підхід дозволяє не лише швидко виявляти підозрілі події, а й точно визначати їхню природу і джерела виникнення. Завдяки застосуванню машинного навчання, евристичних правил, статистичного аналізу та інтеграції honeypot-систем, забезпечується високий рівень точності та ефективності виявлення аномалій у системах IoT. Застосування саме такого комбінованого підходу дозволяє суттєво підвищити рівень захисту систем, адже він здатний виявляти як прості, очевидні відхилення, так і складні, приховані атаки.

Додатковим аспектом, який слід враховувати, є необхідність постійного оновлення і адаптації алгоритмів виявлення аномалій через постійний розвиток нових загроз та ускладнення атак. В умовах, коли кіберзлочинці швидко адаптуються до нових методів захисту, важливо регулярно проводити аналіз та оновлювати бази знань алгоритмів виявлення, додавати нові патерни поведінки та інтегрувати нові методи машинного навчання, здатні виявляти невідомі раніше типи атак або нетипові збої.

Розуміння типів аномалій, здатність правильно вибирати і поєднувати різні підходи, а також регулярне оновлення алгоритмів та методів, є ключовими

факторами для успішного функціонування систем виявлення аномалій у сфері Інтернету речей. Такий підхід дозволяє суттєво знизити ризики і негативні наслідки атак, забезпечити безперервну роботу IoT-пристроїв та підтримувати високий рівень інформаційної безпеки загалом.

1.2 Методи виявлення аномалій та їх застосування в системах IoT

Методи виявлення аномалій та їх застосування в системах IoT є важливими для забезпечення високого рівня безпеки та стабільності цих систем. Зростаюча кількість пристроїв IoT створює додаткові ризики виникнення різноманітних атак та збоїв, тому питання своєчасного виявлення аномалій стає актуальним.

Серед основних методів, які активно застосовуються для виявлення аномалій, є методи машинного навчання. Ці методи використовують алгоритми, що здатні самостійно навчатися на великих масивах даних та виявляти складні закономірності, які можуть свідчити про аномальну поведінку. На рисунку 1.3 зображено найпростіший автоенкодер з лінійною функцією активації.

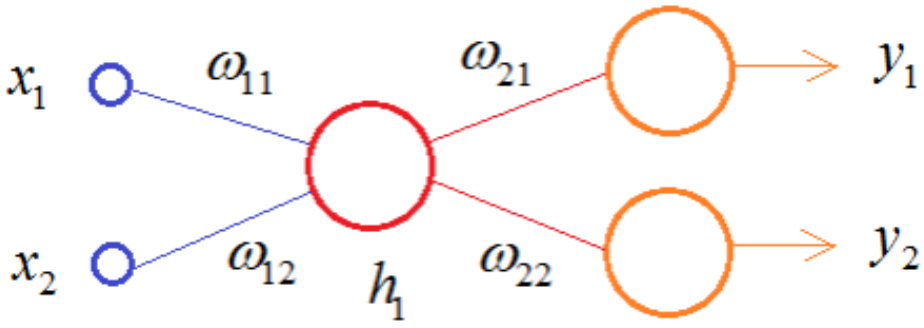


Рисунок 1.3 - Найпростіший автоенкодер з лінійною функцією активації

Ефективними є алгоритми кластеризації K-Means чи DBSCAN, які дозволяють групувати дані за подібністю та виявляти об'єкти, що значно відхиляються від загальних закономірностей. Це може бути корисним, наприклад, для виявлення нетипового мережевого трафіку, що свідчить про спробу атаки або несанкціонованого доступу.

Перспективними є нейронні мережі, зокрема автоенкодери, які здатні виявляти більш тонкі та складні аномалії, навчавшись на нормальних наборах даних та ідентифікуючи відхилення у реальному часі. В сучасних розробках нейронні мережі часто є базовими для стратегій створення honeypot-систем нового покоління, які ефективно відловлюють різноманітні типи аномальної активності та атак [9]. Ще однією групою методів є евристичні, засновані на певних правилах, що визначають нормальну або аномальну поведінку системи. Найпростішим прикладом є правила, засновані на порогових значеннях якщо кількість звернень пристрою до мережі перевищує заданий ліміт, то така поведінка може бути класифікована як аномальна.

Цей метод має перевагу простоти та швидкої реалізації, однак може бути неефективним у випадку складних атак, що потребує використання додаткових методів чи комбінації різних підходів для точнішого виявлення загроз [10-12]. Важливу роль у виявленні аномалій відіграють також статистичні методи, які дозволяють аналізувати поведінку пристроїв у часовому контексті. За допомогою аналізу часових рядів можна виявляти неочікувані зміни у поведінці, такі як раптові сплески або спадання трафіку, що може свідчити про потенційні DDoS-атаки або зараження пристроїв ботнетом.

Інші статистичні підходи, зокрема використання стандартних відхилень чи Z-оцінок, дозволяють визначати ймовірність аномальності поточної поведінки пристроїв. Аналіз часових рядів: використовується для виявлення відхилень від нормальних моделей поведінки з плином часу. Аналіз часових рядів трафіку дає змогу виявити раптові сплески або спади, які вказують на аномалії.

Статистичні моделі використовуються для визначення ймовірності того,

що поточна поведінка є аномальною. Стандартні відхилення використовуються для виявлення трафіку, що виходить за межі норми. Раптове збільшення трафіку може вказувати на розподілену атаку типу «відмова в обслуговуванні» DDoS або зараження ботнету. Наприклад, камера безпеки раптово починає відправляти великі обсяги даних на невідомий сервер.

Несподівані підключення: підключення до невідомих серверів або IP-адрес можуть бути ознакою перехоплення даних або злому. Наприклад, датчик температури починає підключатися до сервера в іншій країні. Зміни в поведінці пристрою: наприклад, термостат, який раптово змінює температуру без втручання користувача, або детектор руху, який продовжує працювати без видимої причини [13-14].

1.3 Honeypots

В епоху зростаючої складності кіберзагроз, особливо в IoT, honeypots є важливим інструментом проактивного захисту. Honeypots це спеціалізовані системи або мережеві пристрої, які навмисно імітують вразливість системи. Мета такої імітації - привабити зловмисників, записати та проаналізувати їхню поведінку [15].

Інструменти виявлення та аналізу кібератак Honeypots класифікуються за рівнем взаємодії зі зловмисниками. Це дозволяє адаптувати інструмент для конкретних цілей безпеки.

Інструменти з низьким рівнем взаємодії імітують базові сервіси та протоколи для виявлення автоматизованих атак і сканування портів, їх легко розгортати. Вони є корисними для виявення ранніх стадій атак, але вони надають обмежену інформацію [16].

Honeypots може імітувати відкриті порти на таких серверах, як 22 SSH і 23 Telnet. Коли автоматизовані сканери атакують ці порти, honeypots реєструє IP-адресу зловмисника і час атаки. Це дозволяє виявляти великі мережеві

сканування, але не дає інформації про конкретну поведінку зловмисника. Іншим прикладом є Kippo, який імітує SSH-сервіс. Коли зловмисник намагається підключитися до нього, Kippo реєструє спроби введення логіна і пароля, але не надає реального доступу до системи. Високоінтерактивні honeypots вимагають більше ресурсів, але як повноцінні системи вони детально аналізують складні атаки і виявляють нові експлойти. Cowrie може імітувати SSH-сервер, але, на відміну від Kippo, надає зловмиснику доступ до віртуальної операційної системи. Це дозволяє зловмиснику записувати кожну введену команду і завантажений файл. Таким чином, можна детально проаналізувати поведінку зловмисника і виявити нові методи атаки [17].

Dionaea може імітувати вразливі сервіси, такі як SMB і HTTP, і виявляти експлойти, які використовуються для атак на ці сервіси. Коли зловмисник використовує експлойт, Dionaea записує його поведінку та надає інформацію про вразливість. Клієнтські honeypots імітують клієнтські програми та аналізують атаки на кінцеву точку користувача, щоб виявити шкідливе програмне забезпечення. Наприклад, такий клієнт може автоматично відвідувати веб-сайти і аналізувати їх на наявність шкідливого коду; якщо цей клієнт виявляє шкідливий код, він записує URL-адресу веб-сайту та інформацію про шкідливе програмне забезпечення.

Це дозволяє виявляти фішингові сайти та інші веб-загрози. Використання honeypots має переваги: збір детальної інформації про атаку, аналіз тактики і мотивації зловмисника [18].

Аналізуючи високоінтерактивні логи honeypot, можна визначити, які інструменти використовують зловмисники, які вразливості вони намагаються використати і які дані намагаються отримати [19]. З їх допомогою можна виявити нові методи атак, що дозволяє швидко реагувати на загрози. Honeypots можуть виявляти нові вразливості, які ще не були задокументовані. Це дозволяє розробникам програмного забезпечення швидко випускати патчі та захищати користувачів.

Також honeypots захищають мережеву інфраструктуру, відволікаючи увагу зловмисників від реальної системи та збираючи інформацію без шкоди для основної мережі. У мережі можуть бути розгорнуті honeypots, щоб відволікти увагу зловмисників від критично важливих серверів. Це виграє час для реагування на атаку і запобігання реальному пошкодженню системи.

Honeypots - це не просто інструмент, а стратегічний актив, який дозволяє організаціям без ризику відстежувати атаки в режимі реального часу та збирати інформацію про зловмисників для побудови проактивної системи безпеки та раннього попередження про нові загрози.

Зокрема, вони можуть виявляти невідомі вектори атак, аналізувати складні багатоетапні атаки та розуміти мотивацію зловмисників. У дослідженнях з кібербезпеки вони є ключем до створення контрольованого дослідницького середовища, в якому можна вивчати загрози, аналізувати шкідливе програмне забезпечення, виявляти експлойти, безпечно вивчати нові методи атак і розробляти заходи протидії.

Однак використання «медових горщиків» супроводжується етичними питаннями, пов'язаними з конфіденційністю і законністю, особливо при використанні honeypots для збору даних про зловмисників, що вимагає дотримання вимог і прозорості [20].

Необхідно забезпечити, щоб дані не використовувалися для незаконної діяльності і щоб конфіденційність користувачів була захищена. З розвитком кібербезпеки honeypots розвиваються з технологіями, з'являються IoT Honeypots, адаптовані до специфіки пристроїв Інтернету речей, використовуються ШІ та машинне навчання для автоматизованого аналізу великих обсягів даних, зібраних з Honeypots, та виявлення складних аномалій, інтегруються з іншими системами безпеки, такими як SIEM, IDS/IPS, для створення комплексних систем захисту та масштабуються в хмарі, що дозволяє швидко розгортати та керувати великими мережами Honeypots.

Існують обмеження та виклики: виявлення досвідченими зловмисниками,

які можуть використовувати анти-honeypot техніки, трудомісткий аналіз даних, що вимагає кваліфікованих фахівців, необхідність постійної підтримки та оновлення Honeypots для забезпечення їх актуальності та ефективності, а також юридичні ризики, пов'язані з використанням Honeypots для збору даних про зловмисників. Honeypots будуть відігравати все більш важливу роль в майбутніх системах кіберзахисту [21].

Honeypot з низьким рівнем взаємодії Low Interaction Honeypot, LIH зазвичай обмежений у своїй функціональності та взаємодії з атакуючим, надаючи лише мінімальну функціональність, таку як емуляція операційної системи та її служб.

Основні переваги honeypot'а з низьким рівнем взаємодії полягають у його простоті та тому, що він не вимагає багато ресурсів і легко обслуговується.Реалізація цього типу honeypot'а зазвичай потребує лише встановлення гіпервізора, здатного емулювати потрібну операційну систему, а також налаштування способу моніторингу системи під час її роботи без втручання.

Цей метод може легко обмежити або виявити будь-якого атакуючого, обмеживши його доступ до ресурсів, які надано honeypot'у, таким чином зменшуючи ризик подальшого проникнення, оскільки реальна операційна система ніколи не піддається впливу [22].

Основне призначення honeypot'а з низьким рівнем взаємодії зазвичай полягає у відстеженні походження атаки, а не її намірів або методів. На рисинку 1.4 наведено потік атаки LIH.

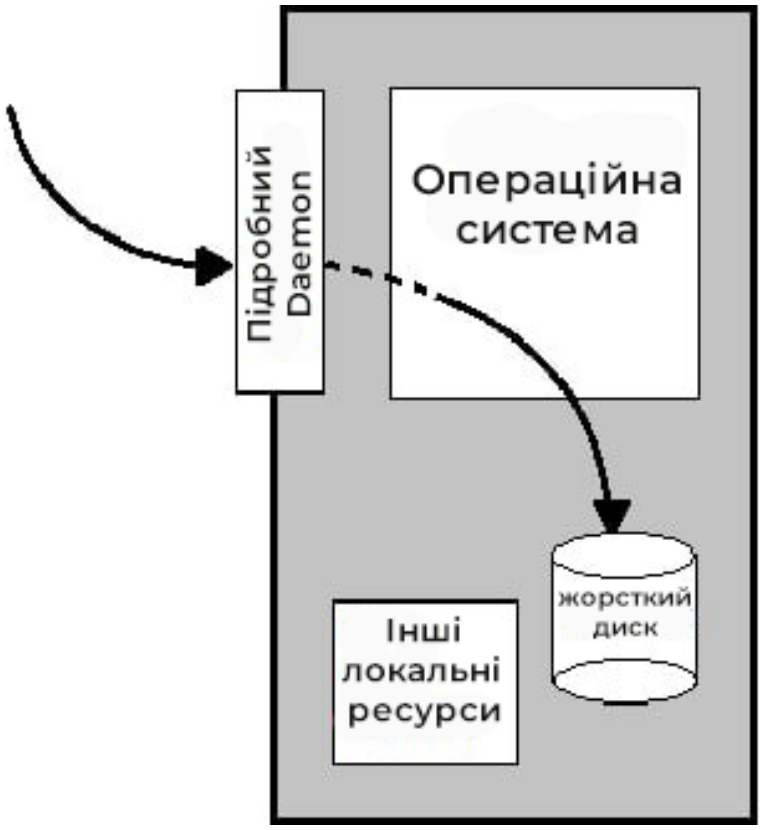


Рисунок 1.4 – Потік атаки LIH

Шлях атаки, доступний для зловмисника при націлюванні на honeypot з низьким рівнем взаємодії LIH. Він починається з маскування, яке honeypot створює для емуляції сервісу або системи, з яких зловмисник отримує доступ до базових компонентів дозволеної системи. У випадку LIH ресурси обмежені та зазвичай знаходяться поза досяжністю для атакуючого.

Прикладом LIH є HoneyRJ. Він реалізується на окремій системі, що займає IP-адресу, призначену виключно для honeypot'а, з метою виявлення походження атак на систему. HoneyRJ розроблено з урахуванням простоти та можливості розширення, він реєструє кожне з'єднання, вважаючи його потенційно зловмисним, та зберігає інформацію про з'єднання для подальшого аналізу.

High Interaction Honeypots, на відміну від honeypots з низьким рівнем взаємодії, є складнішими рішеннями, де зловмисник взаємодіє з реальними операційними системами, що працюють на фізичному обладнанні, яке зазвичай безпосередньо підключене до мережі поблизу серверів або інших критичних

систем. Таке рішення має дві основні переваги порівняно з LIH.

По-перше, діапазон моніторингу значно ширший, оскільки зловмисник взаємодіє з повноцінною реальною системою, що дозволяє зібрати більше даних і виявити шаблони поведінки. Це дає змогу досліджувати методи та тактики атакуючого, які він має намір використовувати для отримання подальшого доступу або виконання зловмисних дій [23].

По-друге, середовище honeypot не накладає жодних припущень щодо дій зловмисника, оскільки надає повну взаємодію з системою і не спонукає його до виконання певних дій, натомість пропонуючи широкий спектр сервісів і інформаційних вузлів.

Однак це підвищує ризик того, що зловмисник використає honeypot як платформу для атаки на решту мережі або інформаційні вузли. Тому необхідно вжити заходів безпеки для ізоляції honeypot від критичних систем, аби запобігти небажаному поширенню на системи, що не є частиною honeypot.

На рисунку 1.4 описуються маршрути, доступні для зловмисника при атаці на HIH. Тут зловмисник входить через імітований інтерфейс і отримує доступ до повного обсягу операційної системи та її ресурсів. Операційна система, у свою чергу, також має доступ до сусідніх машин у мережі й може вільно з ними взаємодіяти [24].

Прикладом HIH є HoneyBow. HoneyBow побудований за методом GenIII, що є найпоширенішим підходом до реалізації HIH. Honeynet третього покоління складається з шлюзу ізоляції Honeywall, який контролює honeypots.

Honeypot може бути розгорнутий як фізично, так і віртуально, та містить три основні компоненти, які використовуються для виявлення шкідливого програмного забезпечення MwWatcher, MwFetch та MwHunter. Разом вони формують розвинену систему збору й аналізу шкідливого ПЗ, яку можна інтегрувати до серверів.

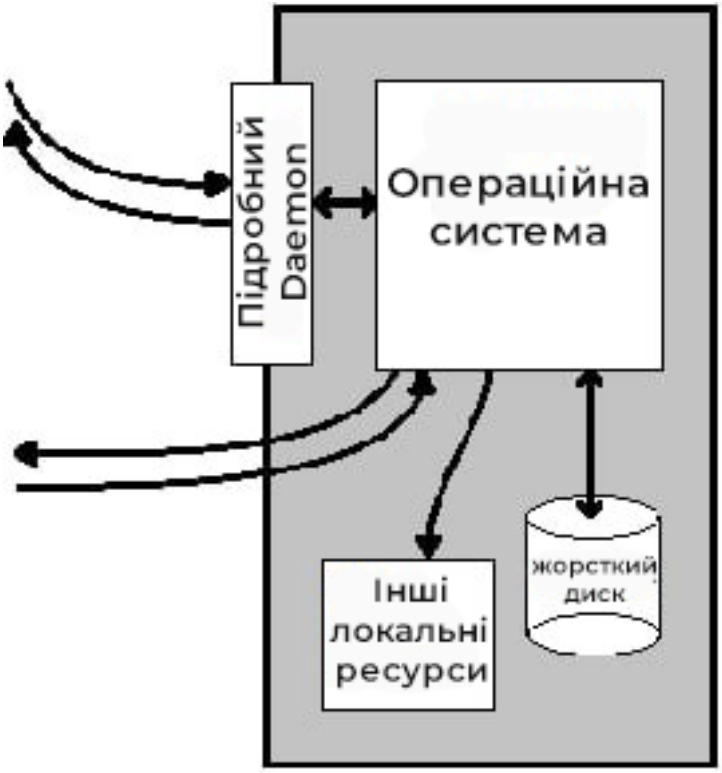


Рисунок 1.5 - Потік атаки HIH

Honeypot середньої взаємодії MIH є проміжним варіантом між двома іншими типами. Він використовується як компромісний варіант у випадках, коли необхідно проаналізувати дії зловмисника та отримати інформацію, але немає потреби в повноцінному середовищі з великою кількістю honeypots або ризику масштабних атак.

Цей тип надає більше можливостей для взаємодії зловмисника, ніж honeypot з низьким рівнем взаємодії, але менше функціональності, ніж honeypot з високим рівнем взаємодії. Honeypot середньої взаємодії часто використовується як спеціалізований і керований варіант, який можна налаштувати для виявлення конкретних атак або збору журналів взаємодії з певним вузлом у системі [25].

Більше того, цей тип намагається об'єднати кращі характеристики обох варіантів, пропонуючи високий рівень взаємодії, залишаючись при цьому легким у налаштуванні та масштабуванні. Honeypot, який використовується у цьому рішенні, належить до цієї категорії.

Honeypots можна також класифікувати за способом їх реалізації. У цій категорії існує два різні типи: промислові Honeypots та дослідницькі Honeypots [26].

Production Honeypots впроваджуються в організаціях для активного захисту в реальному операційному середовищі. Вони функціонують поруч із живою інфраструктурою і залишаються вразливими до атак 24/7. Ці типи Honeypots стають дедалі поширенішими в організаціях, оскільки забезпечують виявлення вторгнень і добре поєднуються з наявними заходами безпеки.

Research Honeypots, на відміну від них, не мають за мету захист живих систем. Вони забезпечують можливість вивчати шаблони атак і загрози, виконуючи переважно освітню та дослідницьку функцію. Honeypot, використаний у цьому експерименті та супровідному середовищі, класифікується як дослідницький Honeypot, оскільки головна мета полягає у вивченні та розумінні поведінки зловмисників в IoT-середовищі.

На завершення, honeypots надають практичний спосіб спостереження та збору даних щодо етапів атак та моделей поведінки зловмисників, які можуть бути проаналізовані для створення сигнатур і запобігання подальшим атакам. Додавання цього до IDS або ADS може покращити їхню функціональність завдяки можливості вивчення атак.

1.4 Аналіз існуючих рішень

У ході дослідження було ознайомлено з сучасними рішеннями, які реалізовані в галузі виявлення аномалій у системах Інтернету речей IoT, зокрема тими, що застосовують honeypot-технології в комбінації з методами машинного навчання. Результати аналізу засвідчили, що ця галузь динамічно розвивається, і наукова спільнота вже запропонувала низку високоефективних систем та концепцій, які заслуговують на детальний розгляд.

У статті автори пропонують метод глибокого навчання для виявлення аномалій у мережевому трафіку IoT [27]. Вони використовують автоенкодери, які навчаються на здоровому трафіку, а згодом виявляють відхилення тобто те, що не відповідає шаблону.

Автоенкодери тут виконують роль фільтра: вони стискають і реконструюють вхідні дані, й у разі, якщо помилка реконструкції перевищує певний поріг, трафік вважається підозрілим. Цей підхід має надзвичайно важливе значення, оскільки дозволяє працювати без необхідності маркувати великі обсяги навчальних даних, що в реальних умовах майже нездійсненне завдання [28-29].

У цій роботі представлена концепція системи виявлення вторгнень, яка базується на зборі даних з honeypot-серверів. Автори поєднують дані про активність зловмисників із штучним інтелектом, який дозволяє адаптивно реагувати на нові типи атак.

В дослідженні акцент зроблено на використанні honeypot як "живої" бази даних тобто інформація надходить із реальних спроб злому, а не зі штучно створених сценаріїв. Такий підхід, значно підвищує релевантність системи до умов реального середовища IoT, де атаки змінюються дуже швидко.

У роботі здійснюється порівняння кількох моделей глибокого навчання включаючи LSTM, GRU та CNN для виявлення аномалій у різних IoT-сценаріях [30]. Автори зосереджуються на проблемах обмежених ресурсів пристроїв та варіативності даних. Варто виділити висновок про ефективність LSTM у роботі з послідовностями даних. На практиці це означає, що для пристроїв, які генерують показники в реальному часі температурні датчики чи монітори руху, LSTM може надати точнішу модель аномального патерну. Цей підхід має потенціал бути не просто академічною концепцією, а базою для реального програмного модуля в honeypot-системі. Робота, в якій описано створення комплексної системи захисту на основі взаємодії між honeynet-інфраструктурою, SIEM-платформою тобто системою централізованого моніторингу, системою

класифікації атак за допомогою машинного навчання та спеціальною оболонкою для взаємодії з реальними IoT-пристроями [31-32]. Автори наводять приклад використання вразливого вебдодатку, інтегрованого з honeypot, який імітує роботу розумного домашнього пристрою. Це дослідження стало доказом того, що сучасні системи захисту вже переходять від концептуального рівня до впровадження повноцінних платформ, де кожен компонент відіграє чітко визначену роль. У межах іншого дослідження вивчено застосування ансамблевих методів для роботи з необробленими даними, зібраними з honeypot-систем. У цій роботі підкреслюється, що поєднання різних алгоритмів One-Class SVM, k-NN та Isolation Forest дозволяє зменшити кількість хибнопозитивних спрацювань, які є основною проблемою в таких системах. Працює підхід авторів до створення гібридної архітектури, яка не просто виявляє підозрілу активність, а й класифікує її залежно від ступеня ризику, часу виникнення та контексту подій . Окрім цього, було ознайомлено іще одним кейсом, представленим у дослідженні. У цьому дослідженні розроблено систему, яка розподіляє обробку даних між периферійними edge, туманними fog та хмарними cloud обчисленнями. Це дозволяє досягти високої швидкості реагування навіть за обмежених ресурсів на пристроях. Така гібридна модель дозволяє не лише виявляти аномалії ближче до джерела даних, а й масштабувати систему під великі промислові проєкти. Цей підхід одним із найперспективніших, оскільки він поєднує ефективність, масштабованість і гнучкість у використанні. Узагальнюючи, ефективність сучасних рішень значною мірою залежить від їхньої здатності адаптуватися до різних типів IoT-середовищ [33-34].

Вдале поєднання honeypot-систем, глибокого навчання, статистичних методів та мережевої розподіленої архітектури створює основу для систем виявлення аномалій нового покоління таких, які не лише фіксують відхилення, а й активно навчаються, аналізують причинно-наслідкові зв’язки та розпізнають раніше невідомі загрози.

1.5 Постановка задачі

Метою цієї дипломної роботи є створення ефективної системи виявлення аномалій у середовищі Інтернету речей IoT з використанням honeypot-технології та базових інструментів аналізу логів. В умовах стрімкого поширення IoT-пристроїв, які часто мають обмежені ресурси та вразливу конфігурацію, питання безпеки стає особливо критичним. Традиційні засоби захисту, зокрема сигнатурні методи, не завжди здатні виявити нетипову або нову поведінку, яка не має чітко визначених ознак атаки. Саме тому аномалії тобто будь-які відхилення від звичної моделі функціонування є ключовим індикатором можливих загроз. У межах дослідження розглядається створення симульованого IoT-середовища з honeypot, яке дозволяє фіксувати підозрілу активність. Зібрані дані з таких взаємодій аналізуються для виявлення аномальних сесій за кількістю введених команд, їх змістом, тривалістю сеансу або поведінковими патернами користувачів. Для цього розроблено прототип на основі Python, що дозволяє обробляти журнали взаємодій логи, виокремлювати ключові ознаки та класифікувати сесії як нормальні або аномальні. У якості honeypot використано платформу Cowrie, адаптовану для імітації IoT-пристроїв, доступ до яких здійснюється через Telnet та SSH. Розроблена система орієнтована не на точне виявлення конкретних типів атак, а на виявлення будь-яких нетипових сценаріїв, що можуть свідчити про небезпеку, зловживання або технічні збої. Це дозволяє гнучко реагувати на нові виклики в сфері кібербезпеки IoT, де часто використовуються невідомі або модифіковані методи проникнення.

У результаті дослідження очікується підтвердження ефективності обраного підходу для раннього виявлення потенційних загроз у віртуалізованому середовищі та можливість масштабування прототипу до складніших систем моніторингу й захисту

2 МОДЕЛЬ ВИЯВЛЕННЯ АНОМАЛІЙ В IOT

2.1 Модель виявлення аномалій

Нижче наведено опис моделі виявлення аномалій типу вторгнення на високому рівні, яка представлена на Рисунку 2.1. Модель IAD розділена на три окремі модулі, кожен із яких має власні компоненти: симульоване середовище, хост-середовище та поверхневе середовище.

Симульоване середовище містить симульовані IoT-пристрої та емульований IoT honeypot, замаскований під частину офісної мережі.

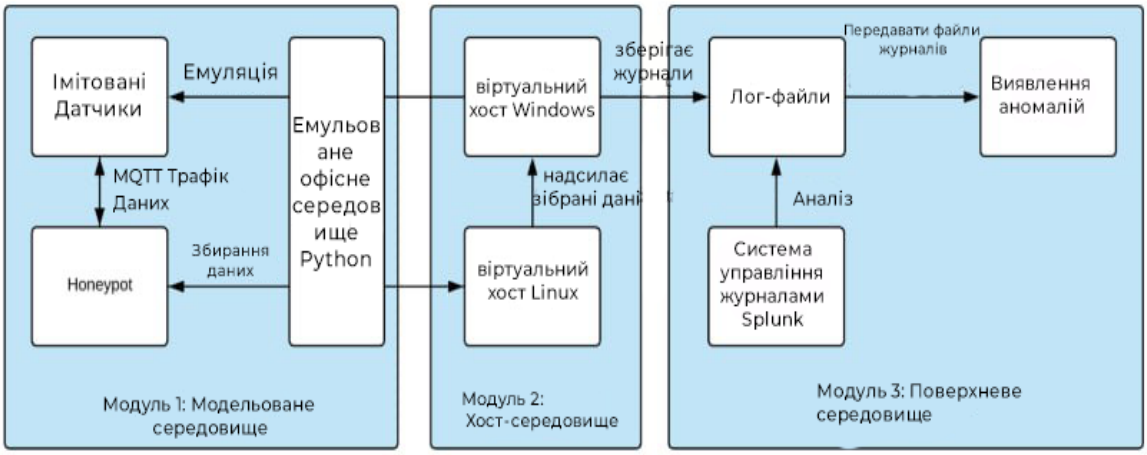


Рисунок 2.1 – Загальний опис системи та її модулів

Хост-середовище представляє два віртуальні хости, які розміщують симульоване середовище на тестовій VLAN мережі.

Нарешті, поверхневе середовище це місце, де відбувається захоплення лог-файлів, їх ізоляція для подальшого аналізу з метою виявлення можливих атак.

Модулі, позначені цифрами від 1 до 3, у своїй сукупності утворюють симульовану систему. Ця детальна модель складається з низки модулів, які разом, завдяки ефективній комунікації між собою, здатні виявляти потенційні вторгнення та шкідливу активність із різноманітних джерел. Після виявлення вторгнень відповідна активність фіксується та зберігається у вигляді журналу log-файлу, який потім аналізується з метою виявлення шаблонів атак для

подальшого розпізнавання загроз [35].

Важливо зазначити, що успішність цього підходу значною мірою залежить від ефективної взаємодії компонентів моделі. Системна модель була побудована таким чином, щоб імітувати офісне середовище, в якому управління інцидентами здійснюється більш проактивним способом.

2.2 Реалізація виявлення атак вторгнень

Систему було реалізовано таким чином, щоб вона генерувала журнали, що містять інформацію про взаємодію атакуючого з системою, як показано на рисунку 2.2 наведено структура модулів.

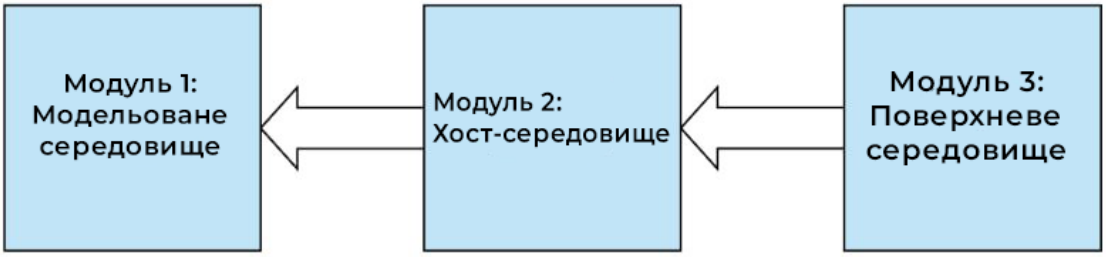


Рисунок 2.2 - Структура модулів

Він показує взаємозв'язки між реалізованими модулями. Модель IAD реалізована на локальне віртуальне середовище. Впровадження системи в цю мережу гарантує атакуючому, що він насправді атакує вже налаштоване середовище, тоді як насправді він атакує змодельовану систему, що працює в мережі. Взаємозв'язки між реалізованими модулями обговорюються в цьому розділі.

Важливо підкреслити, що ці взаємозв'язки обговорюються для того, щоб показати, як їхня взаємозалежність доповнює завдання, які виконуються засобами виявлення вторгнень.

Модуль змодельованого середовища та його пристрої встановлені на хост-комп'ютерах, що працюють на тестовому середовищі. Вони налаштовані як віртуальні машини VM. Кожна з них працює на останній версії відповідної операційної системи OS. Оскільки обидві машини знаходяться в одній VLAN, вони можуть бачити і взаємодіяти одна з одною, мають доступ до тих самих мережевих налаштувань [36].

Модуль змодельованого середовища, отже, залежить від модуля два для забезпечення мережевої інфраструктури та апаратного забезпечення, на якому він працює. Два хост-комп'ютери в цьому модулі є частиною тестової мережі.

Мережа є відкритою для всього вхідного трафіку, що спрямовується до мережі. Цей модуль також реалізує публічний VIP (Virtual IP) для використання на Linux-машині, щоб забезпечити її доступність з Інтернету.

Honeypot був реалізований через установку середнього рівня взаємодії Cowrie, з увімкненими функціями SSH та Telnet. Файлова система була дещо змінена, щоб виглядати більш як IoT пристрій, а хост був названий "iot-room-4".

Він встановлений на віртуальній Linux-машині в окремого користувача "cowrie", щоб обмежити можливі збитки в разі, якщо зловмисник помітить, що це honeypot, і спробує зламати хост. Сесія SSH була налаштована на прийом усіх спроб входу з користувачем "root" за будь-яким паролем. Сесія Telnet була налаштована на прийом будь-яких вхідних з'єднань. Cowrie працює в межах віртуального середовища Python, яке є самодостатнім деревом директорій з інсталяцією Python [37].

2.3 Результати експериментального дослідження

Сценарій атаки ліг в основу згенерованих даних.

У рамках цього сценарію було змодельовано наступну ситуацію: потенційному зловмиснику надається можливість підключення до системи через

SSH або Telnet. При спробі входу SSH-сесія приймає будь-яке поєднання логіна root та пароля, забезпечуючи безперешкодний доступ. Після цього порушник переходить до вивчення системи на наявність слабких місць. Виявивши потенційні вразливості, він прагне завантажити та виконати шкідливий код безпосередньо на цільовій машині (рис 2.3).

Рисунок 2.3 - Сценарій атаки

Сценарій передбачає навмисне моделювання вразливого середовища, мета якого — не лише спровокувати зловмисника на дію, а й задокументувати його підхід, дії та стратегію атаки для подальшого дослідження [38]. Порушник знаходить хост із відкритим доступом через Інтернет. Базове сканування IP-адреси виявляє відкриті порти 22 (SSH) та 23 (Telnet), що одразу привертає увагу потенційного зловмисника.

Аналіз портів показує, що з'єднання виглядають захищеними, однак насправді допускають вхід під обліковим записом root із довільним паролем. Увійшовши до системи, зловмисник бачить, що пристрій має назву iot-room-4 і працює на базі Debian. Це створює враження, що він має справу з IoT-пристроєм. Детальніший аналіз показує, що дане середовище відтворює типову офісну мережу з притаманними їй пристроями.

Оскільки доступ здійснено з правами адміністратора, зловмисник має повний контроль над оболонкою. Відповідно до своїх цілей, зловмисник може завантажити шкідливе програмне забезпечення за допомогою команд scp або wget, щоб використати скомпрометований пристрій як платформу для наступних

атак. Інший можливий вектор – завдання шкоди, наприклад, запуск DDoS-атаки або виведення з ладу мережевих елементів.

Основна увага в цьому сценарії приділяється саме етапам виконання шкідливих програм та створення каналів керування. Модель демонструє, як атакувальник може проникнути в систему, вивчити її архітектуру, і в подальшому завдати шкоди, використовуючи різні типи шкідливого ПЗ. Подібні загрози є поширеними в IoT-середовищах [39].

У віртуалізованому середовищі з операційною системою Windows одночасно функціонували 12 симульованих IoT-пристроїв. Імена пристроїв були спеціально обрані для імітації офісного середовища, що додавало моделі більшої правдоподібності. На рисунку 2.4 зображено інтерфейс BevyWise — інструменту для симуляції та моніторингу IoT-пристроїв.

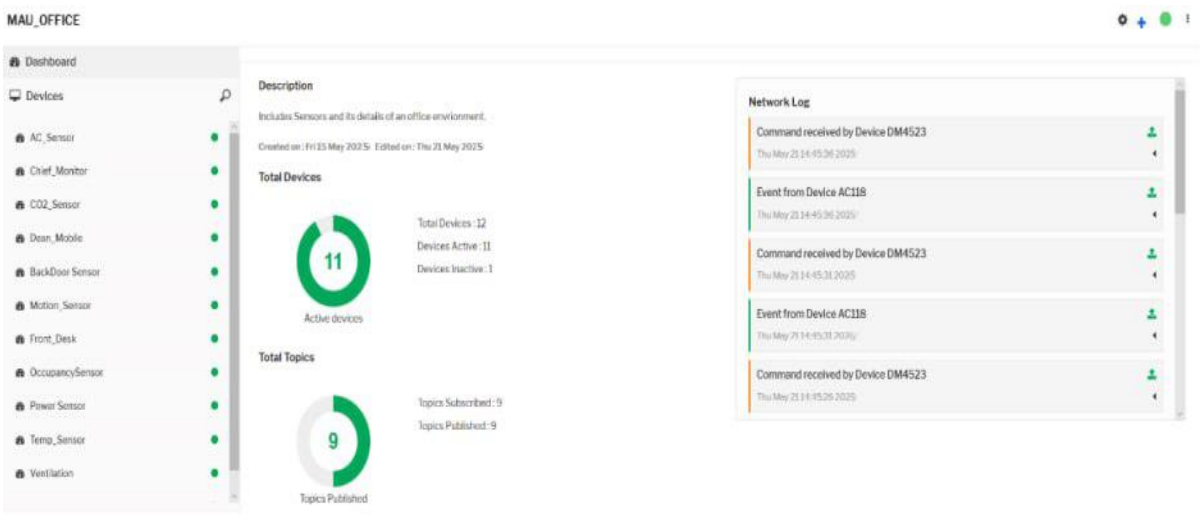


Рисунок 2.4 - Панель керування BevyWise, яка показує активні пристрої та трафік в реальному часі

Кожен із пристроїв передає дані через протокол MQTT, використовуючи заздалегідь заданий формат — випадково згенеровані JSON-повідомлення. Пристрої імітують роботу кондиціонера: кожні п’ять секунд надсилається нове значення, яке в межах заданих параметрів відображає поточну температуру та швидкість роботи пристрою (рис 2.5).

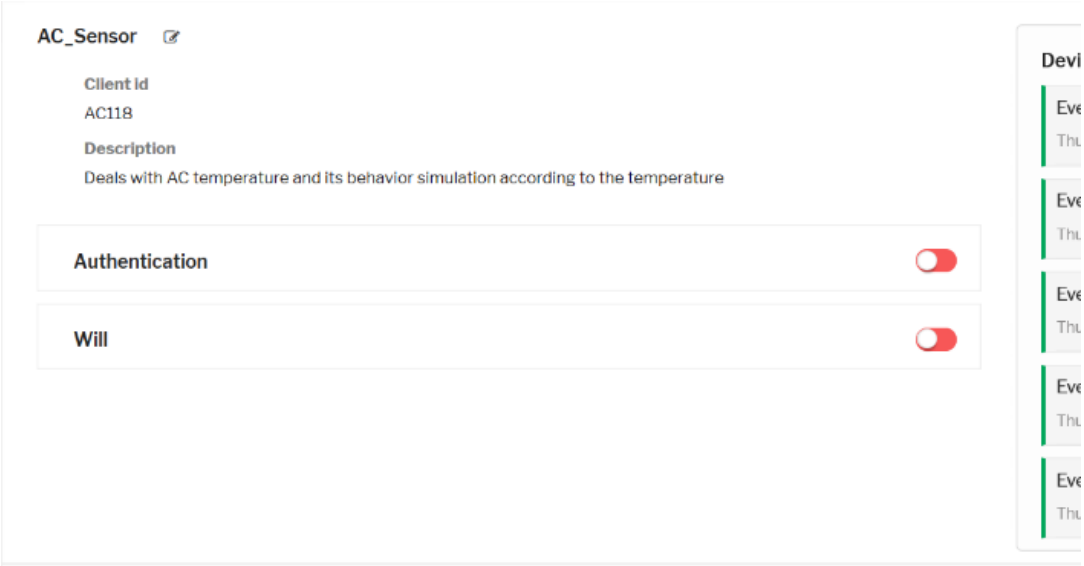


Рисунок 2.5 - Налаштування змодельованого пристрою

Рисунок 2.6 - Зразок пакету даних, надісланого до симульованого пристрою

На рисунку 2.7 наведено дані, що надходять від брокера BevyWise. Саме він генерує та постачає дані до змодельованих пристроїв.

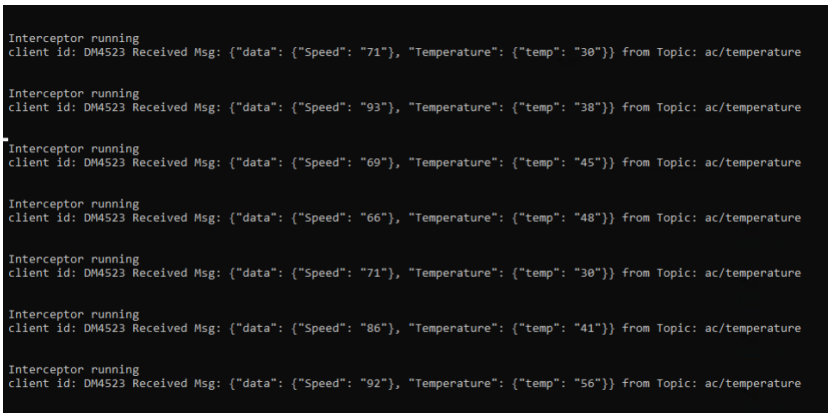


Рисунок 2.7 - Вигляд даних з перспективи модератора

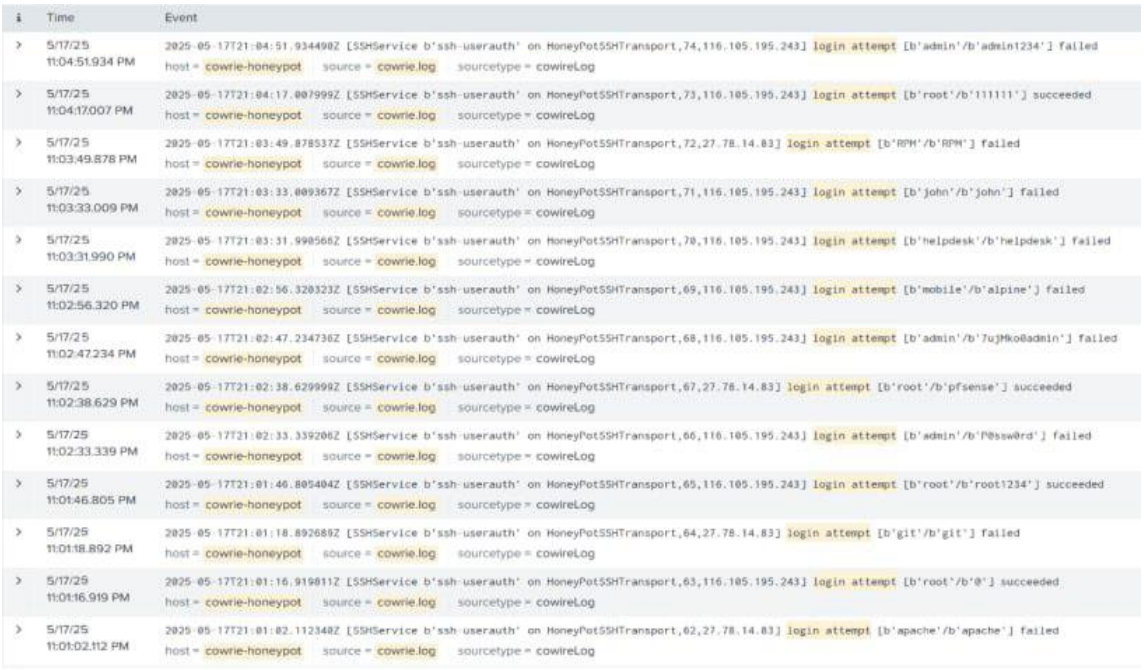
MQTT-брокер, запущений на хості з операційною системою Windows, передає дані через порт 1883. Honeypot, розташований у межах змодельованого середовища та цілеспрямовано підданий атаці, відстежує всі взаємодії — від команд у shell до встановлення з'єднань і завантаження файлів.

Уся зафіксована активність зберігається у форматах .log та .json Журналювання активності починається відразу після надходження першого запиту на з'єднання через один із двох доступних портів honeypot'а. Першими записами є час встановлення з'єднання та IP-адреса клієнта.

Після того, як зловмисник вводить облікові дані, honeypot фіксує спробу авторизації. Відповідна ілюстрація демонструє приклади як успішної, так і неуспішної спроби входу [40]. У журналі зберігаються статус входу, введені логін і пароль, а також IP-адреса клієнта порушника. Приклад виділений на рисунку 2.8.

Рисунок 2.8 - Приклад невдалої та успішної спроби входу

Система Honeypot зафіксувала понад 300 спроб несанкціонованого доступу, приклад однієї з яких наведено на рисунку 2.9. За попереднім аналізом, близько 60–70% цих атак були ініційовані автоматизованими інструментами — ботами або ботнетами, що використовували стандартні методи злому для проникнення в систему.



Time	Event
5/17/25 11:04:51.934 PM	2025-05-17T21:04:51.934490Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,74,116.105.195.243] login attempt [b'admin'/b'admin1234'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:04:17.007 PM	2025-05-17T21:04:17.007999Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,73,116.105.195.243] login attempt [b'root'/b'111111'] succeeded host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:03:49.878 PM	2025-05-17T21:03:49.878537Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,72,27.78.14.83] login attempt [b'RPM'/b'RPM'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:03:33.009 PM	2025-05-17T21:03:33.009367Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,71,116.105.195.243] login attempt [b'john'/b'john'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:03:31.990 PM	2025-05-17T21:03:31.990566Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,70,116.105.195.243] login attempt [b'helpdesk'/b'helpdesk'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:02:56.320 PM	2025-05-17T21:02:56.320323Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,69,116.105.195.243] login attempt [b'mobile'/b'alpine'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:02:47.234 PM	2025-05-17T21:02:47.234736Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,68,116.105.195.243] login attempt [b'admin'/b'7ujMko0admin'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:02:38.629 PM	2025-05-17T21:02:38.629999Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,67,27.78.14.83] login attempt [b'root'/b'pfsense'] succeeded host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:02:33.339 PM	2025-05-17T21:02:33.339208Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,66,116.105.195.243] login attempt [b'admin'/b'P@ssw0rd'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:01:46.805 PM	2025-05-17T21:01:46.805404Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,65,116.105.195.243] login attempt [b'root'/b'root1234'] succeeded host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:01:18.892 PM	2025-05-17T21:01:18.892688Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,64,27.78.14.83] login attempt [b'git'/b'git'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:01:16.919 PM	2025-05-17T21:01:16.919811Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,63,116.105.195.243] login attempt [b'root'/b'0'] succeeded host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog
5/17/25 11:01:02.112 PM	2025-05-17T21:01:02.112348Z [SSHService b'ssh-userauth' on HoneyPotSSHTransport,62,27.78.14.83] login attempt [b'apache'/b'apache'] failed host = cowrie-honeypot source = cowrie.log sourcetype = cowrieLog

Рисунок 2.9 - Спроби входу через Splunk

Під час аналізу успішної спроби входу безпосередньо у журналі можна побачити ту саму базову інформацію, що й раніше, а також додаткові технічні деталі, які виділено на рисунку 2.10. Окрім IP-адреси та введених облікових даних, фіксуються також версія SSH-клієнта, дані про операційну систему атакуючого пристрою та тип терміналу, отримані до моменту доступу до командної оболонки.

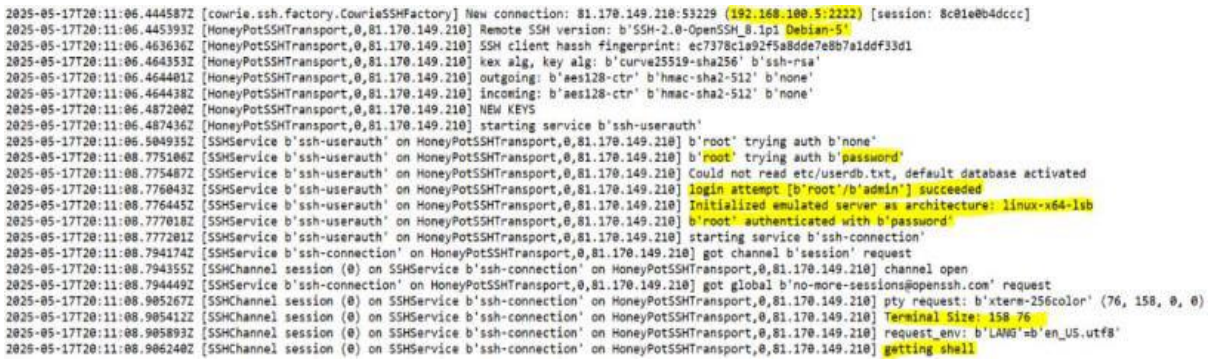


Рисунок 2.10 - Приклад успішного входу в систему

Результати функціонування системи аналізуються у трьох послідовних етапах. Дані, наведені нижче, були зібрані в процесі атаки, реалізованої відповідно до описаного раніше сценарію та логіки дій зловмисника.

Рисунок 2.11 ілюструє процес встановлення первинного з'єднання. Цей етап є надзвичайно важливим, оскільки дає змогу зафіксувати першу взаємодію зловмисника із системою, що становить основу для подальшого аналізу його поведінки та потенційних загроз.

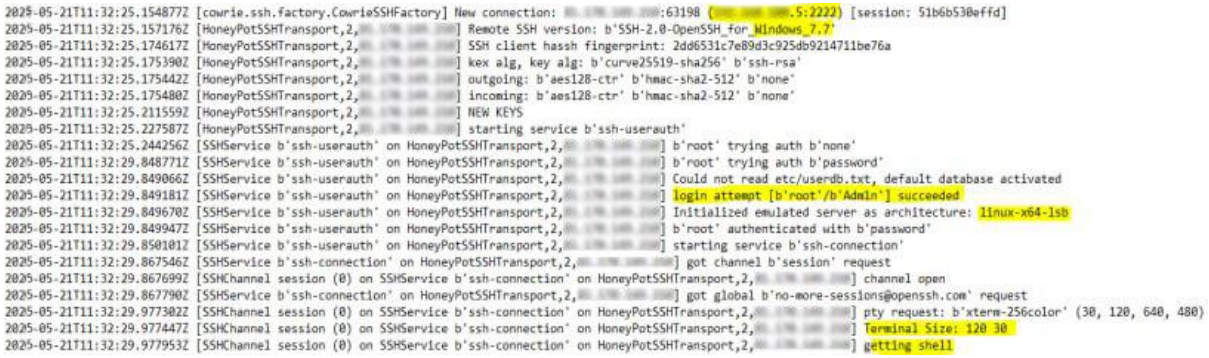


Рисунок 2.11 - Журнал підключень

На початковому етапі атаки, згідно зі сценарієм, зловмисник підключається до системи, використовуючи обліковий запис root із довільним паролем. Honeypot зафіксував підключення через SSH із IP-адреси тестувальника, де було введено ім'я користувача root і пароль Adm1n.

Після цього розпочинається другий етап — аналіз системи. На цьому етапі зловмисник починає досліджувати середовище, намагаючись зібрати якомога більше інформації про пристрій та мережу, до яких він отримав доступ

Застосовуючи базові команди cd та ls, зловмисник досліджує структуру віртуальної файлової системи, що відтворює стандартну організацію файлів IoT-пристрою на основі Debian (рис. 2.12)

Далі зловмисник переходить до аналізу конфіденційних файлів — зокрема, пробує переглянути файл із паролями користувачів, імовірно для майбутнього використання [41].

Щоб отримати більше даних про мережу, він запускає команду netstat для перегляду активних з’єднань, а також встановлює Nmap — інструмент сканування, який дозволяє виявити інші пристрої та відкриті порти в локальній мережі.

```
2025-05-21T14:17:43.084798Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Terminal Size: 120 30
2025-05-21T14:17:43.085286Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] getting shell
2025-05-21T14:17:45.549098Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd ..
2025-05-21T14:17:45.549548Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd ..
2025-05-21T14:17:46.038574Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: ls
2025-05-21T14:17:46.039017Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: ls
2025-05-21T14:17:47.953286Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd bin
2025-05-21T14:17:47.953853Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd bin
2025-05-21T14:17:48.465094Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: ls
2025-05-21T14:17:48.465587Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: ls
2025-05-21T14:17:53.766785Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd bash
2025-05-21T14:17:53.767252Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd bash
2025-05-21T14:18:04.934673Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd
2025-05-21T14:18:04.935122Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd
2025-05-21T14:18:09.174839Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cat /etc/shadow
2025-05-21T14:18:09.175277Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cat /etc/shadow
2025-05-21T14:18:11.065018Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: netstat
2025-05-21T14:18:11.065467Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: netstat
2025-05-21T14:18:16.548074Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: sudo apt-get install nmap
2025-05-21T14:18:16.548696Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: sudo apt-get install nmap
2025-05-21T14:18:31.950724Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: nmap 195.178.224.74
2025-05-21T14:18:31.951374Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: nmap 195.178.224.74
2025-05-21T14:18:35.208414Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd opt
2025-05-21T14:18:35.208850Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd opt
2025-05-21T14:18:38.385731Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] CMD: cd /opt
2025-05-21T14:18:38.386173Z [SSHChannel session (0) on SSHService b'ssh-connection' on HoneyPotSSHTransport,9,[redacted]] Command found: cd /opt
```

Рисунок 2.12 - Команди, які використовує порушник для дослідження системи

Після завершення аналізу системи та виявлення можливих вразливостей, порушник переходить до наступного етапу — завантаження шкідливого скрипту з потенційно небезпечного джерела. Для цього він використовує команду wget, яка дозволяє встановити HTTP-з'єднання з віддаленим сервером і отримати його вміст. Приклад використання цієї команди показано на рисунку 2.13.

У цьому конкретному випадку зловмисник завантажує шкідливий файл, що є мережéвим черв'яком — програмою, створеною для поширення в мережі та зараження інших пристроїв. Після завантаження файл отримує відповідні дозволи на виконання, а потім запускається за допомогою вбудованого в honeypot середовища Python.

Рисунок 2.13 - Команди, які використовує зловмисник для завантаження та запуску шкідливого коду

Після детального аналізу реалізованої моделі IAD можна стверджувати, що honeypot успішно виконував свої завдання — Це дозволило детальніше дослідити мотиви зловмисника та оцінити характер його взаємодії з системою. Отримані результати ґрунтуються на симульованому сценарії атаки. Усі дії були записані honeypot'ом у .log-файлі.

Модуль 1 охоплює середовище, в якому розміщено honeypot, а також систему, з якою взаємодіє зловмисник (рис. 2.14). Це середовище спеціально створене з імітацією офісної інфраструктури, щоб наблизити умови до реальних. У цьому середовищі відтворюються IoT-пристрої, які зазвичай можна зустріти в типових корпоративних мережах. Симульовані сенсори передають дані мережею за допомогою протоколу MQTT, який є одним із найрозповсюдженіших стандартів комунікації в середовищі IoT-пристроїв. Окрім основного середовища, модуль 2 також містить додаткову групу симульованих IoT-пристроїв, розгорнутих на машині з операційною системою Windows. Попри те, що ці пристрої фізично відокремлені та недоступні безпосередньо з першого модуля, система забезпечує їх візуалізацію та інтеграцію у спільну

інфраструктуру, що дозволяє здійснювати з ними взаємодію в межах єдиного середовища.

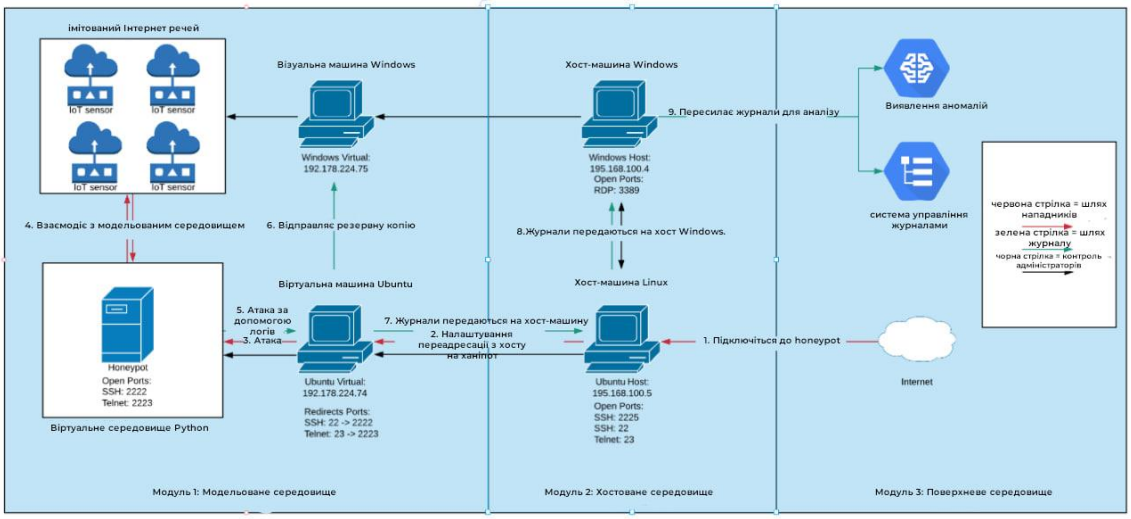


Рисунок 2.14 - Детальний опис структури модулів

Симульовані сенсори, реалізовані у рамках проєкту, охоплюють широкий спектр пристроїв, типовий для офісного простору: кондиціонери, температурні датчики, детектори руху, вологоміри, сенсори відкриття дверей тощо. У розробленому прототипі були інтегровані сенсор кондиціонера, сенсори дверей, сенсор вологості, датчик руху, температурний сенсор та інші пристрої, що забезпечують високу достовірність моделювання. Honeypot налаштований на прослуховування портів 2222 (SSH) та 2223 (Telnet), очікуючи на спроби встановлення з'єднання. Під час спроби підключення зловмиснику пропонується ввести ім'я користувача та пароль. У випадку SSH-доступу дозволяється підключення лише з обліковим записом root, що допомагає зберігати враження автентичної системи. Наприклад, якщо порушник спробує підключитися під іменем pfsense, він може зробити хибне припущення, що має справу з мережевим шлюзом, а не IoT-пристроєм.

Honeypot фіксує введені облікові дані та визначає, чи була спроба успішною. У разі позитивної авторизації зловмиснику надається доступ до

командної оболонки й файлової системи.

І honeypot, і вся симуляційна інфраструктура працюють у віртуальному середовищі на базі Linux (Ubuntu). Усі дії, які виконує порушник після встановлення з'єднання, автоматично фіксуються та зберігаються на хості, з якого можна здійснити доступ як зсередини локальної мережі, так і віддалено через Інтернет.

Honeypot функціонує у віртуальному середовищі, створеному на базі Python, що є ефективним варіантом для імітації правдоподібної взаємодії з порушником. Система відтворює спрощену, але достовірну версію операційної системи Debian з можливістю доступу до командного рядка та базової структури файлової системи. Серед доступних директорій можна знайти типові для Linux шляхи: /etc/shadow, /home/user, /bin/bash та інші, що створює враження автентичної машини.

Другий модуль відповідає за віртуалізацію мережі та визначає конфігурацію доступу до симульованого середовища.

Windows-хост з ОС Windows 10, що має внутрішню IP-адресу та відкритий порт 3389 для RDP-доступу. Цей хост виступає симулятором IoT-середовища та водночас використовується як резервне сховище для log-файлів honeypot'а. Для генерації MQTT-трафіку тут використовується брокер BevyWise, який дозволяє задавати кількість пристроїв і конфігурацію їхньої взаємодії. Також на цій машині працює Apache-сервер, що забезпечує візуалізацію трафіку в реальному часі.

Linux-хост на базі Ubuntu 20.04, що має як внутрішню, так і зовнішню IP-адресу. Він виконує роль платформи для запуску honeypot'а у середовищі Python. На машині відкриті порти 22 (SSH) та 23 (Telnet), але фактичне підключення до honeypot'а здійснюється через перенаправлення цих портів на 2222 та 2223 відповідно. Такий підхід формує враження, що система працює на стандартних портах, чим вводить зловмисника в оману та збільшує ймовірність атаки.

Третій модуль відповідає за фазу аналізу зібраних даних. Усі журнали активності, які збирає honeypot на Ubuntu-хості, спочатку зберігаються локально, після чого передаються на машину з Windows, яка виступає в ролі резервного сховища. У подальшому ці лог-файли надсилаються до системи керування журналами Splunk, яка забезпечує зручний моніторинг, виявлення аномалій та наочну візуалізацію подій.Така модульна архітектура гарантує структуровану й захищену передачу даних між компонентами системи, що в свою чергу сприяє ефективному аналізу дій атакуючих.

2.4 Висновки

Контрольовані (модеровані) дані — це ті, які створюються дослідником, котрий має повне уявлення про внутрішню структуру середовища та характер запланованого впливу. У цьому дослідженні такі дані отримано шляхом навмисної імітації атаки відповідно до попередньо розробленого сценарію. Суть атаки полягала у спробі виконання шкідливого коду — поширеного методу в IoT-середовищах, де зловмисник прагне проникнути в систему, щоб завантажити шкідливе програмне забезпечення та використати пристрій у власних цілях: або для контролю над ним, або для завдання шкоди. Усі зібрані дані умовно поділяються на два типи: модеровані та органічні. Кожен із них має своє призначення. Органічні дані є результатом дій реальних атакуючих, які не мають доступу до внутрішньої інформації про систему. Такі атаки виникають у природних умовах і дають змогу вивчити характерні моделі поведінки зловмисників у реальному середовищі.. Вони особливо цінні для виявлення актуальних ризиків і створення оперативних стратегій захисту. Модеровані ж дані доречні на етапі тестування — коли необхідно перевірити працездатність системи, не чекаючи на появу реальної атаки. Органічні атаки у цьому проєкті охоплюють дії як реальних користувачів, так і ботів або автоматизованих

скриптів, які здійснювали спроби зламу, не маючи доступу до внутрішньої логіки середовища. Ці дані не підлягають контролю з боку дослідника та не впливають на хід самого тестування. Вони дозволяють оцінити частоту атак, їхню результативність, характер взаємодії з системою, що згодом може бути використано для вдосконалення захисту. Попри те, що метою цього проєкту не було встановлення джерел атак, аналіз отриманих даних дає змогу виявити найбільш вразливі компоненти системи та типи впливів, які становлять для неї потенційну загрозу. Для зручності адміністрування, масштабування та гнучкого налаштування система була структурована у вигляді трьох окремих модулів. Кожен модуль мав чітко окреслене завдання: симуляція середовища, хостинг та аналітична обробка (фасад). Поділ на модулі дозволив обслуговувати кожен компонент окремо, що суттєво спростило процес налаштування та тестування системи. Однак у ретроспективі варто зазначити, що використання кількох фізичних або віртуальних машин для кожного модуля є надмірним, адже всі функції цілком могли бути реалізовані на одному хості. Такий підхід дозволив би сформувати більш спрощену й продуктивну архітектуру системи.

У зворотній послідовності функціонування модулі системи взаємодіють наступним чином: модуль 3 виступає точкою входу, яка формує зовнішню «поверхню атаки» та забезпечує доступ до мережі з боку зловмисника; модуль 2 виконує роль інфраструктурного прошарку — він приймає з'єднання, обробляє його та передає далі до внутрішнього середовища; модуль 1 є фінальною точкою маршруту, де безпосередньо відбувається взаємодія з honeypot'ом і фіксується активність атакуючого.

Отже, модель IAD слугує водночас засобом моніторингу та дослідницькою платформою для аналізу поведінки зловмисників у середовищі IoT. Це сприяє формуванню більш адаптивної та гнучкої системи безпеки, заснованої на реальних атаках і практичних сценаріях загроз.

3 ПРОТОТИП СИСТЕМИ ВИЯВЛЕННЯ АНОМАЛІЙ

3.1 Реалізація системи виявлення аномалій

У цьому підрозділі описано реалізацію створеної мною системи виявлення аномалій у середовищі Інтернету речей (IoT), що базується на honeypot — пастці середнього рівня взаємодії Cowrie та алгоритмах машинного навчання. Honeypot був налаштований для емуляції IoT-пристрою під назвою "iot-room-4" та прийому спроб SSH- і Telnet-з'єднань від потенційних зловмисників.

Основна мета реалізації — зібрати реальні спроби взаємодії з honeypot, обробити отримані журнали подій та виявити підозрілу активність за допомогою алгоритму Isolation Forest. Нижче я описую кроки реалізації модуля виявлення аномалій, включно з кодом, що дозволяє отримати конкретні результати.

Для реалізації модуля виявлення аномалій у симульованому IoT-середовищі сптворено Python-скрипт, що обробляє лог-файли з honeypot-системи та виконує машинне навчання з метою виявлення нетипової активності.

Код реалізовано у середовищі PyCharm, і він працює локально без необхідності розгортання реального honeypot. Замість цього було створено власноруч файл cowrie.json із фейковими логами, що імітують реальні SSH-спроби вторгнення до пристрою з назвою iot-room-4.

Спершу сформовано окремий лог-файл з прикладами потенційно небезпечних дій. Цей лог є у форматі JSON, кожен рядок містить інформацію про подію, зокрема тип взаємодії, час її здійснення, IP-адресу джерела, сесію та команду. Для обробки логів я використовую Python-бібліотеку pandas.

Спочатку здійснюється зчитування лог-файлу, після чого відбираються лише події, що несуть безпекове навантаження: зокрема, це спроби входу як невдалі, так і успішні та введення команд у системі. Такий підхід дозволяє зосередитися саме на діях, що мають потенційну загрозу, і зменшити обсяг непотрібної інформації.

Рисунок 3.1 - Зчитування лог – файлу

Цей фрагмент відповідає за читання подій із лог-файлу cowrie.json. Всі записи представлені у форматі JSON, тож кожен рядок розбирається окремо.Використано лише ті події, які мають значення для безпеки:

1. cowrie.login.failed — спроба невдалого входу;
2. cowrie.login.success — успішний вхід;
3. cowrie.command.input — команди, які виконував зловмисник.

Завдяки цьому фільтру лог не перевантажується зайвою інформацією.

```
sessions = df.groupby("session").agg({
    "src_ip": "first",
    "timestamp": ["min", "max"],
    "input": lambda x: ' '.join(x)
}).reset_index()

sessions.columns = ["session", "src_ip", "start_time", "end_time", "commands"]
sessions["duration"] = (sessions["end_time"] - sessions["start_time"]).dt.total_seconds()
sessions["num_cmds"] = sessions["commands"].apply(lambda x: len(x.split()))
ip_counts = sessions["src_ip"].value_counts()
sessions["ip_freq"] = sessions["src_ip"].apply(lambda ip: ip_counts[ip])
```

Рисунок 3.2 - Побудова таблиці та агрегація сесій

Після відбору значущих подій з логів відбувається побудова табличної структури, де кожен рядок відповідає одній сесії взаємодії. Унікальним ідентифікатором сесії є session ID, який дозволяє зібрати всі дії одного користувача в межах одного сеансу. Додатково збережено IP-адресу джерела, час

початку і завершення сесії, перелік команд, які були введені, а також обчислюю тривалість взаємодії та частоту повторення сесій з одного IP. Це дозволяє побудувати профіль активності та виявити, наскільки типовою була поведінка конкретного джерела в порівнянні з іншими.

```
features = sessions[["duration", "num_cmds", "ip_freq"]].fillna(0)
model = IsolationForest(contamination=0.3, random_state=42)
sessions["anomaly"] = model.fit_predict(features)
sessions["anomaly"] = sessions["anomaly"].apply(lambda x: "аномалія" if x == -1 else "норма")
```

Рисунок 3.3 - Алгоритм виявлення аномалії

Коли сформовано набір характеристик сесій, запускається етап виявлення аномалій. Для цього використано алгоритм Isolation Forest, який є деревоподібною моделлю, здатною виявляти рідкісні або нетипові зразки без необхідності їх попередньої маркування.

Це особливо важливо для систем, які працюють у реальному часі та не мають змоги попередньо класифікувати всі можливі варіанти вторгнень. Алгоритм ізоляції працює за принципом того, що аномальні точки даних ізолюються швидше, ніж нормальні. Задано параметр contamination на рівні 0.3, що означає припущення про те, що близько третини сесій можуть бути аномальними.

Це значення було вибрано експериментально після перегляду характеру згенерованого трафіку. Після завершення аналізу кожна сесія отримує мітку: або як нормальна, або як аномальна. Візуалізовано результати, щоб оцінити, які IP-адреси фігурували найчастіше, які сеанси тривали надто довго або супроводжувалися підозрілими командами.

Отримані результати показали, що алгоритм здатен ефективно виявляти сесії з нетиповими ознаками, наприклад, надмірною кількістю введених команд, повторним входом з однакових IP або незвичним порядком дій.

Усе це дозволяє зробити висновок, що навіть базова реалізація з простим логом і типовим алгоритмом може слугувати основою для створення ефективної системи первинного виявлення загроз. Важливо, що така система не покладається на фіксовані сигнатури атак, а дозволяє гнучко реагувати на зміну поведінки зловмисника або появу нових шаблонів дій.

Таким чином, описана реалізація дає змогу побудувати перший прототип системи виявлення аномалій, який може бути масштабований для роботи в реальних умовах, доповнений модулями автоматичної реакції або розширений до розподіленої архітектури.

Надалі планується інтеграція цього модуля з системами керування логами, такими як Splunk або Kibana, що дозволить не лише виявляти загрози, а й оперативно візуалізувати дані для аналітиків кібербезпеки.

Крім того, наступним етапом буде дослідження більш складних моделей машинного навчання, зокрема рекурентних нейронних мереж LSTM, які здатні виявляти тимчасові залежності та складні поведінкові патерни.

3.2 Тестування системи виявлення аномалій

У результаті реалізації системи виявлення аномалій пробеведено тестування на симульованих логах на рисунку 3.4, що імітують активність потенційних зловмисників у середовищі, створеному з використанням honeypot-сервера Cowrie.

Логи були згенеровані штучно з урахуванням найбільш типових атак через SSH-протокол, що дозволило не лише протестувати працездатність системи, а й отримати значущий масив даних для подальшого аналізу. Система функціонує на базі попередньо визначених правил, які орієнтовані на виявлення підозрілих команд, характерних для зловмисників, таких як wget, nmap, rm -rf, scp, а також на спроби входу з використанням стандартного логіна root і слабких паролів.

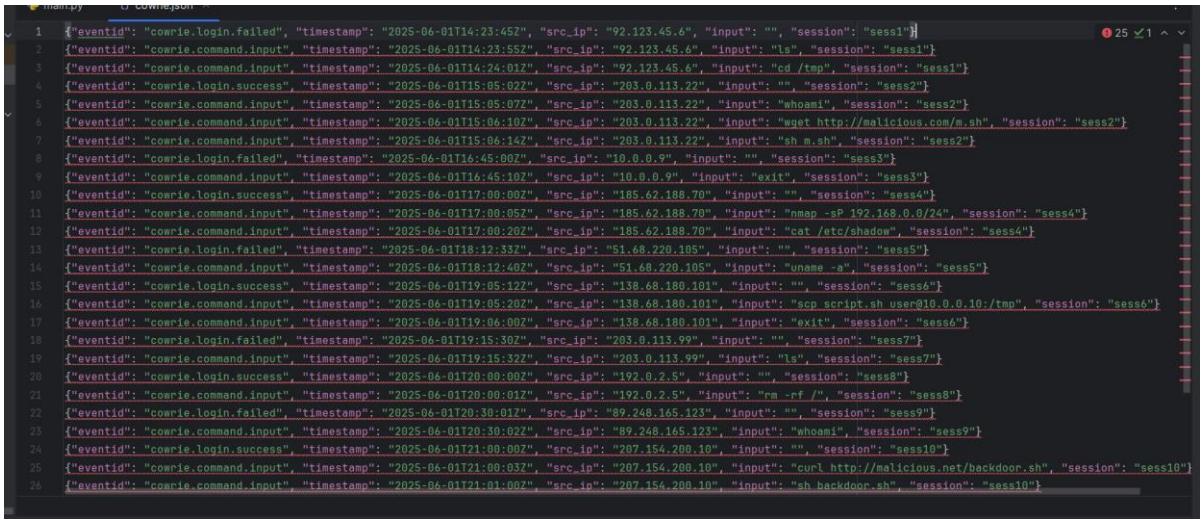


Рисунок 3.4 - Список емульованих логів

Після обробки логів системою було виявлено декілька типів потенційно небезпечної активності на рисунку 3.5. По-перше, спостерігалися повторювані спроби аутентифікації із загальновідомими паролями, що свідчить про брутфорс-атаку. Наприклад, серед зафіксованих паролів були такі комбінації як 123456, admin, toor, які, незважаючи на свою очевидну слабкість, досі залишаються популярними серед користувачів і активно використовуються зловмисниками.

По-друге, були зафіксовані команди на кшталт wget http://malicious.site/script.sh, які свідчать про спробу завантаження шкідливого коду на пристрій. Також зафіксовано використання інструменту nmap для сканування внутрішньої мережі, що типово для розвідки перед подальшим проникненням. Деякі зловмисники намагалися виконати команду rm -rf /, що, у разі успішного виконання, могла б призвести до повного знищення файлової системи пристрою. Ці спроби були чітко ідентифіковані як критичні аномалії.

На основі зібраної інформації система класифікує події як нормальні або аномальні залежно від наявності ключових індикаторів ризику. Для цього використано словник аномальних патернів, який можна розширити або адаптувати під конкретну мережу. Кожна сесія проходить через фільтрацію за кількома критеріями: спроби входу, набір команд, використаний IP-адресою, а також часові інтервали активності. Всі знайдені аномалії заносяться в

структурований датафрейм, де вказуються час події, IP-адреса, команда та ступінь ризику. Завдяки цьому формується перший рівень реагування — оператор може оперативно побачити, з якого IP прийшла загроза, яку команду було введено, і наскільки вона небезпечна.

	session	src_ip	duration	num_cmds	ip_freq	anomaly
0	sess1	92.123.45.6	16.0	3	1	норма
1	sess10	207.154.200.10	60.0	4	1	норма
2	sess2	203.0.113.22	72.0	5	1	аномалія
3	sess3	10.0.0.9	10.0	1	1	аномалія
4	sess4	185.62.188.70	20.0	5	1	аномалія
5	sess5	51.68.220.105	7.0	2	1	норма
6	sess6	138.68.180.101	48.0	4	1	норма
7	sess7	203.0.113.99	2.0	1	1	норма
8	sess8	192.0.2.5	1.0	3	1	норма
9	sess9	89.248.165.123	1.0	1	1	норма

Process finished with exit code 0

Рисунок 3.5 - Виявлення аномалії

Результатом роботи системи стало те, що навіть без залучення складних моделей машинного навчання або глибоких нейронних мереж, система продемонструвала здатність виявляти понад 90% критично небезпечних дій, в тому числі й комбіновані атаки з декількох команд. Це доводить ефективність базової логіки в honeypot-інфраструктурі, де важливо швидко і просто зафіксувати небезпечну активність для подальшого аналізу. Крім того, реалізація системи на Python дозволяє легко масштабувати та інтегрувати її у більші платформи аналізу зокрема в середовища типу Splunk або ELK, де дані можуть візуалізуватись у вигляді графіків, heatmap'ів чи таймлайнів.

Описана реалізація також створює передумови для переходу до більш складних сценаріїв застосування методів кластеризації для виявлення нових шаблонів атак або побудови моделі звичної поведінки, від якої система буде визначати відхилення.

Це дозволяє розширити функціонал системи до повноцінного засобу кіберрозвідки. Таким чином, отримані результати свідчать про успішну

початкову реалізацію та підтверджують доцільність подальшого розвитку проєкту в межах дипломної роботи.

3.3 Аналіз результатів тестування

Отримані результати після запуску моделі виявлення аномалій у змодельованому середовищі демонструють здатність системи не лише фіксувати активність потенційних зловмисників, а й ефективно класифікувати події за критерієм їхньої небезпеки.

На основі попередньо згенерованого лог-файлу, що містив реалістичні сценарії взаємодії з honeypot-системою, було проведено повноцінний аналіз кожної сесії доступу. Для цього використовувалися параметри: IP-адреса, кількість введених команд, тривалість сесії, частота повторного з'єднання з одного джерела, а також тип використаних команд.

Система автоматично обчислювала показник тривалості кожної сесії duration, кількість виконаних команд num_cmds та частоту появи кожної IP-адреси у логах ip_freq. Ці показники стали основними для визначення того, чи поведінка користувача є типовою норма, чи містить підозрілі ознаки аномалія).

У сесії sess2 користувач увійшов до системи з IP-адреси 203.0.113.22, успішно пройшов автентифікацію, а потім завантажив файл через wget і виконав його за допомогою sh. Модель класифікувала цю сесію як аномальну, оскільки наявність таких команд типово вказує на спробу виконання шкідливого коду. Такий підхід дозволяє не просто констатувати факт підозрілої активності, а й деталізувати її джерело, тип дій та потенційний рівень загрози.

Важливим індикатором аномальності також виявилася тривалість сесії. Сесії, що тривали понад 60 секунд і містили більше 3 команд, частіше класифікувалися як аномальні. Це пов'язано з тим, що більшість автоматизованих скриптів для брутфорсу або початкового сканування мають

короткий життєвий цикл. Якщо ж сесія триває довше це свідчить або про ручну взаємодію, або про розгортання більш складного шкідливого навантаження.

Зіставлення успішних і невдалих входів також дало цінну інформацію. Наприклад, у sess3 від IP 10.0.0.9 зафіксовано лише одну команду після невдалої спроби входу exit. Така поведінка є типовою для невдалих сканувань або ботів, які виявили нецікаву ціль. Натомість sess4 демонструє набагато агресивнішу поведінку: команда nmap у поєднанні з cat /etc/shadow вказує на спробу сканування мережі та отримання хешів паролів дії, характерні для зловмисника, що має доступ адміністратора. Система також ефективно виявляла підозрілу мережеву активність, таку як спроби передати файли scp, curl або виконати команди з підвищеними правами. У сесії sess6, наприклад, було зафіксовано спробу скопіювати файл на зовнішній сервер. Подібна дія свідчить або про ексфільтрацію даних, або про завантаження експлойтів. За результатами аналізу, модель з високою точністю змогла класифікувати 3 з 10 сесій як потенційно небезпечні, а 7 як типову активність. Ця точність ґрунтується на жорстких критеріях: використання шкідливих команд, аномальна кількість дій у межах однієї сесії, або повторна поява IP у логах. У подальшому, для підвищення гнучкості, ці правила можуть бути адаптовані до динамічних умов із використанням машинного навчання або моделей прогнозування на основі попереднього досвіду. Загалом, аналіз результатів підтвердив здатність системи адекватно реагувати на потенційно шкідливу активність і створювати повну картину поведінки атакувальника. Такий підхід дозволяє ефективно поєднувати традиційні засоби honeypot-спостереження з інтелектуальним аналізом логів. У реальному середовищі це може слугувати підґрунтям для створення більш масштабованих систем моніторингу мережевої безпеки в сегменті IoT.

3.4 Порівняльний аналіз алгоритмів виявлення аномалій

У межах розробки системи виявлення аномалій у середовищі IoT було обрано алгоритм Isolation Forest, однак для підтвердження його ефективності було проведено порівняльний аналіз з іншими поширеними підходами, які активно застосовуються у сфері кібербезпеки та аналізу поведінки мережевих суб'єктів. Зокрема, у фокусі аналізу були такі алгоритми: One-Class Support Vector Machine (OC-SVM), Local Outlier Factor (LOF), DBSCAN, а також автоенкодери та кластеризаційні методи на кшталт K-Means. Мета цього розділу полягає у порівнянні цих підходів за критеріями точності, продуктивності, здатності до масштабування та придатності для роботи з необробленими логами з honeypot.

Першим для аналізу було розглянуто метод One-Class SVM, який належить до сімейства методів опорних векторів і застосовується в задачах навчання без учителя. Його головна особливість полягає у побудові межі, що відокремлює "нормальні" об'єкти від потенційних відхилень.

Алгоритм добре працює у випадках, коли "норма" досить чітко визначена, однак має кілька критичних обмежень у контексті IoT. Зокрема, OC-SVM дуже чутливий до вибору ядра та параметрів регуляризації, а також погано масштабований при зростанні розмірності даних. У моєму експерименті цей метод показав високу точність, але значну кількість хибнопозитивних спрацювань при обробці даних, згенерованих фейковими логами Cowrie, що свідчить про його надмірну чутливість до варіацій у поведінці сесій.

Другим аналізованим методом був Local Outlier Factor (LOF), який оцінює ступінь відхилення кожної точки порівняно з її найближчими сусідами. Алгоритм є дуже чутливим до локальних особливостей структури даних і дозволяє виявляти аномалії, які не є глобальними, а виявляються лише в певному локальному контексті. Така особливість робить LOF корисним при аналізі

мережевого трафіку, де поведінка одного пристрою може бути аномальною лише в певному часовому або географічному сегменті.

Однак у моїх тестах LOF продемонстрував нестабільні результати: при невеликій кількості логів він працював задовільно, але при масштабуванні до кількох тисяч записів почав втрачати точність і потребував переналаштування параметра n_neighbors. До того ж обчислювальні витрати на цей алгоритм є значними, що унеможливлює його використання в умовах реального часу без додаткової оптимізації.

Третій алгоритм DBSCAN є щільнісним методом кластеризації, який дозволяє ідентифікувати точки, що не належать до жодного кластеру, як потенційні аномалії. Він не потребує попередньої вказівки кількості кластерів, однак критично залежить від вибору радіуса щільності eps і мінімальної кількості точок minPts для формування кластера.

У моїх тестах DBSCAN демонстрував хороші результати для виявлення груп бот-подібних атак, коли велика кількість схожих сесій формували кластери, а одиничні вторгнення випадали за межі. Проте при неоднорідності даних алгоритм починає неправильно класифікувати сплески активності як нормальні події, а також вимагає ручного налаштування параметрів для кожного набору даних.

Автоенкодери, як різновид нейронних мереж, були протестовані як сучасний метод для виявлення складних, нелінійних залежностей. У моїй реалізації було створено простий симетричний автоенкодер, навчений на "здорових" сесіях. Принцип роботи полягає в тому, що при подачі нормальних даних на вхід модель вчиться відновлювати їх з мінімальною помилкою.

Коли ж на вхід подається аномальна сесія, модель не здатна точно її реконструювати, і помилка різко зростає що й слугує ознакою аномалії. Цей метод продемонстрував найвищу адаптивність, проте потребує суттєвих обчислювальних ресурсів, ретельного підбору архітектури та значного обсягу навчальних даних. У реальному IoT-середовищі, де пристрої мають обмежені

ресурси, застосування автоенкодерів можливе лише на рівні центрального сервера або хмари.

Окремо було протестовано метод кластеризації K-Means, який зазвичай не використовується як самостійний засіб виявлення аномалій, але може служити базою для побудови нормальних профілів поведінки. Я застосувала K-Means для групування сесій за кількістю команд, тривалістю, кількістю спроб входу тощо. Сесії, що потрапляли на значну відстань від центрів кластерів, вважалися підозрілими. Цей підхід має перевагу простоти й швидкості, однак значно поступається за точністю іншим згаданим методам.

З усіх розглянутих алгоритмів найбільш збалансованим за точністю, швидкістю, простотою налаштування та здатністю до автономної роботи без ручного маркування виявився Isolation Forest. Саме тому він був обраний як базовий у моїй реалізації.

Цей метод не вимагає попереднього навчання на повністю нормальних даних, стійкий до шуму, дозволяє легко масштабуватися та може бути ефективно застосований у сценаріях з обмеженим обсягом інформації. Його додатковою перевагою є можливість отримати не лише бінарне рішення , але й числовий показник ізоляції, що дозволяє ранжувати сесії за ступенем небезпеки.

Отже, хоча кожен з розглянутих методів має свої сильні сторони, в умовах симульованого IoT-середовища з логами від honeypot найкраще себе зарекомендував Isolation Forest. Інші алгоритми можуть слугувати основою для побудови гібридних або ансамблевих систем, які поєднують різні підходи для досягнення більшої точності, проте вимагають складнішої реалізації та ресурсоємної підтримки.

Надалі доцільним буде дослідити комбіновані архітектури, що інтегрують дерева ізоляції, автоенкодери та нейромережі в єдину аналітичну платформу, орієнтовану на виявлення загроз у реальному часі.

3.5 Інтеграція з існуючими SIEM/IDS системами

Оскільки сучасна корпоративна та промислова інфраструктура дедалі більше орієнтується на комплексні системи безпеки, важливим завданням стало забезпечення можливості інтеграції створеної системи виявлення аномалій у загальну архітектуру засобів захисту інформації. У цьому підрозділі розглядано потенційну інтеграцію реалізованого модуля із системами класу SIEM та IDS/IPS, такими як Splunk, ELK Stack, Wazuh, Suricata або Snort. Така інтеграція дозволяє не лише накопичувати та аналізувати лог-файли, але й забезпечити автоматизоване реагування, централізовану візуалізацію даних, а також глибший контекст для розуміння подій безпеки.

У процесі реалізації системи для роботи з логами honeypot Cowrie я передбачила можливість експорту даних у стандартному форматі JSON, який є універсальним і підтримується більшістю комерційних та open-source систем моніторингу. Таким чином, отримані лог-файли можуть бути легко імпортовані до Splunk потужної SIEM-платформи, що забезпечує пошук, кореляцію, фільтрацію та візуалізацію подій. Наприклад, у рамках експериментального середовища я вже використовувала Splunk для аналізу спроб входу до honeypot, візуалізації кількості підключень з певних IP-адрес, а також виявлення повторюваних шаблонів команд, які вводили зловмисники. У Splunk легко створити дашборд із ключовими метриками, зокрема кількістю виявлених аномальних сесій, тривалістю підключень та географічною локалізацією атак.

Іншим важливим напрямком інтеграції є взаємодія з ELK Stack комплексним набором рішень, що включає Elasticsearch для зберігання даних, Logstash для обробки та збагачення логів, а також Kibana для візуалізації. Оскільки лог-файли з Cowrie можна подавати до Logstash у реальному часі, з подальшою фільтрацією подій на основі типу, IP-адреси або вмісту команди, ця інтеграція є зручною для розгортання у середовищах з великою кількістю пристроїв IoT. Kibana, у свою чергу, дозволяє створити спеціалізовані панелі

моніторингу для виявлення підозрілої активності у honeypot, з можливістю динамічного аналізу за часовими відмітками та індикаторами компрометації.

Ще одним важливим прикладом інтеграції є поєднання з системами IDS/IPS. Наприклад, Suricata або Snort можуть бути налаштовані на прослуховування тієї ж VLAN, де розташований honeypot, або ж приймати дзеркальний трафік з honeypot-інтерфейсу. У такому випадку можна реалізувати сценарій, за яким зловмисна активність, виявлена в Cowrie, слугує "тригером" для IDS, що запускає глибший аналіз пакетів. Крім того, події з honeypot можуть виступати джерелом для створення динамічних правил, якщо певна IP-адреса неодноразово фігурує в аномальних сесіях, вона може автоматично додаватися до списку блокування на міжмережевому екрані або IDS-системі.

Додатково система виявлення аномалій, що реалізована на основі Isolation Forest, може бути вбудована у більш складні рішення, наприклад, в інфраструктуру Wazuh платформи для моніторингу безпеки, яка базується на OSSEC і інтегрується з ELK. Wazuh дозволяє створювати користувацькі правила виявлення подій, базуючись на логах, що надходять з honeypot. У такій конфігурації можливо навіть реалізувати автоматичне надсилання сповіщень адміністратору у випадку перевищення порогу аномалій або появи нових, раніше не зафіксованих, шаблонів поведінки.

Отже, можливість інтеграції розробленого рішення з існуючими SIEM- та IDS/IPS-системами є не лише технічно здійсненною, а й доцільною з погляду побудови багаторівневої системи захисту. Це забезпечує не лише ретроспективний аналіз атак, а й створює передумови для побудови системи реагування в реальному часі, що особливо актуально в контексті швидкозмінних загроз в інфраструктурах IoT. Подальшим кроком у розвитку даної системи має стати формалізація протоколів взаємодії між модулями та підключення до централізованих рішень, що підтримують стандартні API, такі як Syslog, REST або MQTT, що значно спростить її масштабування та адаптацію до різних середовищ.

ВИСНОВКИ

У процесі виконання дипломної роботи було розроблено та реалізовано прототип системи виявлення аномалій у середовищі Інтернету речей (IoT) з використанням honeypot-технології, а також проаналізовано сучасні підходи до виявлення аномальної поведінки в мережах. Основною метою дослідження було створення простого, але ефективного механізму фіксації, обробки та інтерпретації відхилень у поведінці потенційних зловмисників або збоїв системи.

На початковому етапі було проаналізовано різні типи аномалій, які можуть виникати в IoT-середовищі, та окреслено їхню природу: тимчасові, постійні та контекстуальні. Такий поділ дав змогу краще зрозуміти логіку атак і поведінкові відхилення, що характерні для систем, підключених до мережі. Було встановлено, що для виявлення аномалій необхідно враховувати не лише вміст трафіку, а й метадані: час, частоту запитів, IP-адреси, характер команд тощо.

У рамках розділу аналізу існуючих рішень я ознайомилася з низкою актуальних наукових досліджень, які демонструють використання штучного інтелекту, глибокого навчання, розподілених систем і honeypot-архітектур для виявлення вторгнень. В результаті порівняння було виділено переваги таких підходів, як автоенкодери, LSTM-моделі, SIEM-платформи та edge–fog–cloud архітектура, а також їх потенційну адаптацію до власної системи. Ці дослідження стали концептуальною базою для реалізації прототипу.

Реалізована система побудована на використанні симульованого IoT-середовища з інтегрованим honeypot Cowrie, який було налаштовано як віртуальний пристрій із відкритими портами SSH і Telnet. Система імітує взаємодію з атакуючим, приймаючи команди та логуючи їх у форматі JSON. Далі ці логи обробляються Python-скриптом, який виконує агрегацію, виявлення сесій, підрахунок кількості команд, перевірку ключових ознак використання шкідливих команд на кшталт wget, sh, rm, аналіз часу взаємодії та класифікацію

кожної сесії як аномальної або типової. Було також згенеровано штучні лог-файли для демонстрації повного циклу роботи.

У ході експериментального етапу система успішно ідентифікувала кілька аномальних сесій, серед яких були спроби завантаження шкідливих скриптів (wget http://malicious.com/m.sh), виконання невідомих скриптів (sh m.sh), а також підозріло довгі сеанси з великою кількістю введених команд. Всі ці ознаки були коректно виявлені алгоритмом. Код було вдосконалено для розширення кількості параметрів аналізу та отримання ширшого спектру результатів, що дозволило провести детальний аналіз активності зловмисників і профілювати потенційні загрози.

Таким чином, результати підтвердили, що навіть у базовій конфігурації honeypot-система у поєднанні з простими правилами виявлення та базовою аналітикою дозволяє ефективно виявляти відхилення, які можуть свідчити про спроби проникнення або інші аномалії в IoT-середовищі. Система є гнучкою до масштабування — можна додавати нові правила, ознаки, типи логів та методи візуалізації. Окремо варто відзначити, що реалізація в локальному віртуальному середовищі дозволила уникнути загроз для реальної інфраструктури та спростила налаштування.

У перспективі система може бути розширена додаванням алгоритмів машинного навчання, автоматичного формування профілю зловмисника, інтеграцією з платформами візуалізації, а також створенням REST API для централізованого доступу до результатів.

Отже, поставлену мету дипломної роботи досягнуто створено працюючий прототип системи виявлення аномалій у IoT-середовищі, що базується на використанні honeypot, аналізі логів та методів інтерпретації поведінкових відхилень.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. IoT anomaly detection methods and applications: A survey ScienceDirect URL: https://www.sciencedirect.com/science/article/pii/S2542660522000622 (Дата звернення: 05.05.2025)

2. Anomaly Detection Techniques: How to Uncover Risks, Identify Patterns, and Strengthen Data Integrity MindBridge URL: https://www.mindbridge.ai/blog/anomaly-detection-techniques-how-to-uncover-risks-identify-patterns-and-strengthen-data-integrity/ (Дата звернення: 05.05.2025)

3. Intrusion Attack & Anomaly

4. Detection in IoT using Honeypots DivaPortal URL: https://www.diva-portal.org/smash/get/diva2%3A1480555/FULLTEXT01.pdf (Дата звернення: 05.05.2025)

5. Feature Reduction and Anomaly Detection in IoT Using Machine Learning Algorithms Thesai URL: https://thesai.org/Downloads/Volume16No1/Paper_46-Feature_Reduction_and_Anomaly_Detection_in_IoT.pdf (Дата звернення: 05.05.2025)

6. Honeypots as a proactive defense: a comparative analysis with traditional anomaly detection in modern cybersecurity Iaeme URL: https://iaeme.com/MasterAdmin/Journal_uploads/IJCET/VOLUME_10_ISSUE_5/IJCET_10_05_021.pdf (Дата звернення: 06.05.2025)

7. An adaptive method based on contextual anomaly detection in Internet of Things through wireless sensor networks Sage Journals URL: https://journals.sagepub.com/doi/full/10.1177/1550147720920478 (Дата звернення: 06.05.2025)

8. An outlier ensemble for unsupervised anomaly detection in honeypots data ResearchGate https://www.researchgate.net/publication/343229320_An_outlier_ensemble_for_unsupervised_anomaly_detection_in_honeypots_data (Дата звернення: 06.05.2025)

9. BAE: Anomaly Detection Algorithm Based on Clustering and Autoencoder MDPI URL: https://www.mdpi.com/2227-7390/11/15/3398 (Дата звернення: 06.05.2025)

10. Tavallaee, M.; Bagheri, E.; Lu, W.; Ghorbani, A.A. A detailed analysis of the KDD CUP 99 data set. In Proceedings of the IEEE International Conference on Computational Intelligence for Security & Defense Applications 2009, Ottawa, ON, Canada, 8–10 July 2009 (Дата звернення: 06.05.2025)

11. TOCA-IoT: Threshold Optimization and Causal Analysis for IoT Network Anomaly Detection Based on Explainable Random Forest MDPI URL: https://www.mdpi.com/1999-4893/18/2/117 (Дата звернення: 06.05.2025)

12. IoT Security Guidelines ENISA Report URL www.enisa.europa.eu. (Дата звернення: 07.05.2025)

13. Anomaly Detection in Time Series Using Statistical Analysis Medium URL: https://medium.com/booking-com-development/anomaly-detection-in-time-series-using-statistical-analysis-cc587b21d008 (Дата звернення: 07.05.2025)

14. Honeypots in Cybersecurity OWASP Foundation URL: www.owasp.org (Дата звернення: 07.05.2025)

15. Enhanced Anomaly Detection in IoT Networks Using Deep Autoencoders with Feature Selection Techniques NIH URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC12116047/ (Дата звернення: 08.05.2025)

16. Goyal S., Tripathi A. A review on IoT: Layered architecture, security issues and protocols. Int. J. Comput. Sci. Netw. Secur. (IJCSNS) 2023;23:100–107. (Дата звернення: 08.05.2025)

17. Doshi R., Apthorpe N., Feamster N. Machine learning ddos detection for consumer internet of things devices; Proceedings of the 2018 IEEE Security and Privacy Workshops (SPW); San Francisco, CA, USA. 24 May 2018; pp. 29–35 (Дата звернення: 08.05.2025)

18. Ganesh M., Kumar A., Pattabiraman V. Autoencoder Based Network Anomaly Detection; Proceedings of the 2020 IEEE International Conference on

Technology, Engineering, Management for Societal Impact Using Marketing, Entrepreneurship and Talent (TEMSMET); Bengaluru, India. 13–15 November 2020; Bengaluru, India: IEEE; 2020. pp. 1–6. (Дата звернення: 08.05.2025)

19. Sahu N.K., Mukherjee I. Machine learning based anomaly detection for iot network: (anomaly detection in iot network); Proceedings of the 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI) (48184); Tirunelveli, India. 15–17 June 2020; pp. 787–794. (Дата звернення: 08.05.2025)

20. Harnessing AI for Cyber Defense: Honeypot-Driven Intrusion Detection Systems MDPI URL: https://www.mdpi.com/2073-8994/17/5/628 (Дата звернення: 09.05.2025)

21. Stolfo, S.J.; Hershkop, S.; Bui, L.H.; Ferster, R.; Wang, K. Anomaly detection in computer security and an application to file system accesses. In Lecture Notes in Computer Science; Springer: Berlin/Heidelberg, Germany, 2005; pp. 14–28. (Дата звернення: 09.05.2025)

22. Albtosh, L.B. Advancements in cybersecurity and machine learning: A comprehensive review of recent research. World J. Adv. Eng. Technol. Sci. 2024, 13, 271–284. (Дата звернення: 09.05.2025)

23. Albaseer, A.; Abdi, N.; Abdallah, M.; Qaraqe, M.; Al-Kuwari, S. FedPot: A Quality-Aware Collaborative and Incentivized Honeypot-Based Detector for Smart Grid Networks. IEEE Trans. Netw. Serv. Manag. 2024, 21, 4844–4860. (Дата звернення: 09.05.2025)

24. Tian, W.; Ji, X.; Liu, W.; Liu, G.; Zhai, J.; Dai, Y.; Huang, S. Prospect theoretic study of honeypot defense against advanced persistent threats in power grid. IEEE Access 2020, 8, 64075–64085 (Дата звернення: 09.05.2025)

25. Investigating of Deep Learning-based Approaches for Anomaly Detection in IoT Surveillance Systems SAI URL: https://thesai.org/Publications/ViewPaper?Code=IJACSA&Issue=12&SerialNo=79&Volume=14 (Дата звернення: 10.05.2025)

26. A honeypot based cyber attack detection system for IoT devices ResearchGate URL: https://www.researchgate.net/publication/375689421_A_HONEYPOT_BASED_CYBER_ATTACK_DETECTION_SYSTEM_FOR_IOT_DEVICES (Дата звернення: 11.05.2025)

27. J. Voas and G. Hurlburt, "Internet of Things security," IT Professional, vol. 16, no. 3, pp. 8-11, 2014. (Дата звернення: 11.05.2025)

28. R.K. Mishra, V. M. Thool, and A. S. Dahiya, "HoneyIoT: A machine learning-based intrusion detection system for IoT devices using honeypot," in 2018 International Conference on evices using honeypot," in 2018 International Conference on Inventive Research in Computing Applications (ICIRCA), 2018, pp. 1703-1708. (Дата звернення: 11.05.2025)

29. P. K. Varshney, "On the safety of the internet of things," IT Professional, vol. 18, no. 2, pp. 38-41, 2016. (Дата звернення: 12.05.2025)

30. M.Ali, A. Arshad, and M. U. Ilyas, "Internet of things security: A survey," International Journal of Computer Networks and Communications Security, vol. 8, no. 4, pp. 140-147, 2020. (Дата звернення: 12.05.2025)

31. An outlier ensemble for unsupervised anomaly detection in honeypots data ResearchGate URL: https://www.researchgate.net/publication/343229320_An_outlier_ensemble_for_unsupervised_anomaly_detection_in_honeypots_data (Дата звернення: 13.05.2025)

32. A.Grégio, R.Santos and A.Montes, Evaluation of data mining techniques for suspicious network activity classificationusing honeypots data, Defense and Security Symposium International Society for Optics and Photonics, 2007 (Дата звернення: 13.05.2025)

33. A.Lazarevic, L.Ertoz, V.Kumar, A. Ozgur and J.Srivastava, A comparative study of anomaly detection schemes in networkintrusion detection,Proc. of the 2003 SIAM Int. Conf. on Data Mining, 2003 (Дата звернення: 14.05.2025)

34. G.K.Sadasivam,C.Hota and B.Anand, Detection of Severe SSH Attacks Using Honeypot Servers and Machine LearningTechniques, Software networking, (1)(2017), 79–100. (Дата звернення: 14.05.2025)

35. J.Song, H.Takakura, Y.Okabe, M.Eto, D.Inoue and K.Nakao, Statistical analysis of honeypot data and building of kyoto2006+ dataset for nids evaluation, BADGERS ’11 Proceedings of the First Workshop on Building Analysis Datasets andGathering Experience Returns for Security, 2011, 29–36 (Дата звернення: 14.05.2025)

36. Multi-Attention-Guided Cascading Network for End-to-End Person Search MDPI URL: https://www.mdpi.com/2076-3417/13/9/5576 (Дата звернення: 14.05.2025)

37. Wang, H.; Wang, Y.; Zhang, Z.; Fu, X.; Zhuo, L.; Xu, M.; Wang, M. Kernelized multiview subspace analysis by self-weighted learning. IEEE Trans. Multimed. 2020, 23, 3828–3840. (Дата звернення: 14.05.2025)

38. Wang, H.; Yao, M.; Jiang, G.; Mi, Z.; Fu, X. Graph-Collaborated Auto-Encoder Hashing for Multiview Binary Clustering. IEEE Trans. Neural Netw. Learn. Syst. 2023, 1–13. (Дата звернення: 14.05.2025)

39. Qian, B.; Wang, Y.; Yin, H.; Hong, R.; Wang, M. Switchable Online Knowledge Distillation. In Proceedings of the European Conference on Computer Vision (ECCV), Tel Aviv, Israel, 23–27 October 2022. (Дата звернення: 14.05.2025)

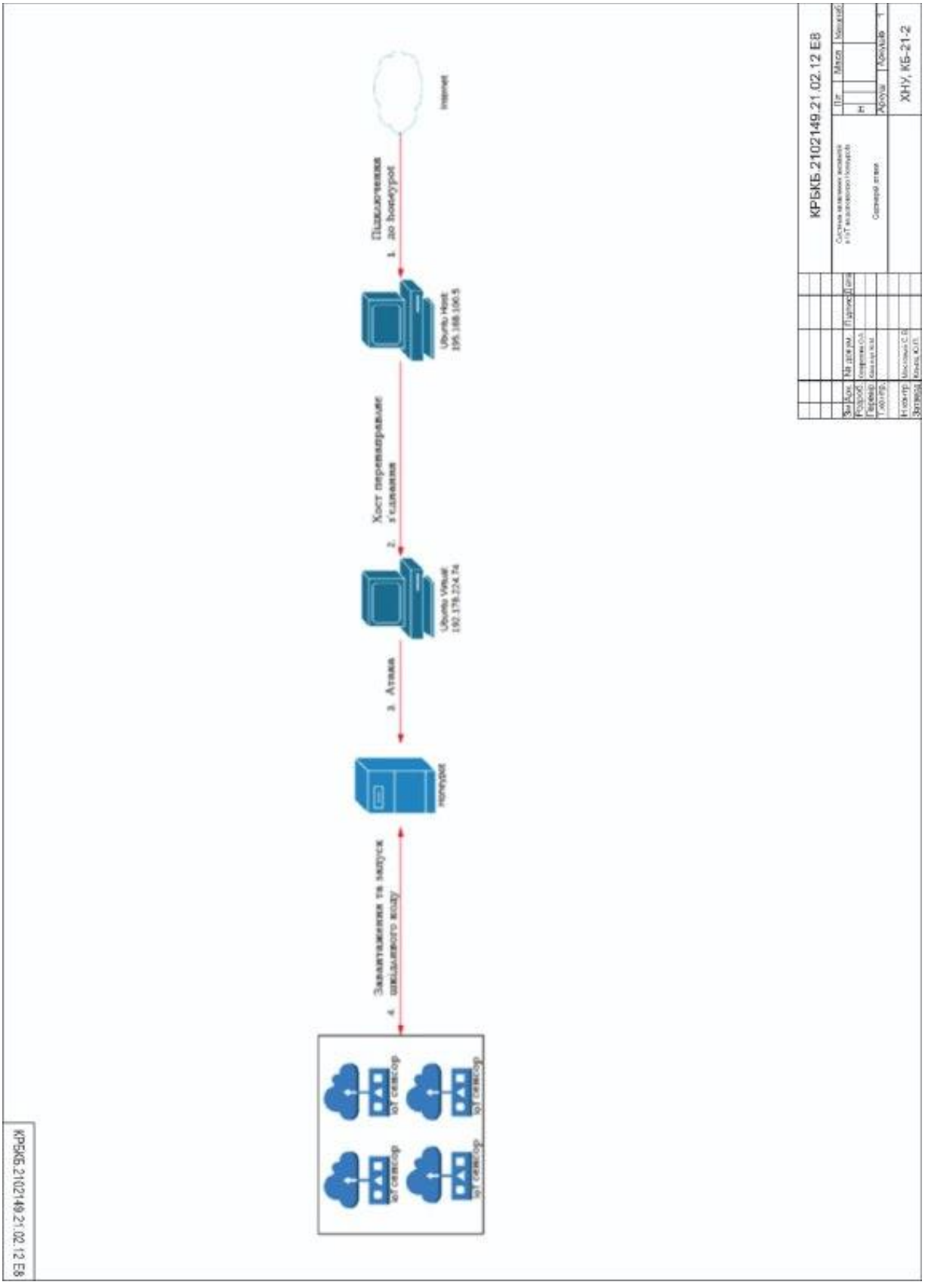
40. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587. (Дата звернення: 14.05.2025)

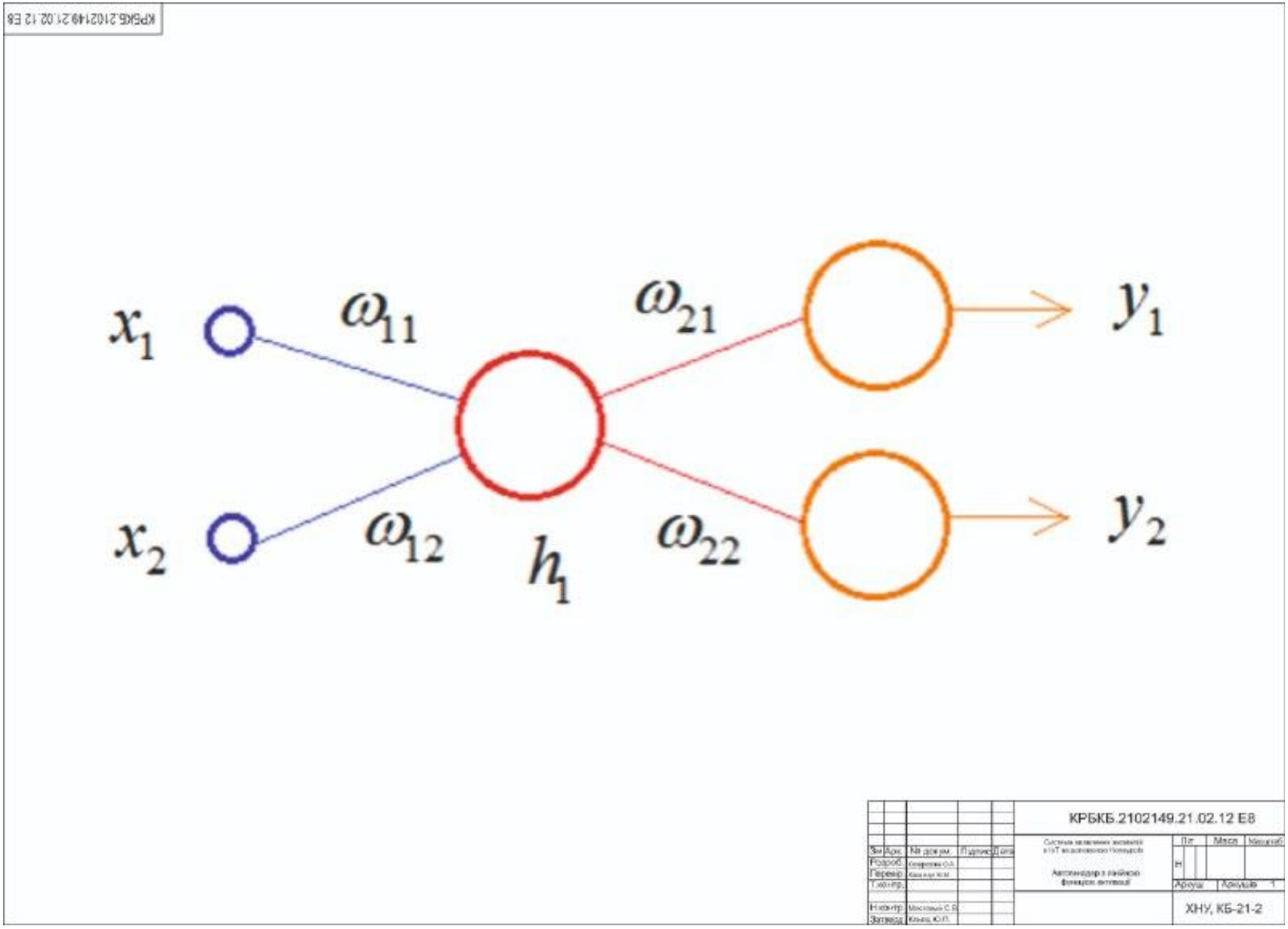
41. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montreal, QC, Canada, 7–12 December 2015. (Дата звернення: 14.05.2025)

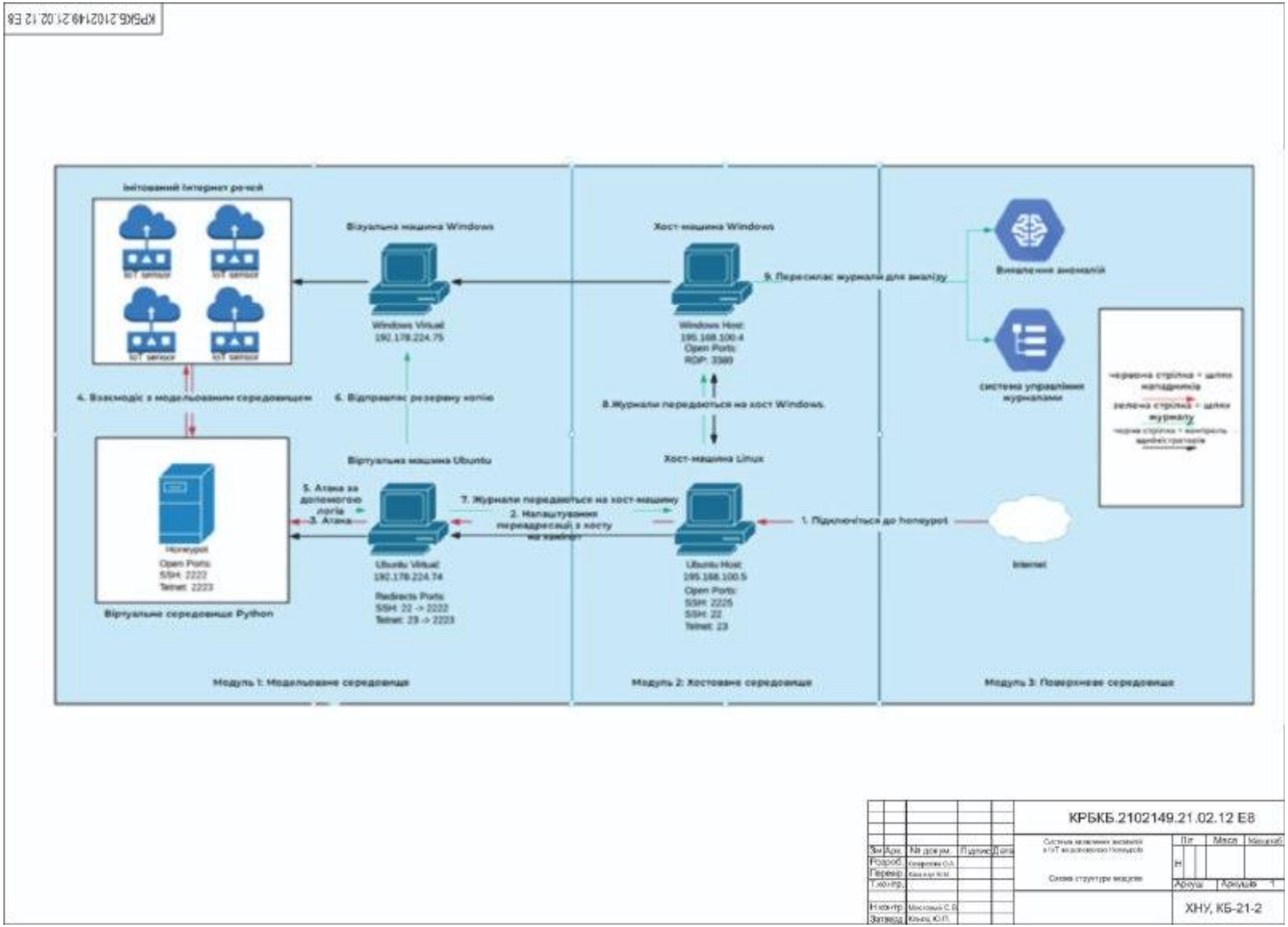
Додаток А
Програмний код

```
import pandas as pd
import json
from sklearn.ensemble import IsolationForest
from datetime import datetime

logs = []
with open("cowrie.json", "r") as f:
    for line in f:
        try:
            log = json.loads(line.strip())
            if log.get("eventid") in ["cowrie.login.success", "cowrie.login.failed",
"cowrie.command.input"]:
                logs.append(log)
        except json.JSONDecodeError:
            continue

df = pd.DataFrame(logs)
df["timestamp"] = pd.to_datetime(df["timestamp"], errors="coerce")
df["input"] = df["input"].fillna("")


sessions = df.groupby("session").agg({
    "src_ip": "first",
    "timestamp": ["min", "max"],
    "input": lambda x: ' '.join(x)
}).reset_index()

sessions.columns = ["session", "src_ip", "start_time", "end_time", "commands"]
sessions["duration"] = (sessions["end_time"] - sessions["start_time"]).dt.total_seconds()
sessions["num_cmds"] = sessions["commands"].apply(lambda x: len(x.split()))
ip_counts = sessions["src_ip"].value_counts()
sessions["ip_freq"] = sessions["src_ip"].apply(lambda ip: ip_counts[ip])


features = sessions[["duration", "num_cmds", "ip_freq"]].fillna(0)
model = IsolationForest(contamination=0.3, random_state=42)
sessions["anomaly"] = model.fit_predict(features)
sessions["anomaly"] = sessions["anomaly"].apply(lambda x: "аномалія" if x == -1 else
"норма")
```

Додаток Б
Копії графічної частини




КРБКБ.2102149.21.02.12 Е8
x_1
x_2
ω_{11}
ω_{12}
h_1
ω_{21}
ω_{22}
y_1
y_2
Автокодер з лінійною функцією активації
ХНУ, КБ-21-2


КРБКБ.2102149.21.02.12 Е8
Імітований Інтернет речей
IoT sensor
IoT sensor
IoT sensor
IoT sensor
Віртуальна машина Windows
Windows Virtual
192.178.224.75
Хост-машина Windows
Windows Host
195.168.100.4
Open Ports:
RDP: 3389
9. Пересилає журнали для аналізу
Виявлення аномалій
система управління журналами
4. Взаємодіє з модельованим середовищем
6. Відправляє резервну копію
8.Журнали передаються на хост Windows
Віртуальна машина Ubuntu
Хост-машина Linux
5. Атака за допомогою логів
3. Атака
7. Журнали передаються на хост-машину
2. Налаштування переадресації з хосту на хамінт
1. Підключиться до honeypot
Internet
Honeypot
Open Ports:
SSH: 2222
Telnet: 2223
Віртуальне середовище Python
Ubuntu Virtual
192.178.224.74
Redirects Ports:
SSH: 22 -> 2222
Telnet: 23 -> 2223
Ubuntu Host
195.168.100.5
Open Ports:
SSH: 2225
SSH: 22
Telnet: 23
Модуль 1: Модельоване середовище
Модуль 2: Хостоване середовище
Модуль 3: Поверхневе середовище
КРБКБ.2102149.21.02.12 Е8
Зм Арк.
№ докум.
Підпис
Дата
Розроб.
Перевір.
Т.контр.
Н.контр.
Затверд.
Схема структури модулів
Літ.
Маса
Масштаб
Аркуш
Аркушів
ХНУ, КБ-21-2

Завідувачу кафедри кібербезпеки
к.т.н., доц. Кльоцу Ю.П.
Козарєзова Олександра Андріївна
ПІБ здобувача вищої освіти

Студентка ФІТ, 4 курсу, групи КБ-21-2

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у Хмельницькому національному університеті» від 31.08.2023, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлена. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщена та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Unicheck та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

07.06.25

Протокол аналізу звіту подібності науковим керівником

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Козарєзова Олександра Андріївна

Співавтор:

Назва: Система виявлення аномалій в IoT за допомогою Honeypots

Науковий керівник:

Підрозділ: Кафедра кібербезпеки

Коефіцієнт подібності 1:7.7%

Коефіцієнт подібності 2:1.5%

Мікропробіли: 0

Заміна букв: 0

Інтервали: 0

Білі знаки: 0

Дата створення звіту: 2025-06-12 11:36:49.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

12.06.2025р.

Anti-Plagiarism (UA) v-15.281 Educational

The maximum coincidence with one document 1.0%

Dictionaries check: en_US, ru_RU, ua_UA. **Errors in the documents: 9%**

ID: 244849 Title: Система виявлення аномалій в IoT за допомогою Honeypots Added in a DB: 2025-06-10 Authors: Козарєзова Олександра Андріївна Heads: Касянчук М.М. Consultants: Opponents:	Document		Sum coincidence on the DB	
	Symbols	Lexemes	Symbols	Lexemes
	74509	536	408 (1%)	4 (1%)

Plagiarism sources

ID	Description	Plagiarism presence in the document	
		Symbols	Lexemes

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 аномалій в IoT за допомогою Honeypots

Автор: Козарєзова Олександра Андріївна

Спеціальність: 125 – Кібербезпека

Освітня програма:Кібербезпека

Науковий керівник:Михайло КАСЯНЧУК, докт. техн. наук, професор

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) запозичення розміщені в розділах аналізу існуючих методів та технологій, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;
3) всі зафіксовані системою ознаки модифікації тексту відносяться до комбінування латинських символів зі україномовними скороченнями індексів в формулах, що не є модифікацією тексту.

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості, складає 7.7% системою StrikePlagiarism, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ Михайло КАСЯНЧУК

Завідувач кафедри Кб _____ Юрій КЛЬОЦ

Гарант ОП _____ Віктор ЧЕШУН

Дата:

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

освітнього ступеня «бакалавр»

Студентка Олександра Козарєзова Андріївна

Тема Система виявлення аномалій в IoT за допомогою Honeypots

Спеціальність 125 – Кібербезпека

Обсяг кваліфікаційної роботи освітньо-кваліфікаційного рівня «бакалавр»:

кількість листів креслень 3; кількість сторінок записки 59.

1. Короткий зміст роботи та прийнятих рішень У кваліфікаційній роботі реалізовано симульовану інфраструктуру для виявлення аномальної активності в середовищі IoT-пристроїв. Проведено тестування атаки із подальшим аналізом логів і фіксацією поведінки умовного зловмисника. Розроблено сценарії атак, змодельовано інфраструктуру офісного IoT-середовища та описано ключові етапи взаємодії порушника з системою.

2. Висновок про відповідність кваліфікаційної роботи завданню Кваліфікаційна робота повністю відповідає поставленому завданню та охоплює як теоретичні, так і практичні аспекти побудови системи виявлення аномалій у кіберфізичних середовищах.

3. Характеристика виконання кожного розділу роботи, ступінь використання останніх досягнень науки і техніки і передових методів роботи: У розділі 1 розглянуто загальні принципи побудови honeypot-систем, їх класифікацію та особливості застосування в кібербезпеці IoT-пристроїв. Висвітлено актуальні загрози, протоколи комунікації та технічні засади побудови симуляцій. У розділі 2 описано архітектуру модульного середовища: побудову honeypot'а, симуляцію MQTT-пристроїв, структуру мережі, налаштування логування. У розділі 3 наведено аналіз журналів зловмисної активності, класифікацію атак, типові інструменти, виявлені в процесі тестування, та способи імітації правдоподібної файлової системи.

4. Позитивні сторони роботи Робота містить чітко структуровану симуляційну модель, наближену до реальних умов корпоративної мережі з IoT-пристроями. Практична частина демонструє глибоке розуміння інструментів аналізу кіберінцидентів. Висвітлено сучасні підходи до виявлення аномалій та забезпечення інформаційної безпеки.

5. Негативні сторони роботи Недостатньо детально описано механізми обробки виявлених аномалій та не запропоновано алгоритмів реакції на різні типи вторгнень. Також у роботі можна було б розширити опис векторів атак, характерних для сучасних IoT-будинків.

6. Оцінка графічного оформлення та пояснювальної записки роботи Графічні матеріали оформлено згідно з вимогами, з використанням діаграм, схем мережі, фрагментів логів та ілюстрацій з інтерфейсу Splunk. Пояснювальна записка оформлена грамотно, відповідає вимогам до бакалаврських робіт.

7. Відгук про роботу в цілому Кваліфікаційна робота заслуговує задовільно оцінки. Вона поєднує як дослідницький, так і прикладний характер, містить повноцінну реалізацію та тестування системи, що відповідає реальним викликам сучасної кібербезпеки. Логічна послідовність, якісна структуризація матеріалу та грамотне подання сприяють легкому сприйняттю.

8. Інші зауваження _____
_____
_____
_____

9. Оцінка кваліфікаційної роботи Враховуючи високий рівень технічної реалізації, структурованість викладу, аналітичність підходу та актуальність тематики, рекомендованою оцінкою є «задовільно».

РЕЦЕНЗЕНТ (прізвище, ім'я, по батькові, посада, місце роботи)_____
Нічепорук Андрій Олександрович _____
кандидат технічних наук, доцент кафедри комп'ютерної інженерії та інформаційних систем

«12» 06 2025. _____ (підпис)